Lecture Notes in Artificial Intelligence 5476

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

Thanaruk Theeramunkong
Boonserm Kijsirikul Nick Cercone
Tu-Bao Ho (Eds.)

Advances in Knowledge Discovery and Data Mining

13th Pacific-Asia Conference, PAKDD 2009
Bangkok, Thailand, April 27-30, 2009
Proceedings

Springer

Series Editors

Randy Goebel, University of Alberta, Edmonton, Canada
Jörg Siekmann, University of Saarland, Saarbrücken, Germany
Wolfgang Wahlster, DFKI and University of Saarland, Saarbrücken, Germany

Volume Editors

Thanaruk Theeramunkong
Thammasat University
Sirindhorn International Institute of Technology
131 Moo 5 Tiwanont Road, Bangkadi, Muang, Pathumthani 12000, Thailand
E-mail: thanaruk@siit.tu.ac.th

Boonserm Kijsirikul
Chulalongkorn University
Faculty of Engineering, Department of Computer Engineering
Bangkok 10330, Thailand
E-mail: boonserm@cp.eng.chula.ac.th

Nick Cercone
York University, Faculty of Science & Engineering
355 Lumbers Building, 4700 Keele Street, Toronto ON M3J 1P3, Canada
E-mail: ncercone@yorku.ca

Tu-Bao Ho
Japan Advanced Institute of Science and Technology
School of Knowledge Science
1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan
E-mail: bao@jaist.ac.jp

Library of Congress Control Number: Applied for

CR Subject Classification (1998): I.2, H.2.8, H.3, H.5.1, G.3, J.1, K.4

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-642-01306-6 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-01306-5 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

springer.com

© Springer-Verlag Berlin Heidelberg 2009
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12663194 06/3180 5 4 3 2 1 0

Preface

The Pacific-Asia Conference on Knowledge Discovery and Data Mining has been held every year from 1997. PAKDD 2009, the 13th in the series, was held in Bangkok, Thailand during April 27-30, 2008. PAKDD is a major international conference in the areas of data mining (DM) and knowledge discovery in database (KDD). It provides an international forum for researchers and industry practitioners to share their new ideas, original research results and practical development experiences from all KDD-related areas including data mining, data warehousing, machine learning, databases, statistics, knowledge acquisition and automatic scientific discovery, data visualization, causal induction and knowledge-based systems.

For PAKDD 2009, we received 338 research papers from various countries and regions in Asia, Australia, North America, South America, Europe, and Africa. Every submission was rigorously reviewed by at least three reviewers with a double blind protocol. The initial results were discussed among the reviewers and finally judged by the Program Committee Chairs. When there was a conflict, an additional review was provided by the Program Committee Chairs. The Program Committee members were deeply involved in the highly selective process. As a result, only 39 papers (approximately 11.5% of the 338 submitted papers) were accepted as regular papers, 73 papers (21.6% of them) were accepted as short papers.

The PAKDD 2009 conference program also included five workshops: the Pacific Asia Workshop on Intelligence and Security Informatics (PAISI 2009), a workshop on Advances and Issues in Biomedical Data Mining (AIBDM 2009), a workshop on Data Mining with Imbalanced Classes and Error Cost (ICEC 2009), a workshop on Open Source in Data Mining (OSDM 2009), and a workshop on Quality Issues, Measures of Interestingness and Evaluation of data mining models (QIMIE 2009). PAKDD 2009 would not have been successful without the support of committee members, reviewers, workshop organizers, tutorial speakers, invited speakers, competition organizers, organizing staffs, and supporting organizations. We are indebted to the members of Steering Committee for their invaluable suggestions and support throughout the organization process. We highly appreciate the Program Committee members, and external reviewers for their technical effort in providing straightforward scientific comments and impartial judgments in the review process of PAKDD 2009. We thank our Tutorial Co-chairs Vincent S. Tseng and Shusaku Tsumoto for kindly coordinating the fruitful tutorials. We wish to thank our General Workshop Co-chairs Manabu Okumura and Bernhard Pfahringe for selecting and coordinating the great workshops. Many thanks are given to the distinguished keynote speakers, invited speakers and tutorial presenters for their attractive and motivational talks and lectures. We thank the General Co-chairs Masaru Kitsuregawa and Vilas

Wuwongse for their useful guidance and their sharp advice in various spectrums related to the conference arrangements. We are also grateful to the Local Arrangements Chair Chotirat Ratanamahatana and our Local Arrangements Committee in both Thammasat University and Chulalongkorn University for their unlimited help toward the success of the conference. Last but not the least, we would like to give special thanks to Cholwich Nattee, who arranged the publication of PAKDD 2009 in the *Lecture Notes in Computer Science* series, and to Wirat Chinnan and Swit Phuvipadawat for their support of the PAKDD 2009 conference website. While the arrangement of PAKDD 2009 involved so many people, we would like to extend an additional thank-you to those who contributed to PAKDD 2009 but whose names may not be listed.

We greatly appreciate the support from various institutions. The conference was organized by the Sirindhorn International Institute of Technology (SIIT), Thammasat University (TU) and co-organized by the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University (CU), and the Asian Institute of Technology (AIT). It was sponsored by the National Electronics and Computer Technology Center (NECTEC, Thailand), the Thailand Convention and Exhibition Bureau (TCEB), and the Air Force Office of Scientific Research/Asian Office of Aerospace Research and Development (AFOSR/AOARD).

Finally, we wish to thank all authors and all conference participants for their contribution and support. We hope all participants took this opportunity to share and exchange ideas with each other and enjoyed PAKDD 2009 in the wonderful city of Bangkok.

February 2009

Thanaruk Theeramunkong
Boonserm Kijsirikul
Nick Cercone
Ho Tu Bao

Organization

PAKDD 2009 is organized by Sirindhorn International Institute of Technology of Thammasat University, Department of Computer Engineering of Chulalongkorn University, and the School of Engineering and Technology of Asian Institute of Technology.

PAKDD 2009 Conference Committee

Honorary Chairs

David Cheung	University of Hong Kong, China
Hiroshi Motoda	Osaka University, Japan

Local Honorary Chairs

Surapon Nitikraipot	Rector of Thammasat University, Thailand
Pirom Kamolratanakul	Rector of Chulalongkorn University, Thailand
Said Irandoust	President of AIT, Thailand

General Chairs (Conference Chairs)

Masaru Kitsuregawa	Tokyo University, Japan
Vilas Wuwongse	Asian Institute of Technology, Thailand

Program Committee Chairs

Thanaruk Theeramunkong	SIIT, Thammasat University, Thailand
Boonserm Kijsirikul	Chulalongkorn University, Thailand
Nick Cercone	York University, Canada
Ho Tu Bao	JAIST, Japan

Workshop Chairs

Manabu Okumura	Tokyo Institute of Technology, Japan
Bernhard Pfahringe	University of Waikato, New Zealand

Tutorial Chairs

Vincent S. Tseng	National Cheng Kung University, Taiwan
Shusaku Tsumoto	Shimane University, Japan

Journal Publication

Yasushi Sakurai	NTT, Japan
Nick Cercone	York University, Canada

Local Arrangements Committee Chair

Chotirat Ratanamahatana	Chulalongkorn University, Thailand

Members

Dararat Srisai	Chulalongkorn University, Thailand
Ithipan Methasate	SIIT, Thammasat University, Thailand
Juniar Ganis	SIIT, Thammasat University, Thailand
Kovit Punyasoponlert	Chulalongkorn University, Thailand
Nattapong Tongtep	SIIT, Thammasat University, Thailand
Nichnan Kittiphattanabawon	SIIT, Thammasat University, Thailand
Pakinee Aimmanee	SIIT, Thammasat University, Thailand
Pasakorn Tangchanachaianan	Chulalongkorn University, Thailand
Peerasak Intarapaiboon	SIIT, Thammasat University, Thailand
Piya Limcharoen	SIIT, Thammasat University, Thailand
Ratthachat Chatpatanasiri	Chulalongkorn University, Thailand
Sudchaya Saengthong	SIIT, Thammasat University, Thailand
Surapa Thiemjarus	SIIT, Thammasat University, Thailand
Swit Phuvipadawat	SIIT, Thammasat University, Thailand
Tanasanee Phienthrakul	Chulalongkorn University, Thailand
Thanasan Tanhermhong	SIIT, Thammasat University, Thailand
Thatsanee Charoenporn	NECTEC, Thailand
Thawatchai Suwannapong	SIIT, Thammasat University, Thailand
Vit Niennattrakul	Chulalongkorn University, Thailand
Warakorn Gulyanon	SIIT, Thammasat University, Thailand
Wirat Chinnan	SIIT, Thammasat University, Thailand

Publication Chairs

Cholwich Nattee	SIIT, Thammasat University, Thailand
Jakkrit TeCho	SIIT, Thammasat University, Thailand

Publicity Chairs

Chutima Pisarn	Prince of Songkla University, Thailand
Kritsada Sriphaew	Tokyo Institute of Technology, Japan
Thatsanee Charoenporn	NECTEC, Thailand

PAKDD 2009 Program Committee

Chairs and Co-chairs

Thanaruk Theeramunkong	SIIT, Thammasat University, Thailand
Boonserm Kijsirikul	Chulalongkorn University, Thailand
Nick Cercone	York University, Canada
Ho Tu Bao	JAIST, Japan

Members

Ah-Hwee Tan
Aidong Zhang
Aijun An
Aixin Sun
Akihiro Inokuchi
Akira Shimazu
Aleksandar Lazarevic
Alfredo Cuzzocrea
Alipio M. Jorge
Alok Choudhary
Amanda Clare
Ambuj K. Singh
Annalisa Appice
Anne M. Denton
Anthony Bagnall
Aris Anagnostopoulos
Ashkan Sami
Atsuyoshi Nakamura
Aurawan Imsombut
Baoning Wu
Beatriz de la Iglesia
Ben Kao
Benjamin C.M. Fung
Bernhard Pfahringer
Bettina Berendt
Bradley Malin
Carlos Alberto
Alejandro Castillo Ocaranza
Chai Wutiwiwatchai
Chandan Reddy
Chang-Tien Lu
Chaveevan Pechsiri
Chengkai Li
Chengqi Zhang
Chih-Jen Lin
Choochart Haruechaiyasak
Chotirat Ann Ratanamahatana
Christian Dawson
Christophe Giraud-Carrier
Chun-hung Li
Chung-Hong Lee
Chunsheng Yang
Chutima Pisarn
Claudio Lucchese
Clement Yu
Dacheng Tao
Daisuke Ikeda
Daniel C. Neagu
Dao-Qing Dai
Daoqiang Zhang
David Taniar
Daxin Jiang
Dejing Dou
Dell Zhang
Demetris Zeinalipour
Desheng Dash Wu
Di Wu
Diane Cook
Diansheng Guo
Dimitrios Katsaros
Dimitris Margaritis
Dit-Yan Yeung
Doina Caragea
Domenico Talia
Dou Shen
Dragan Gamberger
Du Zhang
Eamonn Keogh
Ee-Peng Lim
Eibe Frank
Evaggelia Pitoura
Evimaria Terzi
Fabian Moerchen
Fabio Roli
Fabrizio Silvestri
Feifei Li
Fernando Berzal
Francesco Bonchi
Francesco Masulli
Gabriel Fung
Gang Li
Gao Cong
Gemma Garriga
George Karypis
Georges Grinstein
Giovanni Semeraro
Giuseppe Manco
Graham Williams

Grigorios Tsoumakas
Guido Cervone
Guozhu Dong
Hai Wang
Haimonti Dutta
Hideo Bannai
Hiroki Arimura
Hiroyuki Kawano
Hiroyuki Kitagawa
Hisashi Kashima
Hisham Al-Mubaid
Hong Gao
Howard Ho
Hsin-Chang Yang
Hsin-Vonn Seow
Hua Lu
Hui Wang
Hui Xiong
Hui Yang
Huidong Jin
Huiyu Zhou
Hung Son Nguyen
Ira Assent
Ivor W. Tsang
Jaakko Hollmen
Jake Chen
Jan Ramon
Jan Rauch
Jason T.L. Wang
Jean-Gabriel Gustave Ganascia
Jean-Marc Petit
Jeremy Besson
Jialie Shen
Jian Yin
Jianyong Wang
Jieping Ye
Jimmy Huang
Jin Tian
Jing Peng
Jinyan Li
Jiong Yang
Joõ P. Gama
Joern Schneidewind
Johannes Fürnkranz
John Keane
Josep Domingo-Ferrer
Juggapong Natwichai
Junbin Gao
Jure Leskovec
K. Selcuk Candan
Kaidi Zhao
Kaiqi Huang
Kanishka Bhaduri
Kay Chen Tan
Keith C.C. Chan
Kevin Curran
Kitsana Waiyamai
Konstantinos Kalpakis
Kun Liu
Latifur Rahman Khan
Limsoon Wong
Lipo Wang
Lisa Hellerstein
Longbing Cao
Luis Torgo
Marco Maggini
Marut Buranarach
Masashi Shimbo
Masoud Jamei
Maybin Muyeba
Mehmet Koyuturk
Michael Schmidt
Michelangelo Ceci
Min Yao
Ming Hua
Mingli Song
Mithun Prasad
Mitsunori Ogihara
Mohamed F. Mokbel
Mohamed Medhat Gaber
Myra Spiliopoulou
N.Ch. Sriman Narayana Iyengar
Ngoc Thanh Nguyen
Nikunj Chandrakant Oza
Ning Zhong
Ninghui Li
Nucharee Premchaiswadi
Orlando De Jesus
Osman Abul
P.K. Mahanti

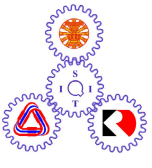
Panagiotis Karras
Pang-Ning Tan
Patricia Riddle
Paulo Cortez
Petra Kralj Novak
Petros Drineas
Philippe Lenca
Qingxiang Wu
Radha Krishna Murthy Karuturi
Raj Krishna Bhatnagar
Rajendra Akerkar
Rajesh Reghunadhan
Reda Alhajj
Richi Nayak
Ronald Rousseau
Rosa Meo
Rui Camacho
Ruoming Jin
Seiji Yamada
Salvatore Orlando
San-Yih Hwang
Sanjay Ranka
Sanparith Marukatat
Satoshi Oyama
Shen-Shyang Ho
Sheng Zhong
Shenghuo Zhu
Shichao Zhang
Shu-Ching Chen
Shun Ishizaki
Silvia Chiusano
Spiros Papadimitriou
Srikanta Tirthapura
Srinivasan Jagannathan
Stefan Rueping
Suman Nath
Sung Ho Ha
Surapa Thiemjarus
Szymon Jaroszewicz
Tadashi Nomoto
Takeaki Uno
Takehisa Yairi
Takenobu Tokunaga
Tamas Sarlos
Tamer Kahveci
Taneli Mielikäinen
Tansel Ozyer
Tanya Y. Berger-Wolf
Tao Li
Tao Mei
Tetsuya Yoshida
Thanaruk Theeramunkong
Themis Palpanas
Thepchai Supnithi
Tianhao Zhang
Tie-Yan Liu
Tim Oates
Tina Eliassi-Rad
Tom Croonenborghs
Tomoyuki Uchida
Torsten Suel
Toshihiro Kamishima
Toshiro Minami
Traian Marius Truta
Tru Cao
Tsuyoshi Murata
Ulf Brefeld
Vagelis Hristidis
Vasilis George Aggelis
Vasilis Megalooikonomou
Vassilis Athitsos
Vincent C.S. Lee
Vincent S. Tseng
Vincenzo Piuri
Virach Sortlertlamvanich
Wagner Meira Jr.
Wai Lam
Wei Fan
Wen-Chih Peng
Wenliang Du
Wilfred Ng
William K. Cheung
Wlodek Zadrozny
Wolfgang Lehner
Woong-Kee Loh
Wynne Hsu
Xiangjun Dong
Xiao-Lin Li
Xiaofeng Meng
Xiaohui Liu

Xiaolei Li
Xiaoli Li Li
Xiaowei Shao
Xindong Wu
Xingquan Zhu
Xintao Wu
Xue Li
Xuelong Li
Yan Zhou
Yang Xiang
Yang Zhang
Yang-Sae Moon
Yanwei Pang
Yasuhiko Morimoto
Yi Feng
Yi-Dong Shen
Yi-Ping Phoebe Chen
Yifeng Zeng
Yihua Wu
Ying Tan
Yiyu Yao
Yong Guan
Yuan Yuan
Yun Fu
Yutaka Matsuo
Zhanhuai Li
Zhaohui Tang
Zhaoyang Dong
Zheng Chen
Zhi-Hua Zhou
Zhongfei (Mark) Zhang
Zhuoming Xu

PAKDD 2009 External Reviewers

Daan He
Ioannis Katakis
Jiye Li
Ratthachat Chatpatanasiri
Xiangdong An

Organized by




Sirindhorn International Institute of Technology
Thammasat University



Chulalongkorn University



Asian Institute of Technology

Sponsoring Institutions



National Electronics and Computer Technology Center (NECTEC), Thailand



Thailand Convention and Exhibition Bureau (TCEB), Thailand





The Air Force Office of Scientific Research, Asian Office of Aerospace Research and Development (AFOSR/AOARD), USA

Table of Contents

Keynote Speeches

KDD for BSN – Towards the Future of Pervasive Sensing 1
Guang-Zhong Yang

Finding Hidden Structures in Relational Databases 2
Jeffrey Xu Yu

The Future of Search: An Online Content Perspective 3
Andrew Tomkins

Regular Papers

DTU: A Decision Tree for Uncertain Data 4
Biao Qin, Yuni Xia, and Fang Li

Efficient Privacy-Preserving Link Discovery 16
Xiaoyun He, Jaideep Vaidya, Basit Shafiq, Nabil Adam, Evimaria Terzi, and Tyrone Grandison

On Link Privacy in Randomizing Social Networks 28
Xiaowei Ying and Xintao Wu

Sentence-Level Novelty Detection in English and Malay 40
Agus T. Kwee, Flora S. Tsai, and Wenyin Tang

Text Categorization Using Fuzzy Proximal SVM and Distributional Clustering of Words 52
Mani Arun Kumar and Madan Gopal

Cool Blog Classification from Positive and Unlabeled Examples 62
Kritsada Sriphaew, Hiroya Takamura, and Manabu Okumura

Thai Word Segmentation with Hidden Markov Model and Decision Tree 74
Poramin Bheganan, Richi Nayak, and Yue Xu

An Efficient Method for Generating, Storing and Matching Features for Text Mining 86
Shing-Kit Chan and Wai Lam

Robust Graph Hyperparameter Learning for Graph Based Semi-supervised Classification 98
Krikamol Muandet, Sanparith Marukatat, and Cholwich Nattee

Regularized Local Reconstruction for Clustering 110
Jun Sun, Zhiyong Shen, Bai Su, and Yidong Shen

Clustering with Lower Bound on Similarity 122
Mohammad Al Hasan, Saeed Salem, Benjarath Pupacdi, and Mohammed J. Zaki

Approximate Spectral Clustering 134
Liang Wang, Christopher Leckie, Kotagiri Ramamohanarao, and James Bezdek

An Integration of Fuzzy Association Rules and WordNet for Document Clustering 147
Chun-Ling Chen, Frank S.C. Tseng, and Tyne Liang

Nonlinear Data Analysis Using a New Hybrid Data Clustering Algorithm 160
Ureerat Wattanachon, Jakkarin Suksawatchon, and Chidchanok Lursinsap

A Polynomial-Delay Polynomial-Space Algorithm for Extracting Frequent Diamond Episodes from Event Sequences 172
Takashi Katoh, Hiroki Arimura, and Kouichi Hirata

A Statistical Approach for Binary Vectors Modeling and Clustering 184
Nizar Bouguila and Khalid Daoudi

Multi-resolution Boosting for Classification and Regression Problems ... 196
Chandan K. Reddy and Jin-Hyeong Park

Interval Data Classification under Partial Information: A Chance-Constraint Approach 208
Sahely Bhadra, J. Saketha Nath, Aharon Ben-Tal, and Chiranjib Bhattacharyya

Negative Encoding Length as a Subjective Interestingness Measure for Groups of Rules 220
Einoshin Suzuki

The Studies of Mining Frequent Patterns Based on Frequent Pattern Tree 232
Show-Jane Yen, Yue-Shi Lee, Chiu-Kuang Wang, Jung-Wei Wu, and Liang-Yu Ouyang

Discovering Periodic-Frequent Patterns in Transactional Databases 242
Syed Khairuzzaman Tanbeer, Chowdhury Farhan Ahmed, Byeong-Soo Jeong, and Young-Koo Lee

Quantifying Asymmetric Semantic Relations from Query Logs by Resource Allocation 254
Zhiyuan Liu, Yabin Zheng, and Maosong Sun

Acquiring Semantic Relations Using the Web for Constructing Lightweight Ontologies 266
Wilson Wong, Wei Liu, and Mohammed Bennamoun

Detecting Abnormal Events via Hierarchical Dirichlet Processes 278
Xian-Xing Zhang, Hua Liu, Yang Gao, and Derek Hao Hu

Active Learning for Causal Bayesian Network Structure with Non-symmetrical Entropy 290
Guoliang Li and Tze-Yun Leong

A Comparative Study of Bandwidth Choice in Kernel Density Estimation for Naive Bayesian Classification 302
Bin Liu, Ying Yang, Geoffrey I. Webb, and Janice Boughton

Analysis of Variational Bayesian Matrix Factorization 314
Shinichi Nakajima and Masashi Sugiyama

Variational Bayesian Approach for Long-Term Relevance Feedback 327
Sabri Boutemedjet and Djemel Ziou

Detecting Link Hijacking by Web Spammers 339
Young-joo Chung, Masashi Toyoda, and Masaru Kitsuregawa

A Data Driven Ensemble Classifier for Credit Scoring Analysis 351
Nan-Chen Hsieh, Lun-Ping Hung, and Chia-Ling Ho

A Multi-partition Multi-chunk Ensemble Technique to Classify Concept-Drifting Data Streams 363
Mohammad M. Masud, Jing Gao, Latifur Khan, Jiawei Han, and Bhavani Thuraisingham

Parameter Estimation in Semi-Random Decision Tree Ensembling on Streaming Data 376
Peipei Li, Qianhui Liang, Xindong Wu, and Xuegang Hu

Exploiting the Block Structure of Link Graph for Efficient Similarity Computation 389
Pei Li, Yuanzhe Cai, Hongyan Liu, Jun He, and Xiaoyong Du

Online Feature Selection Algorithm with Bayesian ℓ_1 Regularization 401
Yunpeng Cai, Yijun Sun, Jian Li, and Steve Goodison

Feature Selection for Local Learning Based Clustering 414
Hong Zeng and Yiu-ming Cheung

RV-SVM: An Efficient Method for Learning Ranking SVM 426
Hwanjo Yu, Youngdae Kim, and Seungwon Hwang

A Kernel Framework for Protein Residue Annotation 439
Huzefa Rangwala, Christopher Kauffman, and George Karypis

Dynamic Exponential Family Matrix Factorization 452
Kohei Hayashi, Jun-ichiro Hirayama, and Shin Ishii

A Nonparametric Bayesian Learning Model: Application to Text and Image Categorization 463
Nizar Bouguila and Djemel Ziou

Short Papers

Safe-Level-SMOTE: Safe-Level-Synthetic Minority Over-Sampling TEchnique for Handling the Class Imbalanced Problem 475
Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap

Using Highly Expressive Contrast Patterns for Classification - Is It Worthwhile? 483
Elsa Loekito and James Bailey

Arif Index for Predicting the Classification Accuracy of Features and Its Application in Heart Beat Classification Problem 491
Muhammad Arif, Fayyaz A. Afsar, Muhammad Usman Akram, and Adnan Fida

UCI++: Improved Support for Algorithm Selection Using Datasetoids 499
Carlos Soares

Accurate Synthetic Generation of Realistic Personal Information 507
Peter Christen and Agus Pudjijono

An Efficient Approximate Protocol for Privacy-Preserving Association Rule Mining 515
Murat Kantarcioglu, Robert Nix, and Jaideep Vaidya

Information Extraction from Thai Text with Unknown Phrase Boundaries 525
Peerasak Intarapaiboon, Ekawit Nantajeewarawat, and Thanaruk Theeramunkong

A Corpus-Based Approach for Automatic Thai Unknown Word Recognition Using Ensemble Learning Techniques 533
Jakkrit TeCho, Cholwich Nattee, and Thanaruk Theeramunkong

A Hybrid Approach to Improve Bilingual Multiword Expression Extraction 541
Jianyong Duan, Mei Zhang, Lijing Tong, and Feng Guo

Addressing the Variability of Natural Language Expression in Sentence Similarity with Semantic Structure of the Sentences 548
Palakorn Achananuparp, Xiaohua Hu, and Christopher C. Yang

Scalable Web Mining with Newistic 556
Ovidiu Dan and Horatiu Mocian

Building a Text Classifier by a Keyword and Unlabeled Documents..... 564
Qiang Qiu, Yang Zhang, and Junping Zhu

A Discriminative Approach to Topic-Based Citation Recommendation 572
Jie Tang and Jing Zhang

Romanization of Thai Proper Names Based on Popularity of Usages 580
Akegapon Tangverapong, Atiwong Suchato, and Proadpran Punyabukkana

Budget Semi-supervised Learning 588
Zhi-Hua Zhou, Michael Ng, Qiao-Qiao She, and Yuan Jiang

When does Co-training Work in Real Data? 596
Charles X. Ling, Jun Du, and Zhi-Hua Zhou

Classification of Audio Signals Using a Bhattacharyya Kernel-Based Centroid Neural Network 604
Dong-Chul Park, Yunsik Lee, and Dong-Min Woo

Sparse Kernel Learning and the Relevance Units Machine 612
Junbin Gao and Jun Zhang

Pairwise Constrained Clustering for Sparse and High Dimensional Feature Spaces 620
Su Yan, Hai Wang, Dongwon Lee, and C. Lee Giles

Clustering Documents Using a Wikipedia-Based Concept Representation 628
Anna Huang, David Milne, Eibe Frank, and Ian H. Witten

An Instantiation of Hierarchical Distance-Based Conceptual Clustering for Propositional Learning 637
Ana Funes, Cesar Ferri, Jose Hernández-Orallo, and Maria José Ramírez-Quintana

Computing Substitution Matrices for Genomic Comparative Analysis ... 647
Minh Duc Cao, Trevor I. Dix, and Lloyd Allison

Mining Both Positive and Negative Impact-Oriented Sequential Rules from Transactional Data 656
Yanchang Zhao, Huaifeng Zhang, Longbing Cao, Chengqi Zhang, and Hans Bohlscheid

Aggregated Subset Mining 664
Albrecht Zimmermann and Björn Bringmann

Hot Item Detection in Uncertain Data 673
Thomas Bernecker, Hans-Peter Kriegel, Matthias Renz, and Andreas Zuefle

Spanning Tree Based Attribute Clustering 681
Yifeng Zeng, Jorge Cordero Hernandez, and Shuyuan Lin

The Effect of Varying Parameters and Focusing on Bus Travel Time Prediction 689
João M. Moreira, Carlos Soares, Alípio M. Jorge, and Jorge Freire de Sousa

Transfer Learning Action Models by Measuring the Similarity of Different Domains 697
Hankui Zhuo, Qiang Yang, and Lei Li

On Optimal Rule Mining: A Framework and a Necessary and Sufficient Condition of Antimonotonicity 705
Yannick Le Bras, Philippe Lenca, and Stéphane Lallich

Discovering Action Rules That Are Highly Achievable from Massive Data 713
Einoshin Suzuki

Extracting Fuzzy Rules for Detecting Ventricular Arrhythmias Based on NEWFM 723
Dong-Kun Shin, Sang-Hong Lee, and Joon S. Lim

Trace Mining from Distributed Assembly Databases for Causal Analysis 731
Shohei Hido, Hirofumi Matsuzawa, Fumihiko Kitayama, and Masayuki Numao

Let's Tango – Finding the Right Couple for Feature-Opinion Association in Sentiment Analysis 741
Kam Tong Chan and Irwin King

An Efficient Candidate Pruning Technique for High Utility Pattern Mining 749
Chowdhury Farhan Ahmed, Syed Khairuzzaman Tanbeer, Byeong-Soo Jeong, and Young-Koo Lee

Grouped ECOC Conditional Random Fields for Prediction of Web User Behavior 757
Yong Zhen Guo, Kotagiri Ramamohanarao, and Laurence A.F. Park

CLHQS: Hierarchical Query Suggestion by Mining Clickthrough Log ... 764
Depin Chen, Ning Liu, Zhijun Yin, Yang Tong, Jun Yan, and Zheng Chen

X-Tracking the Changes of Web Navigation Patterns 772
Long Wang and Christoph Meinel

Tree-Based Method for Classifying Websites Using Extended Hidden Markov Models 780
Majid Yazdani, Milad Eftekhar, and Hassan Abolhassani

Emotion Recognition of Pop Music Based on Maximum Entropy with Priors 788
Hui He, Bo Chen, and Jun Guo

Simultaneously Finding Fundamental Articles and New Topics Using a Community Tracking Method 796
Tieyun Qian, Jaideep Srivastava, Zhiyong Peng, and Phillip C.Y. Sheu

Towards a Novel Association Measure via Web Search Results Mining 804
Xiaojun Wan and Jianguo Xiao

A New Local Distance-Based Outlier Detection Approach for Scattered Real-World Data 813
Ke Zhang, Marcus Hutter, and Huidong Jin

Mining Outliers with Faster Cutoff Update and Space Utilization 823
Chi-Cheong Szeto and Edward Hung

Outlier Detection in Axis-Parallel Subspaces of High Dimensional Data 831
Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek

K-Dominant Skyline Computation by Using Sort-Filtering Method 839
Md. Anisuzzaman Siddique and Yasuhiko Morimoto

Effective Boosting of Naïve Bayesian Classifiers by Local Accuracy Estimation 849
Zhipeng Xie

COMUS: Ontological and Rule-Based Reasoning for Music Recommendation System 859
Seungmin Rho, Seheon Song, Eenjun Hwang, and Minkoo Kim

Spatial Weighting for Bag-of-Visual-Words and Its Application in Content-Based Image Retrieval 867
Xin Chen, Xiaohua Hu, and Xiajiong Shen

Item Preference Parameters from Grouped Ranking Observations 875
Hideitsu Hino, Yu Fujimoto, and Noboru Murata

Cross-Channel Query Recommendation on Commercial Mobile Search Engine: Why, How and Empirical Evaluation 883
Shunkai Fu, Bingfeng Pi, Ying Zhou, Michel C. Desmarais, Weilei Wang, Song Han, and Xunrong Rao

Data Mining for Intrusion Detection: From Outliers to True Intrusions 891
Goverdhan Singh, Florent Masseglia, Céline Fiot, Alice Marascu, and Pascal Poncelet

A Multi-resolution Approach for Atypical Behaviour Mining 899
Alice Marascu and Florent Masseglia

Change Analysis in Spatial Data by Combining Contouring Algorithms with Supervised Density Functions 907
Chun Sheng Chen, Vadeerat Rinsurongkawong, Christoph F. Eick, and Michael D. Twa

Centroid Neural Network with Spatial Constraints 915
Dong-Chul Park

Diversity in Combinations of Heterogeneous Classifiers 923
Kuo-Wei Hsu and Jaideep Srivastava

Growth Analysis of Neighbor Network for Evaluation of Damage Progress 933
Ken-ichi Fukui, Kazuhisa Sato, Junichiro Mizusaki, Kazumi Saito, Masahiro Kimura, and Masayuki Numao

A Parallel Algorithm for Finding Related Pages in the Web by Using Segmented Link Structures 941
Xiaoyan Shen, Junliang Chen, Xiangwu Meng, Yujie Zhang, and Chuanchang Liu

Boosting Biomedical Information Retrieval Performance through Citation Graph: An Empirical Study 949
Xiaoshi Yin, Xiangji Huang, Qinmin Hu, and Zhoujun Li

Similarity-Based Feature Selection for Learning from Examples with Continuous Values 957
Yun Li, Su-Jun Hu, Wen-Jie Yang, Guo-Zi Sun, Fang-Wu Yao, and Geng Yang

Application-Independent Feature Construction from Noisy Samples 965
Dominique Gay, Nazha Selmaoui, and Jean-François Boulicaut

Estimating Optimal Feature Subsets Using Mutual Information Feature Selector and Rough Sets 973
Sombut Foitong, Pornthep Rojanavasu, Boonwat Attachoo, and Ouen Pinngern

Speeding Up Similarity Search on a Large Time Series Dataset under Time Warping Distance 981
Pongsakorn Ruengronghirunya, Vit Niennattrakul, and Chotirat Ann Ratanamahatana

A Novel Fractal Representation for Dimensionality Reduction of Large Time Series Data 989
Poat Sajjipanon and Chotirat Ann Ratanamahatana

Clustering Data Streams in Optimization and Geography Domains 997
Ling-Yin Wei and Wen-Chih Peng

CBDT: A Concept Based Approach to Data Stream Mining 1006
Stefan Hoeglinger, Russel Pears, and Yun Sing Koh

Meaningful Subsequence Matching under Time Warping Distance for Data Stream 1013
Vit Niennattrakul and Chotirat Ann Ratanamahatana

An Aggregate Ensemble for Mining Concept Drifting Data Streams with Noise 1021
Peng Zhang, Xingquan Zhu, Yong Shi, and Xindong Wu

On Pairwise Kernels: An Efficient Alternative and Generalization Analysis 1030
Hisashi Kashima, Satoshi Oyama, Yoshihiro Yamanishi, and Koji Tsuda

A Family-Based Evolutional Approach for Kernel Tree Selection in SVMs 1038
Ithipan Methasate and Thanaruk Theeramunkong

An Online Incremental Learning Vector Quantization 1046
Ye Xu, Shen Furao, Osamu Hasegawa, and Jinxi Zhao

On Mining Rating Dependencies in Online Collaborative Rating Networks 1054
Hady W. Lauw, Ee-Peng Lim, and Ke Wang

Learning to Extract Relations for Relational Classification 1062
Steffen Rendle, Christine Preisach, and Lars Schmidt-Thieme

Author Index 1073

KDD for BSN – Towards the Future of Pervasive Sensing

Guang-Zhong Yang

Imperial College London

Abstract. With increasing sophistication and miniaturisation of wireless sensor technologies, integrated microsensors no more than a few millimetres in size combined with onboard processing and wireless data transfer has become a reality. The provision of ubiquitous and pervasive monitoring of physical, physiological, and biochemical parameters in any environment and without activity restriction and behaviour modification is the primary motivation of Body Sensor Network (BSN) research. The general scope of BSN is broad, ranging from monitoring of patients with chronic disease and care for the elderly, to general well-being monitoring and performance evaluation in sports. It also has important applications in gaming and human-computer-interaction. One of the significant challenges of BSN is the provision of context aware sensing with effective multi-sensor fusion, data inferencing, mining, and trend analysis. Other research issues currently being addressed include novel miniaturised bioelectrical, biochemical, biophysical, and mechanical sensors; low power RF transceiver, energy scavenging, and battery technologies; biocompatibility, materials, system integration and miniaturisation; autonomic sensor networks and light-weight communication protocols and standards. This talk will address some of the key research topics and current advances in BSN, particularly those related to the KDD community. It will also cover the use of bio-inspired design for providing distributed inferencing and ultra-low power on-node processing, demonstrating how this alternate paradigm based on the strategies used by biological systems can be used to deal with the challenges of scale, complexity, heterogeneity, and uncertainty involved in pervasive sensing.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, p. 1, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Finding Hidden Structures in Relational Databases

Jeffrey Xu Yu

The Chinese University of Hong Kong, Hong Kong

Abstract. Relational database management systems have been widely used over decades. An important research issue is to find hidden structural information in large relational databases. By hidden structural information we mean the information that cannot be easily found using a traditional query language SQL. In this talk, we discuss how to find hidden structural information in a relational database by viewing a relational database as a large directed graph where nodes represent tuples and edges represent foreign key references between tuples in the database. We discuss how to find trees and communities in such a large graph for user-given keywords. We also discuss how to find frequent and additional keywords associated with the structures identified in a relational database using SQL.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, p. 2, 2009.
© Springer-Verlag Berlin Heidelberg 2009

The Future of Search: An Online Content Perspective

Andrew Tomkins

Yahoo! Research

Abstract. Nonprofessional creation of public online content has outstripped professional content creation of all forms, both online and offline. And two orders of magnitude more content is created daily to flow through social networks, with as much as two more orders of magnitude still to come as user engagement increases. Content is diversifying in creation, consumption, and nature. Web search engines provide rapid targeted access to this page content, and increasingly to other information such as news articles, weather, movie showtimes, and product and restaurant listings. In this talk, I'll discuss these trends from the standpoint of the search engine, I'll cover some research results in this area, and I'll close with some challenges for the future.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, p. 3, 2009.
© Springer-Verlag Berlin Heidelberg 2009

DTU: A Decision Tree for Uncertain Data

Biao Qin[1], Yuni Xia[1], and Fang Li[2]

[1] Department of Computer and Information Science, Indiana University - Purdue University Indianapolis, USA
{biaoqin,yxia}@cs.iupui.edu
[2] Department of Mathematics, Indiana University - Purdue University Indianapolis, USA
lfang@iupui.edu

Abstract. Decision Tree is a widely used data classification technique. This paper proposes a decision tree based classification method on uncertain data. Data uncertainty is common in emerging applications, such as sensor networks, moving object databases, medical and biological bases. Data uncertainty can be caused by various factors including measurements precision limitation, outdated sources, sensor errors, network latency and transmission problems. In this paper, we enhance the traditional decision tree algorithms and extend measures, including entropy and information gain, considering the uncertain data interval and probability distribution function. Our algorithm can handle both certain and uncertain datasets. The experiments demonstrate the utility and robustness of the proposed algorithm as well as its satisfactory prediction accuracy.

1 Introduction

Decision trees is a simple yet widely used method for classification and predictive modeling. A decision tree partitions data into smaller segments called terminal nodes. Each terminal node is assigned a class label. The non-terminal nodes, which include the root and other internal nodes, contain attribute test conditions to separate records that have different characteristics. The partitioning process terminates when the subsets cannot be partitioned any further using predefined criteria. Decision trees are used in many domains. For example, in database marketing, decision trees can be used to segment groups of customers and develop customer profiles to help marketers produce targeted promotions that achieve higher response rates.

This paper studies decision tree based classification methods for uncertain data. In many applications, data contains inherent uncertainty. A number of factors contribute to the uncertainty, such as the random nature of the physical data generation and collection process, measurement and decision errors, unreliable data transmission and data staling. For example, there are massive amounts of uncertain data in sensor networks, such as temperature, humidity, and pressure. Uncertainty can also arise in categorical data. For instance, a tumor is typically classified as benign or malignant in cancer diagnosis and treatment.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 4–15, 2009.
© Springer-Verlag Berlin Heidelberg 2009

In practice, it is often very difficult to accurately classify a tumor due to the experiment precision limitation. The lab results inevitably give false positives or false negatives some of the time. Therefore, doctors may often decide tumors to be benign or malignant with certain probability or confidence. [24]

Since data uncertainty is ubiquitous, it is important to develop classification models for uncertain data. In this paper, We focus on the decision tree based classification approach. We choose the decision tree because of its numerous positive features. Decision tree is simple to understand and interpret. It requires little data preparation, while some other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Decision tree can handle both numerical and categorical data, while many other techniques are usually specialized in analyzing datasets that have only one type of variable. Decision tree uses a white box model. If a given situation is observable in a model the explanation for the condition is easily explained by Boolean logic. Besides, it is possible to validate a decision tree model using statistical tests. Decision tree is also robust and scalable. It performs well with large data in a short period of time.

In this paper, we propose a new decision tree for classifying and predicting both certain and uncertain data (DTU). The main contributions of this paper are:

1. We integrate the uncertainty data model into the design of the decision tree.
2. We develop the DTU based on the widely used C4.5 classification tree so that it can handle both numerical and categorical data with uncertainty.
3. We prove through experiments that DTU has satisfactory performance even when the training data is highly uncertain.

This paper is organized as follows. In the next section, we will discuss related work. Section 3 describes the uncertaint data model. Section 4 shows the measures for identifying the best split for uncertain data. Section 5 illustrates the DTU algorithms in detail. The experimental results are shown in Section 6 and Section 7 concludes the paper.

2 Related Work

Classification is a well-studied area in data mining. Many classification algorithms have been proposed in the literature, such as decision tree classifiers [17], Bayesian classifiers [14], support vector machines (SVM) [20], artificial neural networks [3] and ensemble methods [9]. In spite of the numerous classification algorithms, building classification based on uncertain data has remained a great challenge. There are early work performed on developing decision trees when data contains missing or noisy values [11,15,18]. Various strategies have been developed to predict or fill missing attribute values. However, the problem studied in this paper is different from before - instead of assuming part of the data has missing or noisy values, we allow the whole dataset to be uncertain, and

the uncertainty is not shown as missing or erroneous values but represented as uncertain intervals and probability distribution functions. There are also some previous work performed on classifying uncertain data in various applications [4,10,12]. All of the above methods try to solve specific classification tasks instead of developing a general algorithm for classifying uncertain data. And Qin et al. [24] propose a rule-based classification algorithm for uncertain data.

Recently, more research has been conducted in uncertain data mining. Most of them focus on clustering uncertain data [8,13,16]. The key idea is that when computing the distance between two uncertain objects, the probability distributions of objects are used to compute the expected distance. Xia et al. [22] introduce a new conceptual clustering algorithm for uncertain categorical data. Aggarwal [2] proposes density based transforms for uncertain data mining. There is also some research on identifying frequent itemsets and association mining [7,23] from uncertain datasets. The support of itemsets and confidence of association rules are integrated with the existential probability of transactions and items. Burdicks [5] discuss OLAP computation on uncertain data. None of them address the issue of developing a general classification and prediction algorithm for uncertain data.

3 Data Uncertainty

In this section, we will discuss the uncertainty model for both numerical and categorical attributes. Here we focus on the attributes uncertainty and assume the class type is certain.

When the value of a numerical attribute is uncertain, the attribute is called an uncertain numerical attribute (UNA), denoted by $A_i^{u_n}$. Further, we use $A_{ij}^{u_n}$ to denote the jth instance of $A_i^{u_n}$. The concept of UNA has been introduced in [6]. The value of $A_i^{u_n}$ is represented as a range or interval and the probability distribution function (PDF) over this range. Note that $A_i^{u_n}$ is treated as a continuous random variable. The PDF $f(x)$ can be related to an attribute if all instances have the same distribution, or related to each instance if each instance has a different distribution.

An uncertain interval instance of $A_i^{u_n}$, denoted by $A_{ij}^{u_n}.U$, is an interval $[A_{ij}^{u_n}.l, A_{ij}^{u_n}.r]$ where $A_{ij}^{u_n}.l, A_{ij}^{u_n}.r \in R$, $A_{ij}^{u_n}.r \geq A_{ij}^{u_n}.l$. The uncertain PDF of $A_{ij}^{u_n}$, denoted by $A_{ij}^{u_n}.f(x)$, is a probability distribution function of $A_{ij}^{u_n}$, such that $\int_{A_{ij}^{u_n}.l}^{A_{ij}^{u_n}.r} A_{ij}^{u_n}.f(x)dx = 1$ and $\int_{A_{ij}^{u_n}.l}^{A_{ij}^{u_n}.r} A_{ij}^{u_n}.f(x)dx = 0$ if $x \notin A_{ij}^{u_n}.U$.

A dataset can also have categorical attributes that are allowed to take on uncertain values. We call such attributes uncertain categorical attributes(UCA), denoted by $A_i^{u_c}$. Further, we use $A_{ij}^{u_c}$ to denote the attribute value of the jth instance of $A_i^{u_c}$. The notion of UCA was proposed in [19].

$A_{ij}^{u_c}$ takes values from the categorical domain Dom with cardinality $|Dom| = n$. For a certain dataset, the value of an attribute A is a single value d_k in Dom, $Pr(A = d_k) = 1$. In the case of an uncertain dataset, we record the information by a probability distribution over Dom instead of a single value. Given a categorical domain $Dom = \{d_1, \ldots, d_n\}$, an uncertain categorical attribute (UCA)

A^{u_c} is characterized by probability distribution over Dom. It can be represented by the probability vector $P = \{p_1, \ldots, p_n\}$ such that $P(A_{ij}^{u_c} = v_k) = p_{jk}$ and $\sum_{k=1}^{n} p_{jk} = 1 (1 \leq k \leq n)$.

4 Attribute Test Condition: Identifying the Best Split

The key issue of a decision tree induction algorithm is to decide the way records be split. Each step of the tree-grow process needs to select an attribute test condition to divide the records into smaller subsets. Widely used splitting measures such as information entropy and the Gini index are not applicable to uncertain data. In this section, we will define splitting measures for both uncertain numerical data and uncertain categorical data.

4.1 Uncertain Numerical Attributes

As described earlier, the value of an uncertain numerical attribute is an interval with associated PDF. Table 1 shows an example of UNA. The data in this table are used to predict whether borrowers will default on loan payments. Among all the attributes, the Annual Income is an UNA, whose precise value is not available. We only know the range of the Annual Income of each person and the PDF $f(x)$ over that range. The probability distribution function of the UNA attribute Annual Income is assumed to be uniform distribution.

Each uncertain numerical value has a maximal value and a minimal value, which we call critical points. For each UNA, we can order all critical points of an uncertain numerical attribute in an ascending sort with duplicate elimination. Then the UNA can be partitioned. One partition may overlap with the UNA of many instances. When an instance with UNA overlaps with a partition [a, b),

Table 1. Uncertain Numerical Data

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	110-120	No
2	No	Married	100-120	No
3	No	Single	60-85	No
4	Yes	Married	110-145	No
5	No	Divorced	110-120	Yes
6	No	Married	50-80	No
7	Yes	Divorced	170-250	No
8	No	Single	85-100	Yes
9	No	Married	80-100	No
10	No	Single	120-145	Yes
11	No	Divorced	105-125	Yes
12	No	Divorced	80-95	No

the probability its UNA actually falls in that partition is $\int_a^b f(x)dx$. Based on the probability of each individual instance falling in a partition [a, b), we can compute the probabilistic number of instances falling in that partition, which we call probabilistic cardinality.

The probabilistic cardinality of the dataset over a partition Pa = [a, b) is the sum of the probabilities of each instance whose corresponding UNA falls in [a, b). That is, $PC(Pa) = \sum_{j=1}^{n} P(A_{ij}^{u_n} \in [a,b)) = \sum_{j=1}^{n} \int_a^b A_{ij}^{u_n}.f(x)dx$. The probabilistic cardinality for class C_j of the dataset over a partition Pa=[a, b) is the sum of the probability of each instance T_j in C_j whose corresponding UNA falls in [a, b). That is, $PC(Pa, C) = \sum_{j=1}^{n} P(A_{ij}^{u_n} \in [a,b) \wedge C_{T_j} = C_j)$, where $C_{T_j} = C_j)$ denotes the class label of instance T_j.

Refer to the dataset in Table 1, the probabilistic cardinality for the partition [110, 120) on the Annual Income is the sum of the probabilities of instances that have Annual Income falling in [110, 120). Suppose the annual income for each instance is uniformly distributed over its uncertain interval; instances 1, 2, 4, 5 and 11 have overlap with [110,120), and the probability for instance 1 with annual income in [110, 120) is $P(I1 \in [110, 120)) = (120-110)/(120-110) = 1$. Similarly, $P(I2 \in [110, 120)) = 0.5$, $P(I4 \in [110, 120)) = 0.29$, $P(I5 \in [110, 120)) = 1$, and $P(I11 \in [110, 120)) = 0.5$; therefore, the probabilistic cardinality for this dataset over partition [110, 120) is 3.29. The probabilistic cardinality for class Default-Borrower = NO over the partition [110, 120) on the Annual Income is the sum of the probabilities of instances who are not DefaultBorrowers with Annual Income falling in [110, 120). Among instances 1, 2, 4, 5 and 11 who have overlap with [110,120), only instances 1, 2 and 4 are in class NO; therefore, the probabilistic cardinality for DefaultBorrower = NO over partition [110, 120) is 1.79. Similarly, the probabilistic cardinality for DefaultBorrower = Yes over partition [110, 120) is 1.5.

With the two previous definitions, we can now define the probabilistic entropy for uncertain data as follows:

Definition 1. *The* Probabilistic Entropy *for a dataset D is* $ProbInfo(D) = -\sum_{i=1}^{m} \frac{PC(D,i)}{PC(D)} \times log_2(\frac{PC(D,i)}{PC(D)})$.

Suppose attribute A is selected as the split attributecan, and it partitions the dataset D into k subsets, $\{D_1, D_2, \ldots, D_k\}$. Then the probabilistic entropy, or expected information based on the partitioning is given by $ProbInfo_A(D) = \sum_{j=1}^{k} \frac{PC(D_j)}{PC(D)} \times ProbInfo(D_j)$. The term $PC(D_j)$ acts as the weight of the jth partition. The smaller the entropy value, the greater the purity of the subset partitions. The encoding information that would be gained by branching on A is $ProbGain(A) = ProbInfo(D) - ProbInfo_A(D)$.

Probabilistic Entropy also tends to favor attributes that have a large number of distinct values. The information gained by a test is maximal when there is one case in each subset D_j. To overcome this problem, the splitting criterion should be modified to take into account the number of outcomes produced by the attribute test condition. This criterion is defined as $ProbGain_ratio(A) = \frac{ProbGain(A)}{ProbSplitInfo_A(D)}$. Here, $ProbSplitInfo_A(D) = -\sum_{j=1}^{k} \frac{PC(D_j)}{PC(D)} \times log_2(\frac{PC(D_j)}{PC(D)})$

and k is the total number of splits. If an attribute produces a large number of splits, its split information will also be large, which in turn reduces its gain ratio.

4.2 Uncertain Categorical Data

An uncertain discrete attribute (UCA) $A_i^{u_c}$ is characterized by probability distribution over Dom. As mentioned earlier, it can be represented by the probability vector $\{p_1, \ldots, p_n\}$ such that $P(A_{ij}^{u_c} = d_j) = p_j (1 \leq j \leq n)$.

Table 2. Uncertain Categorical Data

ID	Make	Date	Problem	Location	Class
1	Explorer	4/5/08	(Brake:0.5; Tire:0.5)	CA	0
2	Camry	8/3/02	(Trans:0.2; Tire:0.8)	IN	1
3	Civic	9/12/99	(Exhaust:0.4; Brake:0.6)	TX	0
4	Pontiac	4/2/01	(Tire:1.0)	IL	1
5	Caravan	1/23/04	(Trans:0.3; Brake:0.7)	NY	1

Table 2 shows an example of UCA [19]. This dataset records vehicle problem information. The problem can be caused by the brake, tire, transmission or other parts. It is derived from the text field in the given tuple using a text classifier/miner. As text miner result tend to be uncertain, the Problem field is a UCA.

Similar to uncertain numerical data, the probabilistic cardinality of the dataset over d_j is the sum of the probabilities of each instance whose corresponding UCA equals to d_j. That is, $PC(d_j) = \sum_{j=1}^{n} P(A_{ij}^{u_c} = d_j)$. The probabilistic cardinality for class C of the dataset over d_j is the sum of the probabilities of each instance in C_j whose corresponding UCA equals to d_j. That is, $PC(d_j, C) = \sum_{j=1} P(A_{ij}^{u_c} = d_j \wedge C_j = C)$.

Refer to the dataset in Table 2, the probabilistic cardinality over Problem = Brake is the sum of the probabilities of each instance whose Problem attribute is Brake, which is 1.8. The probabilistic cardinality for class 0 over "Problem = Brake" is the overall probabilities of instances in class 0 whose Problem attribute is Brake, which is 1.1. Based on the probabilistic cardinality for each class C, we can then compute the probabilistic information entropy and probabilistic information gain ratio if the data is split on the categorical attribute "Problem", following the same process as for uncertain numerical data. If it has the highest probabilistic information gain, then "Problem" will be chosen as the next splitting attribute.

5 Algorithms for DTU

5.1 Decision Tree Induction Algorithm

The algorithm is shown in Algorithm 1. The basic strategy is as follows:

Algorithm 1. DTU Induction

input: the training dataset D; the set of candidate attributes att-list
output: An uncertain decision tree

begin

1: create a node N;
2: **if** (D are all of the same class, C) **then**
3: return N as a leaf node labeled with the class C;
4: **else if** (attribute-list is empty) **then**
5: return N as a leaf node labeled with the highest weight class in D;
6: **end if**;
7: select a test-attribute with the highest probabilistic information gain ratio to label node N;
8: **if** (test-attribute is numeric or uncertain numeric) **then**
9: binary split the data from the selected position y;
10: **for** (each instance R_j) **do**
11: **if** (test-attribute $\leq y$) **then**
12: put it into D_l with weight $R_j.w$;
13: **else if** (test-attribute $> y$) **then**
14: put it into D_r with weight $R_j.w$;
15: **else**
16: put it into D_l with weight $R_j.w * \int_{x_1}^{y} f(x)dx$;
17: put it into D_r with weight $R_j.w * \int_{y}^{x_2} f(x)dx$;
18: **end if**;
19: **end for**;
20: **else**
21: **for** (each value $a_i (i = 1, \ldots, n)$ of the attribute) **do**
22: grow a branch D_i for it;
23: **end for**;
24: **for** (each instance R_j) **do**
25: **if** (test-attribute is uncertain) **then**
26: put it into D_i with $R_j.a_i.w * R_j.w$ weight;
27: **else**
28: put it into a certain D_i with weight $R_j.w$;
29: **end if**
30: **end for**;
31: **end if**;
32: **for** each D_i **do**
33: attach the node returned by DTU(D_i, att-list);
34: **end for**;

end

1. The tree starts as a single node representing the training samples (step 1).

2. If the samples are all of the same class; then the node becomes a leaf and is labeled with that class (steps 2 and 3).

3. Otherwise, the algorithm uses a probabilistic entropy-based measure, known as the probabilistic information gain ratio, as the criteria for selecting the attribute that will best separate the samples into an individual class (step 7). This attribute becomes the "test" attribute at the node.

4. If the test attribute is numerical or uncertain numerical, we split for the data at the selected position y (steps 8 and 9).

5. A branch is created for test-attribute $\leq y$ or test-attribute $> y$ respectively. If an instance's test attribute value $[x_1, x_2]$ is less than or equal to y ($x_2 \leq y$), it is put into the left branch with the instance's weight $R_j.w$. If an instance's test attribute value $[x_1, x_2]$ is larger than y ($x_1 > y$), it is put into the right branch with the instance's weight $R_j.w$. If an attribute's value $[x_1, x_2]$ covers the split point y ($x_1 \leq y < x_2$), it is put into the left branch with weight $R_j.w * \int_{x_1}^{y} f(x)dx$ and the right branch with weight $R_j.w * \int_{y}^{x_2} f(x)dx$. Then the dataset is divided into D_l and D_r (steps 10-19).

6. If the test attribute is categorical or uncertain categorical, we split the data multiway (steps 21-30). A branch is created for each value of the test attribute, and the samples are partitioned accordingly. For each value a_i of the attribute, an instance is put into D_i with $R_j.w$ weight when the attribute is certain. If the attribute is uncertain, assume the probability of the attribute value a_i be $R_j.a_i.p$, then the instance is put into the branch a_i with the weight $R_j.a_i.p * R_j.w$.

7. The algorithm recursively applies the same process to generate a decision tree for the samples.

8. The recursive partitioning process stops only when either of the following conditions becomes true:

1) All samples for a given node belong to the same class (steps 2 and 3), or

2) There are no remaining attributes on which the samples may be further partitioned (step 4). In this case, the highest weight class is employed (step 5). This involves converting the given node into a leaf and labeling it with the class having the highest weight among samples. Alternatively, the class distribution of the node samples may be stored.

5.2 Prediction with DTU

Once a DTU is constructed, it can be used for predicting class types. The prediction process starts from the root node, the test condition is applied at each node in DTU, and the appropriate branch is followed based on the outcome of the test. When the test instance R is certain, the process is quite straightforward since the test result will lead to one single branch without ambiguity. When the test is on an uncertain attribute, the prediction algorithm proceeds as follows:

1. If the test condition is on a UNA attribute A and the splitting point is a, suppose $R.A$ is an interval $[x_1, x_2)$ with associated pdf $R.A.f(x)$:

If $a < x_1$, which means the minimal possible value of $R.A$ is larger than a, then $P(R.A > a) = R.w$; we know for sure $R.A > a$ and R follows the right branch;

If $a >= x_2$, which means the maximal possible value of $R.A$ is smaller than a, then $P(R.A < a) = R.w$, and it is certain that $R.A < a$ and R follows the left branch;

If $(x_1 < a < x_2)$, then the probability $R.A < a$ is $P(R.A < a) = R.w * \int_{x_1}^{a} f(x)dx$ and the probability $R.A > a$ is $P(R.A > a) = R.w * \int_{a}^{x_2} f(x)dx$. R

should be in the left branch with probability $R.w * \int_{x_1}^{a} f(x)dx$ and in the right branch with probability $R.w * \int_{a}^{x_2} f(x)dx$.

2. If the test condition is on a UCA attribute A and $a_1, a_2, ...a_k$ are the values for the categorical attribute A, then suppose $R.A$ is an UCA, that is $R.A = \{p_1, p_2, ..., p_k\}$, with $p_i (i = 1, ..., k)$ as the probability of $R.A = a_i$. Then R should be in the ith branch with probability p_i.

For the leaf node of DTU, each class C_i has a probability $PL(C_i)$, which is the probability for an instance to be in class C_i if it falls in this leaf node. $PL(C_i)$ is computed as the fraction of the probabilistic cardinality of instances in class C_i in a leaf node over the total probabilistic cardinality of instances in that node. Assume path L from the root to a leaf node contains t tests, and the data are classified into one class c_i in the end, suppose $P(T_i)$ is the probability that an instance follow the path at the ith test, then the probability for an instance to be in class c_i taking that particular path L is $P_{c_i}^{L} = PL(c_i) * \prod_{i=1}^{t} P(T_i)$.

When predicting the class type for an instance T with uncertain attributes, it is possible that the process takes multiple paths. Suppose there are m paths taken in total, then the probability for T in class c_i is $P_{c_i} = \sum_{i=1}^{m} P_{c_i}^{i}$.

Finally, the instance will be predicted to be of class c_i which has the largest P_{c_i} among all $P_{c_i}, i = 1, ..., n$.

6 Experiments

In this section, we present the experimental results of the proposed decision tree algorithm DTU. We studied the prediction accuracy over multiple datasets.

Based on the J4.8/C4.5 implemented on Weka [21], we implemented the DTU as described in Section 5. The experiments are executed on a PC with an Intel Pentium IV 3.4GHz CPU and 2.0 GB main memory. A collection containing 10 real-world benchmark datasets were assembled from the UCI Repository [1]. We tried to cover the spectrum of properties such as size, attribute numbers and types, number of classes and class distributions. Among these 10 datasets, 5 of them, namely Iris, Sonar, Segment, Diabetes and Glass contain mainly numerical attributes. The remaining 5 datasets, namely Audiology, Bridges, Promoters, Mushroom and voting have mostly categorical attributes.

Due to a lack of real uncertain datasets, we introduce synthetic uncertainty into the datasets. To make numerical attributes uncertain, we convert each numerical value to an uncertain interval with uniform probability distribution function. The uncertain interval is randomly generated around the original value. These are uncertainties from random effects without any bias. If the uncertain interval is within 10% of the original data, we call the dataset with uncertainty 10% and denote it by U10. For example, when the original value is 20, then its U10 may be [18.4, 20.4). We make categorical attributes uncertain by converting them into probability vectors. For example, a categorical attribute A_i may have k possible values $v_j, 1 \leq j \leq k$. For an instance I_j, we convert its value A_{ij} into a probability vector $\mathbf{P} = (p_{j1}, p_{j2}, ..., p_{ji}, ..., p_{jk})$, while p_{jl} is the probability of $A_{ij}^{u_c}$ to be equal to v_l, that is, $P(A_{ij}^{u_c} = v_l) = p_{jl}$. For example, when we

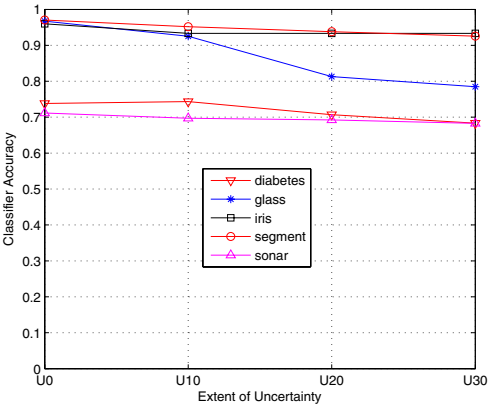


Fig. 1. DTU accuracy on uncertain numerical data sets

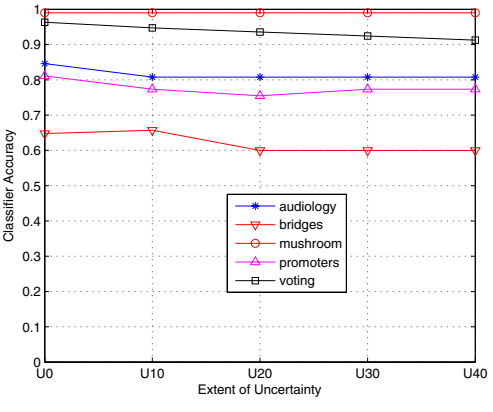


Fig. 2. DTU accuracy on uncertain categorical data sets

introduce 10% uncertainty, this attribute will take the original value with 90% probability, and 10% probability to take any of the other values. Suppose in the original accurate dataset $A_{ij} = v_1$, then we will assign $p_{j1} = 90\%$, and assign $p_{jl}(2 \leq l \leq k)$ to ensure $\sum_{i=2}^{k} p_{jl} = 10\%$. Similarly, we denote this dataset with 10% uncertainty in categorical data by U10. We use U0 to denote accurate or certain datasets.

As prediction accuracy is by far the most important measure of a classifier, we studied the prediction accuracy of DTU classifier first. Figure 1 shows the result for numerical datasets and Figure 2 shows the result for categorical datasets. In both experiments, we use ten-fold cross validation. Data is split into 10 approximately equal partitions; each one is used in turn for testing while the remainder is used for training, that is, 9/10 of data is used for training and 1/10 for test. The

whole procedure is repeated 10 times, and the overall accuracy rate is counted as the average of accuracy rates on each partition. When DTU is applied on certain data, it works as a traditional C4.5 classifier.

For numerical data, the uncertainty varies between 0 to 30%. As shown in Figure 1, when the extent of uncertainty increases, the classifier accuracy declines slowly. For most datasets, the performance decrement are within 5%, even when data uncertainty reaches 30%. The worst performance decrement is for the glass identification dataset, the classifier has over 95% accuracy on certain data, reduces to around 92% when the uncertainty is 10%, to 81% when the uncertainty is 20%, and to 78% when the uncertainty reaches 30% .

The results for categorical datasets are similar, as shown in Figure 2. Overall speaking, the accuracy of DTU classifier remains relatively stable. The overall decrease in classifier accuracy is within 10% even when the uncertainty reaches 40%. Both experiments show DTU is quite robust against data uncertainty.

7 Conclusions

In this paper, we propose a new decision tree algorithm DTU for classifying and predicting uncertain data. We extend the measures used in tradition decision tree, such as information entropy and information gain, for handling data uncertainty. Our experiments demonstrate that DTU can process both uncertain numerical data and uncertain categorical data. It can achieve satisfactory classfication and prediction accuracy even when data is highly uncertain.

References

1. `http://archive.ics.uci.edu/ml/datasets.html`
2. Aggarwal, C.: On density based transforms for uncertain data mining. In: ICDE, pp. 866–875 (2007)
3. Andrews, R., Diederich, J., Tickle, A.: A survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge Based Systems 8(6), 373–389 (1995)
4. Bi, J., Zhang, T.: Support Vector Classification with Input Data Uncertainty. Advances in Neural Information Processing Systems 17, 161–168 (2004)
5. Burdick, D., Deshpande, M.P., Jayram, T.S., Ramakrishnan, R., Vaithyanathan, S.: OLAP over uncertain and imprecise data. The VLDB Journal 16(1), 123–144 (2007)
6. Cheng, R., Kalashnikov, D., Prabhakar, S.: Evaluating probabilistic queries over imprecise data. In: Proceedings of the ACM SIGMOD, pp. 551–562 (2003)
7. Chui, C., Kao, B., Hung, E.: Mining Frequent Itemsets from Uncertain Data. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS, vol. 4426, pp. 47–58. Springer, Heidelberg (2007)
8. Cormode, G., McGregor, A.: Approximation algorithms for clustering uncertain data. In: PODS 2008, pp. 191–199 (2008)
9. Dietterich, T.G.: Ensemble Methods in Machine Learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000)

10. Gonzalez, E.V., Broitman, I.A.E., Vallejo, E.E., Taylor, C.E.: Targeting Input Data for Acoustic Bird Species Recognition Using Data Mining and HMMs. In: Proceedings of the ICDMW 2007, pp. 513–518 (2007)
11. Hawarah, L., Simonet, A., Simonet, M.: Dealing with Missing Values in a Probabilistic Decision Tree during Classification. In: The Second International Workshop on Mining Complex Data, pp. 325–329 (2006)
12. Jebari, C., Ounelli, H.: Genre categorization of web pages, In: Proceedings of the ICDMW 2007, pp. 455–464 (2007)
13. Kriegel, H., Pfeifle, M.: Density-Based Clustering of Uncertain Data. In: Proceedings of the KDD 2005, pp. 672–677 (2005)
14. Langley, P., Iba, W., Thompson, K.: An analysis of Bayesian classifiers. In: Proceedings of the tenth National Conference on artigicial intelligence, pp. 223–228 (1992)
15. Lobo, O., Numao, M.: Ordered estimation of missing values. In: Zhong, N., Zhou, L. (eds.) PAKDD 1999. LNCS, vol. 1574, pp. 499–503. Springer, Heidelberg (1999)
16. Ngai, W.K., Kao, B., Chui, C.K., Cheng, R., Chau, M., Yip, K.Y.: Efficient Clustering of Uncertain Data. In: Proceedings of ICDM 2006, pp. 436–445 (2006)
17. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufman Publishers, San Francisco (1993)
18. Quinlan, J.R.: Probabilistic decision trees. Machine Learning: an Artificial Intelligence Approach 3, 140–152 (1990)
19. Singh, S., Mayfield, C., Prabhakar, S., Shah, R., Hambrusch, S.: Indexing Categorical data with uncertainty. In: Proceedings of ICDE 2007, pp. 616–625 (2007)
20. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, Heidelberg (1995)
21. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufman Publishers, San Francisco (2005)
22. Xia, Y., Xi, B.: Conceptual clustering categorical data with uncertainty. In: Proceedings of international conference on tools with artificial intelligence, pp. 329–336 (2007)
23. Yu, Z., Wong, H.: Mining Uncertain Data in Low-dimensional Subspace. In: Proceedings of ICPR 2006, pp. 748–751 (2006)
24. Qin, B., Xia, Y., Prbahakar, S., Tu, Y.: A Rule-based Classification Algorithm for Uncertain Data. In: The Workshop on Management and Mining Of Uncertain Data (MOUND) (2009)

Efficient Privacy-Preserving Link Discovery

Xiaoyun He[1], Jaideep Vaidya[1], Basit Shafiq[1], Nabil Adam[1],
Evimaria Terzi[2], and Tyrone Grandison[2]

[1]CIMIC, Rutgers University, USA
{xiaoyun,jsvaidya,basit,Adam}@cimic.rutgers.edu
[2]IBM Almaden Research Center, USA
{eterzi,tyroneg}@us.ibm.com

Abstract. Link discovery is a process of identifying association(s) among different entities included in a complex network structure. These association(s) may represent any interaction among entities, for example between people or even bank accounts. The need for link discovery arises in many applications including law enforcement, counter-terrorism, social network analysis, intrusion detection, and fraud detection. Given the sensitive nature of information that can be revealed from link discovery, privacy is a major concern from the perspective of both individuals and organizations. For example, in the context of financial fraud detection, linking transactions may reveal sensitive information about other individuals not involved in any fraud. It is known that link discovery can be done in a privacy-preserving manner by securely finding the transitive closure of a graph. We propose two very efficient techniques to find the transitive closure securely. The two protocols have varying levels of security and performance. We analyze the performance and usability of the proposed approach in terms of both analytical and experimental results.

Keywords: Privacy, Link Discovery, Efficiency.

1 Introduction

Link discovery is a process of identifying association(s) among different entities included in a complex network structure [1,2]. These association(s) may represent any interaction among entities, for example, between people or even bank accounts. The need for link discovery arises in many applications including law enforcement, counter-terrorism, social network analysis, intrusion detection, and fraud detection.

Link discovery in these application domains often involves analysis of huge volumes of data distributed across different sources with different rules and regulations on data sharing. For instance, law enforcement investigations often involve finding links between individuals or discovering association of individuals with specific organizations or groups [2]. To discover such links, information needs to be sifted through various sources such as law enforcement databases, financial transactions, and phone records, etc. The information stored in such data repositories is often confidential. Given the sensitive nature of information that can be revealed from link discovery, privacy is a major concern for both individuals and organizations [3].

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 16–27, 2009.
© Springer-Verlag Berlin Heidelberg 2009

In the past years, there has been increasing interest in developing techniques for link discovery and analysis in network or graph structured data [1,4,5,6]. However, none of these works have considered privacy issues in a distributed context while discovering links among entities. Recent work by Duan et al. [7] first presents a generalized HITS algorithm to rank the linked entities on weighted graph, they solve an orthogonal problem to what we present in this paper. There has also been increasing interest in privacy-preserving data mining [8,9,10], some of which work is of interest.

Recently, He et al.[2] have proposed an approach for privacy-preserving link discovery in a complex and distributed network structure. Specifically, the entities in the network structure are viewed as nodes in a graph with an edge between two nodes representing the association between the corresponding entities. Different portions of the graph (subgraphs) correspond to data repositories owned by different parties. For example, in the context of financial transactions, the graph nodes represent customer accounts and the links represent the transaction among accounts, which may belong to the same bank or different banks. Thus, the entire graph represents the global view including all information repositories. The problem of privacy-preserving link discovery can then be reduced to finding the transitive closure of a distributed graph in a secure manner. He et al.[2] further show that this can be done via a split matrix multiplication protocol, which itself can be implemented using the completely secure scalar product protocol of Goethals et al.[11]. However, due to the required large number of costly encryption / decryption operations, the approach is computationally quite prohibitive.

In this paper, we propose two different methods to improve the computational efficiency of secure link discovery. The first method is based on commutative encryption. This approach leaks a little more information (each party gets to know the final transitive closure matrix involving their own vertices as well a range on when these connections are formed). However, it is significantly more efficient and practical. In the second method, the secure transitive closure is computed over a much smaller graph composed of representative nodes heuristically selected. Instead of including all of its nodes for a complete secure transitive closure computation, each party chooses a fraction of its overall nodes as its representatives set to form its representative matrix, and the secure transitive closure computed over it. The global transitive closure is then approximated based on the representative transitive closure. Our experiments show the effectiveness of our proposed approach.

2 Preliminaries and Problem Definition

In this paper, we consider a distributed environment with k parties $P_1, \ldots, P_k$. The overall distributed network data is modelled as a simple directed graph $G(V, E)$, where V is a set of nodes with $|V| = n$, and $E \subseteq V \times V$ is a set of directed edges. Each party P_i owns part of graph G, denoted by $G(V_i, E_i)$, where $\cup_i V_i = V$, $|V_i| = m_i$ with $\sum_{i=1}^{k} m_i = n$, $V_i \cap V_j = \emptyset (i \neq j)$, and $\cup_i E_i \subseteq E$. Note that some edges in E may cross the boundaries of G_i and $G_j (i \neq j)$ with one node in G_i and another node in G_j. These edges are called inter-edges.

Given two nodes u, v in V we define the predicate DIRPATH(U,V) as evaluating to 1 if there exists a directed path between u and v, and otherwise evaluating to 0. Given any

two distinct nodes $u, v \in V$, we are interested in being able to check whether there is a path from u to v in the global graph G no matter the subgraph in which the two nodes u and v reside. That is, we are interested in evaluating predicate DIRPATH(u, v) for every pair of nodes $u, v \in V$ in a *privacy-preserving* manner. In other words, we require that the process of evaluating predicate DIRPATH should not reveal any additional information to any party after the computation. Evaluating DIRPATH in this setting leads to the following definition of the PRIVACY-PRESERVING LINK DISCOVERY.

Problem 1. Consider directed graph $G(V, E)$ split among k parties $P_1, \ldots, P_k$ as described above. For every pair of nodes $u, v \in V$ evaluate the value of predicate DIRPATH (u, v) in G in a privacy-preserving manner.

We also require the following definitions:

Definition 1. *Given party P_i that keeps graph $G_i(V_i, E_i)$ we define the* set of inter-nodes $V_I^{(i)}$ *to be the union set of nodes that either start or end inter-edges.*

In addition, we denote V_I as the set union of all $V_\mathrm{I}^{(i)} (i = 1, 2, \ldots, k)$.

Definition 2. *Given a node $u \in V_i$, its inter-degree $D_I(u)$ is defined as the total number of edges which are either $(u, v) \in E$ or $(v, u) \in E$, and $v \in V_I^{(j)} (i \neq j)$.*

Definition 3. *Given a node $u \in V_i$, its local-degree $D_L(u)$ is defined as the total number of edges which are either $(u, v) \in E_i$ or $(v, u) \in E_i$.*

Definition 4. *Given a node $u \in V_i$, its combined-degree $D_C(u)$ is defined as the sum of $D_I(u)$ and $D_L(u)$.*

In addition, given a graph $G = (V, E)$, its final full transitive closure is denoted by TC. In our second proposed approach, we derive an approximate transitive closure of G, denoted by TC'. To assess the effectiveness of our proposed heuristic approach, we also define two measures as follows.

Definition 5. *The total accuracy is defined as the total number of matched elements between TC and TC' divided by n^2 (that is, $\frac{total\ number\ of\ matches}{n^2}$), where n is the size of the adjacency matrix.*

Definition 6. *The edge accuracy is defined as the total number of matched non-zero elements between TC and TC' divided by the total number of non-zero elements (that is, $\frac{total\ number\ of\ matched\ non-zeros}{total\ number\ of\ non-zeros}$).*

Note that, in the above definitions, matched elements mean that, given a specified row and column, the corresponding entries in the matrices of TC and TC' have the same value. The reason that we would like to include both total accuracy and edge accuracy as performance metrics is that, in the case of sparse graphs, total accuracy can be very high while having low edge accuracy. In general, a high edge accuracy is an indicator of good performance. In the following, whenever accuracy is mentioned, it means edge accuracy unless otherwise stated.

3 Overview of Secure Transitive Closure

In this section, we briefly provide an overview of secure transitive closure introduced in [2]. For more details, we refer the readers to [2]. The transitive closure [12] of a graph $G = [V, E]$ is a graph $G^* = [V, E^*]$ with edge $(i, j) \in E^*$ if and only if there is a path from vertex i to vertex j in the graph G. A simple matrix multiplication method can be used to compute the transitive closure of a graph. If A represents the adjacency matrix of graph G, then A^n represents the transitive closure G^*, where n is the number of vertices in G.

To address the privacy concerns, the approach proposed in [2] enables secure computation of the transitive closure of a distributed graph without requiring parties to reveal any details about their subgraphs. Specifically, the protocol for secure transitive closure computation is run by all k parties that own a portion of the distributed graph. Let $A^{(i)}$ denote the adjacency matrix corresponding to the subgraph $G_i = (V_i, E_i)$ owned by party P_i. $A^{(i)}$ is a $n \times n$ matrix, where the matrix entry $A^{(i)}[p, q] = 1$ if the edge $(p, q) \in E_i$. All other entries in the matrix $A^{(i)}$ are set to zero. Therefore, the overall adjacency matrix A of the distributed graph is given by: $A = \sum_{i=1}^{k} A^{(i)}$.

The transitive closure A^n of the distributed graph is computed iteratively through matrix multiplication, with the output of the last iteration used in this iteration . For instance in the rth iteration, $(r \leq n)$, the matrix A^{2r} is computed as follows:

$$A^{2r} = A^r A^r = \sum_{i=1}^{k} O^{(i)} \sum_{j=1}^{k} O^{(j)} \tag{1}$$

The Split Matrix Multiplication. Equation 1 used for computation of the transitive closure involves pair wise multiplication of the output split matrices of each party. It is obvious that each party P_i can locally compute $O^{(i)}O^{(i)}$. Therefore, the secure computation of A^{2r} comes down to securely computing $O^{(i)}O^{(j)}(\forall i \neq j)$. Since the matrix multiplication essentially is the scalar product operations, $O^{(i)}O^{(j)}$ can be computed by invoking the secure scalar product protocol proposed in [11].

It is important to note that actual adjacency matrix of the distributed graph in each iteration is never known completely to any party. Rather, as the output of each iteration, each party P_i gets a matrix $O^{(i)}$ consisting of random shares of the global adjacency matrix. Thus, for any given row p and column q $(1 \leq p, q \leq n)$, and iteration r, $\sum_{i=1}^{k} O^{(i)}(p, q) = A^r(p, q)$. In the end, the values of the final matrix A^n are split randomly and returned to each party as matrices $O^{(1)}, O^{(2)}, \ldots, O^{(k)}$.

However,the above approach in [2] is computationally prohibitive. With $log_2 n$ iterations, the total number of encryptions and decryptions required performed is $(k^2 - k) \cdot n^2 \cdot log_2 n$, while the total number of exponentiations and multiplications performed is $(k^2 - k) \cdot n^3 \cdot log_2 n$. Overall the encryption/decryption time dominates. Although the split matrix multiplication approach requires $O(n^2)$ encryption/decryptions, the computational time for large distributed graphs will be significantly high due to the high computational cost of encryption/decryption.

Algorithm 1. Efficient Secure Transitive Closure

Require: k parties, $P_1, \ldots, P_k$, Party P_i has m_i vertices
Require: Let $n = \sum_{i=1}^{k} m_i$, represent the total number of vertices
Require: Let the matrix $A^{(i)}(n \times n)$ represent the local adjacency matrix of party P_i (i.e., the matrix entry $A^{(i)}[p, q] = 1$ if the edge $(p, q) \in E_i$, otherwise 0.)

```
for j ← 1 ... ⌈log_2 n⌉ do
    for p ← 1 ... n do
        for q ← 1 ... n do
            {Assume P_i owns vertex p}
            At P_i: Initiate boolean scalar product protocol described in Algorithm 2 to get output value x
            if x = 0 then
                At P_i: A^(i)[p, q] ← 0
            else
                At P_i: A^(i)[p, q] ← 1
            end if
        end for
    end for
end for
```

4 A Commutative Encryption Based Approach

In this section, we provide an alternative approach that uses a much more efficient protocol for the scalar product requiring relatively fewer encryption/decryption operations. The protocol is depicted in Algorithm 1. This approach also uses split matrix multiplication for secure computation of the transitive closure matrix. However, it employs commutative encryption for computation of the scalar product and works only if boolean values are used. In other words, the split matrices generated in each iteration of the split matrix multiplication needs to be converted into boolean values. This will result in leakage of additional information to the different parties. In particular, each party will know the portion of the final transitive closure matrix involving the party's own vertices. Additionally, each party will know in which iteration a zero value in its local output matrix changes to a non-zero value. As a result, the party will know the range on the number of links to which its local vertices are connected to external vertices. For example, if the matrix entry $O^{(i)}[p, q]$ changes its value from zero to non-zero in the 3rd iteration, then the shortest path between vertex p and q consists of at least 4 and at most 8 links. One way to reduce this leakage is to use a hybrid approach, where the approach proposed in [2] is used for the first few iterations before switching to the new approach.

4.1 Commutative Encryption Based Scalar Product

With boolean vectors, it is possible to get a more efficient scalar product. To see this, note that if we encode the vectors as sets (with position numbers as elements), the scalar product is the same as the size of the intersection set. For example, assume we have vector $\boldsymbol{X} = (1, 0, 0, 1, 1)$ and $\boldsymbol{Y} = (0, 1, 0, 1, 0)$. Then the scalar product

$\boldsymbol{X} \cdot \boldsymbol{Y} = \sum_{i=1}^{5} x_i * y_i$. Now, the corresponding set encodings are $XS = (1, 4, 5)$ and $YS = (2, 4)$. Once can see that the size of the intersection set $|XS \bigcap YS| = 1$ is exactly the same as the scalar product. This idea is used to compute the scalar product.

The basic idea is to use commutative encryption to encrypt all of the items in each party's set. Commutative encryption is an important tool used in many cryptographic protocols. An encryption algorithm is commutative if the order of encryption does not matter. Thus, for any two encryption keys $E1$ and $E2$, and any message m, $E1(E2(m)) = E2(E1(m))$. The same property applies to decryption as well – thus to decrypt a message encrypted by two keys, it is sufficient to decrypt it one key at a time. The basic idea is for each source to encrypt its data set with its keys and pass the encrypted data set to the next source. This source again encrypts the received data using its encryption keys and passes the encrypted data to the next source until all sources have encrypted the data. Since we are using commutative encryption, the encrypted values of the set items across different data sets will be equal if and only if their original values are equal. Thus, all the intersection of the encrypted values gives the logical AND of the vectors, and counting the size of the intersection set gives the total number of 1s (i.e., the scalar product). The encryption prevents any party from knowing the actual value of any local item. This scalar product method only works for boolean vectors, but it will still work in this context, since after each iteration the non-zero values in the local adjacency matrix are set to one by the party owning the corresponding data point.

In our case, for the scalar product, the first vector is owned completely by one party while the second vector is split between all of the parties. One can simply compute all of the local scalar products to add up the sum to get the global scalar product. However, this creates a serious security problem. To see this, assume that a party P_i owning m_i vertices, gets local scalar products from another party P_j owning m_j vertices. Remember that each scalar product gives one linear equation in unknowns. Since party P_i owns m_i vertices, it gets m_i linear equations in m_j unknowns. If $m_i > m_j$, this will completely breach the security of party P_j. Thus, if there is even one party that has more vertices than any of the other parties, it can completely breach the security of the other parties. Since this situation is quite likely, local scalar products cannot be used.

Instead to ensure security, we must carry out the entire scalar product in one go. To do this securely, we must ensure that all of the vectors are encrypted and permuted by all of the parties, thus ensuring that no linkage between vectors can be done. Now, after intersection a party can only learn the total scalar product (not any of its components). Algorithm 2 gives the complete details. This still gives it some linear equations – in fact, it gives in m_i linear equations in $n - m_i$ unknowns. As long as m_i is not more than half of the total number of vertices, security is not breached. In most situations this will be true and this protocol can be used. In cases where this is not true, there is no alternative to the first completely secure protocol.

4.2 Complexity Analysis

We now analytically show that this method is more efficient than the approach in [2]. Assume that c denotes the total number of 1s in the global adjacency matrix in a particular iteration. Further assume that these are split into c_i $1s$ for each row and c'_i $1s$ for each column. Thus, $c = \sum_{i=1}^{n} c_i$. Similarly, $c = \sum_{i=1}^{n} c'_i$.

Algorithm 2. Commutative Encryption based Boolean Scalar Product

Require: k parties, $P_1, \ldots, P_k$
Require: Party P_1 has input vector $\boldsymbol{X} = \{x_1, \ldots, x_n\}$
Require: Party $P_1, \ldots, P_n$ each have input vectors $\boldsymbol{Y_i} = \{y_1, \ldots, y_{m_i}\}$, where m_i represents the number of vertices owned by each party, such that $\sum_{i=1}^{k} m_i = n$
Require: Assume that $\boldsymbol{Y} = [\boldsymbol{Y_1} \ldots \boldsymbol{Y_k}]$
Require: P_1 gets output o such that $o = \boldsymbol{X} \cdot \boldsymbol{Y}$
Require: A global position encoding scheme
1: Each party P_i generates a private and public key pair (sk_i, pk_i) for a commutative encryption system.
2: P_1 converts its vector $\boldsymbol{X}$ to the position set XS
3: Each party P_i converts its local vector $\boldsymbol{Y_i}$ to the position set YS_i based on the global encoding scheme
4: **for** $i = 1 \ldots n$ **do**
5: P_1 encrypts the position set XS with its key E_{pk_1} to get the encrypted vector EXS
6: **end for**
7: **for** $j = 1 \ldots k$ **do**
8: Each Party P_j encrypts its local position set YS_j with its key E_{pk_j} to get the encrypted position set EYS_j
9: **end for**
10: Each party passes its encrypted position set to the next party for encryption with its key until all sets are encrypted by all parties
11: At P_k: $EYS \leftarrow \phi$
12: **for** $j = k \ldots 2$ **do**
13: Party P_j merges its completely encrypted set with the global encrypted set EYS, i.e. $EYS \leftarrow EYS \cup EYS_j$
14: Party P_j arbitrarily permutes EYS and sends it to party P_{j-1}
15: **end for**
16: At P_1: Receive EYS from P_2 and merge EYS_1 into it (i.e., $EYS \leftarrow EYS \cup EYS_1$)
17: P_1 intersects the completely encrypted set EXS with the complete encrypted set EYS to get the output o

In each iteration, for each point in the global adjacency matrix, one efficient commutative encryption based scalar product is carried out. Thus, for row p and column q, the scalar product requires $c_p * k + c'_q * k$ encryptions. Thus, the total cost of each iteration can be given by summing the total number of iterations required for each row and column. However, this assumes that we reencrypt for every row and column for each scalar product, which is quite unnecessary. In reality, it is sufficient to encrypt each row and each column only once. The same encryptions can be used for successive scalar products without revealing any extra information. Thus, total cost, TC is

$$TC = \sum_{p=1}^{n} c_p * k + \sum_{q=1}^{n} c'_q * k = k \sum_{p=1}^{n} c_p + k \sum_{q=1}^{n} c'_q = kc + kc = 2kc$$

In general, c can range between n and n^2. Therefore, in the best case, $TC = 2kn$, while in the worst case, $TC = 2kn^2$. It is important to note that for large distributed graphs typically the values of c are asymptotically closer to the best case value rather

than the worst case. For instance in the distributed graph linking financial transactions across different bank accounts, it is unlikely that a single transaction can be linked to all transactions or even a fraction of these. We can safely assume that the number of transactions that can be linked to a single transaction will always be bounded by a constant, i.e., $c = O(n)$.

5 A Heuristic Approach Based on Representative Selection

In this section, we present a heuristic approach to improve the efficiency of secure computation for those situations where the commutative encryption based approach cannot be used. The basic idea is to have each party choose a fraction of its overall nodes as its representatives set to form the representative matrix. Then, the secure transitive closure is done only over the representative matrix. The global transitive closure is now inferred using the representative transitive closure. With a small representative matrix, this clearly leads to significantly smaller computation costs. However, this pays a price in accuracy. While the links between the representative are accurately discovered, for the remaining nodes, the links may or may not be discovered. While there will be no false positive (a link found where none exists), there can be significant false negatives, based on how few representatives are chosen. Algorithm 3 gives the details.

We need to further discuss two issues – how are the representatives chosen, and how is the global transitive closure inferred from the representative transitive closure. We first discuss the second issue: Specifically, given any pair of nodes (u, v), where $u \in V_i$ and $v \in V_j (i \neq j)$, evaluate DIRPATH(u, v). The following 3 cases may occur:

- both u and v are representatives: in this case, we can directly get the answer from the transitive closure T_R.
- one of u and v is a representative: without loss of generality, we assume that u is a representative. If we can find a node v' which is a representative of the party who owns v and we also know there is a path from u to v' and a path from v' to v based on step 3 and step 1, respectively, then we say a path exists from u to v. Otherwise, no path exists between them.
- neither u nor v is a representative: If we can find a node u' (resp. v') which is a representative of the party who owns u(resp. v) and we also know there is a path from u to u' and a path from v' to v based on step 1, as well as a path exists from u' to v' based on step 3, then we say a path exists from u to v. Otherwise, no path exists between them.

Now, for the first question – how do we select the representatives to maximize accuracy for a given level of efficiency.

Intuitively, the representatives should be chosen from the set of the inter-nodes in each subgraph. This makes sense, since these are the only nodes involved in any edges with intra-edges. These are our only sources of cross-graph information. As we have explained above, combining our cross-graph path information with the local paths in each subgraph will help us to discover the path between any pair of nodes residing in different subgraphs. Assuming that the inter-nodes is a fraction of total nodes, choosing inter-nodes as representatives would easily reduce the required secure computations.

If we include all the inter-nodes as representatives, obviously we would get the exact results of all path information.

In this paper, we employ a greedy heuristic to choose representatives from within the inter-nodes. The idea is to choose representative nodes with high degrees. The intuition is that a node with higher degree should be involved in more paths and thus contribute more information.

One natural and seemingly better selection criteria is to have a greedy global selection - we choose representatives with high degree in the overall global graph G. However, given that each party can only see its own subgraph, this is not ideal, since it would not address privacy concerns.

Instead, we take the approach of greedy local representatives selection. In a greedy local selection, we choose the inter-nodes which have high degree in each party's local subgraph. Again, three different kinds of degree could be used - inter-degree, local-degree, and combined-degree (as we have defined in Section 2). In each case, the corresponding degree of each node is computed. Then, each party keeps a specified percentage of its local nodes with highest degrees being its representatives, which are used for forming representative matrix R. Algorithm 4 presents the details of this. As we show in the experimental evaluation below, this works quite well.

5.1 Experimental Evaluation

In this section we experimentally evaluate the effectiveness of our proposed algorithms. Synthetic random graphs are generated for the test datasets. In fact, random graphs are widely used in the probabilistic method, where one tries to prove the existence of graphs with certain properties. The existence of a property on a random graph implies, via the famous Szemeredi regularity lemma, the existence of that property on almost all graphs [13].

Since we need to have k sub-graphs making up a global graph, we use the igraph package [1] to generate a specific type of global graph G (i.e, Erdős-Renyi random graph). Then, we uniformly at random choose a certain number of nodes (i.e, a specified percentage of the total number of nodes n) to induce each subgraph G_i and the corresponding inter-edges between these subgraphs. In the ER $G(n, p)$ model, a graph is constructed by connecting nodes randomly, where n is the total number of nodes, and p is the probability that each edge is included in the graph, with the presence or absence of any two distinct edges in the graph being independent. We partition the global graph G into 4 equal-size subgraphs, each of which is assumed to be owned by a party. This partition also results in a number of inter-edges connecting the subgraphs.

Figure 1 shows the accuracy results of the greedy local, greedy global, and random approaches with the representative rate goes from 10% up to 100%. All the tests are done on the Erdős-Renyi graphs with the number of nodes n = 1000 and probability p = 0.1%. In the results, lid, lld, and lcd to stand for the local inter-degree, local local-degree, and local combined-degree approach, respectively. In the global case, the global inter-degree, global local-degree, global combined-degree is denoted by gid, gld, and gcd, respectively. The representative rate is the fraction of representative nodes chosen

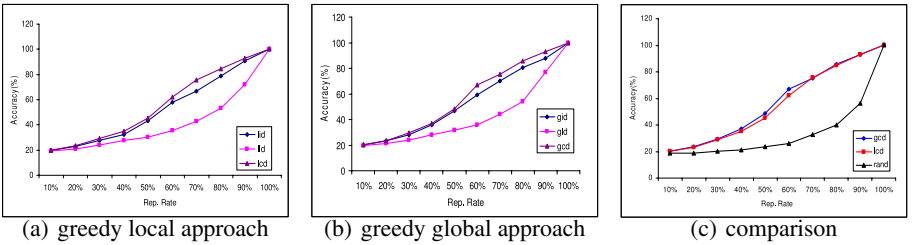
[1] http://cneurocvs.rmki.kfki.hu/igraph/

Algorithm 3. Secure Representatives Approach

Require: k parties $P_1, \ldots, P_k$
Require: Let the matrix $A^{(i)}(n_i \times n_i)$ represent the local adjacency matrix of party P_i
Require: Let the matrix $R(|V_R| \times |V_R|)$ be the representative matrix (i.e., the matrix entry $R[p,q] = 1$ if $p \in V_R^{(i)}, q \in V_R^{(j)}$, otherwise 0.)
1: Each party P_i computes its local transitive closure $T_L^{(i)}$ with input matrix $A^{(i)}$
2: Each party P_i engages in heuristically choosing representatives (Algorithm 4) to have the matrix R
3: Each party P_i participates secure transitive closure computation described in Section 3 with input matrix R to get transitive closure T_R
4: Given a pair of nodes (u, v), where $u \in V_i$ and $v \in V_j (i \neq j)$:
5: **if** both u and v are representatives {i.e.($u \in V_R^{(i)}$ and $v \in V_R^{(j)}$)} **then**
6: DIRPATH$(u, v) \leftarrow T_R(u, v)$
7: **else if** one of u and v is a representative {w.l.o.g assume that u is the representative i.e., $u \in V_R^{(i)}$ and $v \notin V_R^{(j)}$} **then**
8: **if** $\exists v' \in V_R^{(j)}$ and $T_R(u, v') \neq 0$ and $T_L^{(i)}(v', v) \neq 0$ **then**
9: DIRPATH$(u, v) \leftarrow 1$
10: **else**
11: DIRPATH$(u, v) \leftarrow 0$
12: **end if**
13: **else**
14: {neither is a representative, i.e., $u \notin V_R^{(i)}$ and $v \notin V_R^{(j)}$}
15: **if** $\exists u' \in V_R^{(i)}$ and $\exists v' \in V_R^{(j)}$ s.t. $T_L^{(i)}(u, u') \neq 0$ and $T_L^{(j)}(v', v) \neq 0$ and $T_R(u', v') \neq 0$ **then**
16: DIRPATH$(u, v) \leftarrow 1$
17: **else**
18: DIRPATH$(u, v) \leftarrow 0$
19: **end if**
20: **end if**
21: **return** DIRPATH(u, v)

either locally or globally from the inter-nodes. Each approach is run on the same graph, and the results averaged over five runs (with different graphs).

Figure 1(a) shows that, in the greedy local case, both the combined-degree and the inter-degree approaches achieve a better accuracy than the local-degree one, and the



(a) greedy local approach (b) greedy global approach (c) comparison

Fig. 1. Local vs. global vs. random approaches

Algorithm 4. Choose Representatives using Heuristics (DegreeType, Percentage): a greedy local approach

Require: k parties $P_1, \ldots, P_k$ each holding subgraph $G_i(V_i, E_i)$ as parts of global graph $G(V, E)$
Require: DegreeType: chosen from inter-degree, local-degree, or combined-degree
Require: Percentage: representative rate as opposed to the total number of the inter-nodes ($|V_{\mathrm{I}}|$)

1: At P_i:
2: the representative set $RS_i \leftarrow V_{\mathrm{I}}^{(i)}$
3: **for** each node $u \in V_{\mathrm{I}}^{(i)}$ **do**
4: **if** DegreeType = inter-degree **then**
5: Count the inter-degree $D_{\mathrm{I}}(u)$ of u
6: **else if** DegreeType = local-degree **then**
7: Count the local-degree $D_{\mathrm{L}}(u)$ of u
8: **else**
9: Count the combined-degree $D_{\mathrm{C}}(u)$ of u
10: **end if**
11: **end for**
12: Sort RS_i in terms of the degree counts
13: Keep Percentage*$|V_{\mathrm{I}}|$ of nodes with the highest degree in RS_i
14: Each party participates the forming of the matrix R using RS_i (Similar to the formation of adjacency matrix A discussed in Section 2)
15: **return** R

combined-degree is slightly better than the inter-degree. The greedy global approach in figure 1(b) looks similar to the greedy local one. In addition, in both greedy local and greedy global cases, the combined-degree has the best performance. Figure 1(c) compares the global and local combined-degree with random selection. Clearly, both global and local approaches perform much better than the random one. More importantly, we can see that the greedy local combined-degree approach almost performs the same as the greedy global combined-degree. Hence, it demonstrates that our proposed greedy local heuristics approach is promising.

While these results are preliminary, we have run more experiments varying other parameters as well as the graph generation model. We do not report these due to space limitations, but they are quite similar and show that the representatives approach gives a compelling tradeoff of accuracy for efficiency.

6 Concluding Remarks

In this paper, we have proposed two different approaches to improve the efficiency for privacy-preserving link discovery in a complex and distributed network structure. The first approach trades off security for efficiency, while the second trades off accuracy for efficiency. Both of our approaches can reduce the prohibitive computational complexity of the currently existing solution for secure link discovery.

In our future work, we will consider other features of interest such as the degree of closeness of the entities (i.e., number of common neighbors, number of distinct

paths, length of the shortest path, etc) for link discovery and analysis. A more challenging problem is to figure out the maximum flow from one entity to another (the max flow problem). This can be instrumental in computing the amount of resources transported through multiple intermediaries which would be great interest in financial fraud detection.

References

1. Getoor, L., Diehl, C.P.: Link mining: a survey. SIGKDD Explorations 7(2), 3–12 (2005)
2. He, X., Shafiq, B., Vaidya, J., Adam, N.: Privacy-preserving link discovery. In: Wainwright, R.L., Haddad, H. (eds.) SAC, pp. 909–915. ACM, New York (2008)
3. Sweeney, L.: Privacy-enhanced linking. SIGKDD Explorations 7(2), 72–75 (2005)
4. Xu, J.J., Chen, H.: Fighting organized crimes: using shortest-path algorithms to identify associations in criminal networks. Decision Support Systems 38(3), 473–487 (2004)
5. Ganiz, M.C., Pottenger, W.M., Yang, X.: Link analysis of higher-order path in supervised learning datasets. In: Proceedings of the 4th SIAM Workshop on Link Analysis, Counterterrorism and Security (April 2006)
6. Mooney, R., Melville, P., Tang, L., Shavlik, J., Dutra, I., Page, D., Costa, V.: Relational data mining with inductive logic programming for link discovery. In: Proceedings of the National Science Foundation Workshop on Next Generation Data Mining, Baltimore, Maryland (2002)
7. Duan, Y., Wang, J., Kam, M., Canny, J.: A secure online algorithm for link analysis on weighted graph. In: Proceedings of SIAM Workshop on Link Analysis, Counterterrorism and Security (April 2005)
8. Agrawal, R., Srikant, R.: Privacy-preserving data mining. In: Proceedings of the 2000 ACM SIGMOD Conference on Management of Data, pp. 439–450 (2000)
9. Lindell, Y., Pinkas, B.: Privacy preserving data mining. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 36–54. Springer, Heidelberg (2000)
10. Vaidya, J., Clifton, C., Zhu, M.: Privacy-Preserving Data Mining. In: Advances in Information Security, 1st edn. Springer, Heidelberg (2005)
11. Goethals, B., Laur, S., Lipmaa, H., Mielikäinen, T.: On Secure Scalar Product Computation for Privacy-Preserving Data Mining. In: Park, C.-s., Chee, S. (eds.) ICISC 2004. LNCS, vol. 3506, pp. 104–120. Springer, Heidelberg (2005)
12. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: Introduction to Algorithms. McGraw-Hill Book Company, New York (1990)
13. Bollobás, B.: Random Graphs, 2nd edn. Cambridge University Press, Cambridge (2001)

On Link Privacy in Randomizing Social Networks

Xiaowei Ying and Xintao Wu*

University of North Carolina at Charlotte
{xying, xwu}@uncc.edu

Abstract. Many applications of social networks require relationship anonymity due to the sensitive, stigmatizing, or confidential nature of relationship. Recent work showed that the simple technique of anonymizing graphs by replacing the identifying information of the nodes with random ids does not guarantee privacy since the identification of the nodes can be seriously jeopardized by applying subgraph queries. In this paper, we investigate how well an edge based graph randomization approach can protect sensitive links. We show via theoretical studies and empirical evaluations that various similarity measures can be exploited by attackers to significantly improve their confidence and accuracy of predicted sensitive links between nodes with high similarity values.

1 Introduction

Social networks are of significant importance in various application domains such as marketing, psychology, epidemiology and homeland security. Many applications of social networks such as anonymous Web browsing require relationship anonymity due to the sensitive, stigmatizing, or confidential nature of relationship. For example, most people prefer to conceal the truth regarding their illegal or unethical behaviors which are customarily disapproved of by society.

One natural approach is to publishing a node-anonymized version of the network that permits useful analysis without disclosing the identity of the individuals represented by the nodes. The recent work [1, 4] pointed out that this simple technique of anonymizing graphs by replacing the identifying information of the nodes with random ids does not guarantee privacy since the identification of the vertices can be seriously jeopardized by applying subgraph queries. Another approach is to randomizing edges to protect sensitive links [3, 4, 6, 8, 11]. For example, we can remove some true edges and/or add some false edges. After the randomization, the randomized graph is expected to be different from the original one. As a result, the true sensitive or confidential relationship will not be much disclosed even if the identification of the vertices is achieved by attackers.

We will explore how well the edge randomization can protect those sensitive links. In [8], Ying and Wu preliminarily investigated the relationship between the amount of randomization and the attacker's ability to infer the presence of a link and presented a randomization strategy that can preserve the spectral properties (and utility) of the graph. However, the effect on privacy due to randomization was quantified by considering only the magnitude information of randomization. It has been well known that

* This work was supported in part by U.S. National Science Foundation IIS-0546027 and CNS-0831204.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 28–39, 2009.
© Springer-Verlag Berlin Heidelberg 2009

graph topological features have close relations with the existence of links and various proximity measures have been exploited to predict the existence of a future link [5]. In this paper, we will investigate formally how attackers may exploit proximity measure values (derived from the released randomized graph) to breach link privacy. Privacy of a sensitive link is jeopardized if attackers' confidence of prediction is higher than some tolerated threshold or is significantly greater than the a-priori belief (without the exploit of the released randomized data). Hence it is of great importance for data owners to be aware of potential attacks and quantify the magnitude of perturbation to better protect sensitive links.

2 Related Work

Social network analysis has increasing interest in the database, data mining, and theory communities. The current state of the art is that there has been little work dedicated to privacy preserving social network analysis with the exception of some very recent work [1–4, 6, 8–11].

In [1], Backstrom and et al. described a family of attacks such that an adversary can learn whether edges exist or not between specific targeted pairs of nodes from node-anonymized social networks. Similarly in [4], Hay and et al. further observed that the structure of the graph itself (e.g., the degree of the nodes or the degree of the node's neighbors) determines the extent to which an individual in the network can be distinguished.

In [6], Liu and Terzi investigated how to modify a graph via a set of edge addition (or deletion) operations in order to construct a new k-degree anonymous graph, in which every node has the same degree with at least $k-1$ other nodes. In [11], Zhou and Pei anonymized the graph by generalizing node labels and inserting edges until each neighborhood is indistinguishable to at least $k-1$ others. In [2, 10], authors applied a structural anonymization approach called *edge generalization* that consists of collapsing clusters together with their component nodes' structure, rather than add or delete edges from the social network dataset. Although the above proposed approaches would preserve privacy, however, it is not clear how useful the anonymized graph is since many topological features may be lost.

The problems of how to generate a synthetic graph preserving various topological features of a real social network and how attackers may exploit the topological features of the released graph to breach link privacy were recently studied in [9]. However, the attacking model in [9] was based on the probability of existence of a link across all possible graphs in the graph space. In this paper, the attacking model is to exploit the relationship between existence of a link and the similarity measure values of node pairs in one released randomized graph.

We would point out that our problem of attacking methods on a randomized graph is different from the classic link prediction problem investigated in [5]. The classic link prediction focuses on network evolution models and is to predict the existence of a future link between two nodes given a snapshot of a current social network. The change due to randomization is different with that due to network evolutions. Nevertheless, various graph proximity measures used in the classic link prediction could be used by attackers.

3 Link Privacy Analysis

A network $G(n, m)$ is a set of n nodes connected by a set of m links. The network considered here is binary, symmetric, connected, and without self-loops. Let $A = (a_{ij})_{n\times n}$ be its adjacency matrix, $a_{ij} = 1$ if node i and j are connected and $a_{ij} = 0$ otherwise. $\widetilde{G}$ is the randomized graph obtained by randomly adding k false edges followed by deleting k true edges. This strategy keeps the total number of edges in the original graph unchanged. We denote $\widetilde{A} = (\tilde{a}_{ij})_{n\times n}$ be the adjacency matrix of $\widetilde{G}$.

When it comes to link privacy, it is usually $a_{ij} = 1$ that people want to hide, not $a_{ij} = 0$ and attackers are capable of calculating posterior probabilities. Formally, we use $P(a_{ij} = 1)$ to denote the users' prior belief about the event of $a_{ij} = 1$ and use $P(a_{ij} = 1|\widetilde{G})$ to denote its posterior belief about $a_{ij} = 1$. The released graph $\widetilde{G}$ is regarded as jeopardizing the privacy if $P(a_{ij} = 1|\widetilde{G}) > P(a_{ij} = 1)$.

In [8], we preliminarily investigated the relationship between the amount of randomization and the attacker's ability to infer the presence of a link. The results are shown as follows. When the attacker knows only parameter m and n, the prior belief is

$$P(a_{ij} = 1) = \frac{2m}{n(n-1)}. \tag{1}$$

With the released graph and perturbation parameter k, the posterior belief is

$$P(a_{ij} = 1|\tilde{a}_{ij} = 1) = \frac{m-k}{m}, \quad P(a_{ij} = 1|\tilde{a}_{ij} = 0) = \frac{k}{\binom{n}{2} - m} \tag{2}$$

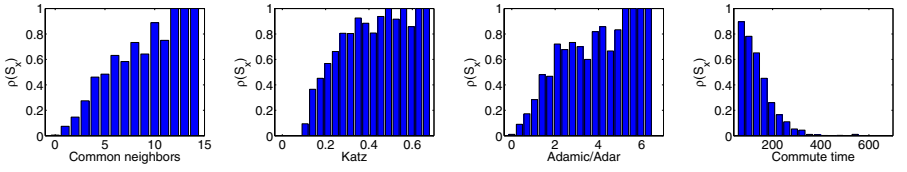
Equation 2 is based on the Addition/Deletion without replacement[1].

In this paper, we further investigate whether topological features of the released network can be exploited by attackers to breach the link privacy. More specifically, we focus on to what extent a given sensitive relationship can be breached by attackers who exploit proximity measure values of node pairs. Proximity measures have been shown to be effective in the classic link prediction problem (i.e., predicting the future existence of links among nodes given a snapshot of a current graph). However, link prediction in our context is to predict the likelihood of existence of original links from the randomized graph. This is challenging since the proximity measure values calculated from the randomized graph can be varied from those of the original graph. In section 3.1, we empirically show the close relationship between various similarity measures of node pairs and probability of link existence between them. In section 3.2, we conduct theoretical studies and quantify how much the posterior belief can be enhanced by exploiting those similarity measures.

3.1 Existence of a Link vs. Similarity Measure

Let m_{ij} be a similarity measure on node pair (i, j) in graph G (a larger value of m_{ij} indicates that nodes i and j are more similar). We apply four similarity measures in

[1] Refer to [8] for the Addition/Deletion with replacement. For large graphs, the difference between the above is small.



(a) Common neighbors (b) Katz (c) Adamic/Adar (d) Commute time

Fig. 1. Similarity measure vs. the prob. of true edges in the original graph ($\rho(S_x)$) for polbooks

this paper. The first one is the number of common neighbors: $CN_{ij} = \sum_{k=1}^{n} a_{ik}a_{kj}$. The second one is the Adamic/Adar measure , which is the weighted number of common neighbors. The weights are assigned based on the information theory: $Ad_{ij} = \sum_{k=1}^{n} \frac{1}{\log d_k} a_{ik}a_{kj}$, where d_k is the degree of node k. The third one is the Katz measure, which is a weighted sum of the number of paths in the graph that connect two nodes, with shorter paths being given the larger weight: $K_{ij} = \sum_{k=1}^{\infty} \beta^k P_{ij}^{(k)}$, where $P_{ij}^{(k)}$ denotes the number of paths from i to j with length equal to k while β is a damping factor. In this paper, we take $\beta = 0.1$. The fourth one is the commute time CT_{ij}, which is the expected steps of random walks from i to j and back to i. The commute time is a distance measure: more similar nodes have smaller CT values.

Let $\rho(\Omega)$ denote the proportion of true edges in the set of node pairs Ω:

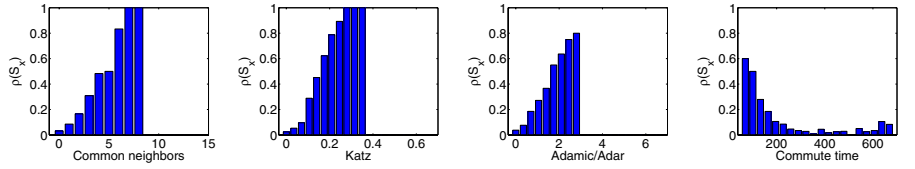
$$\rho(\Omega) = \frac{1}{|\Omega|} \sum_{(i,j)\in\Omega} a_{ij},$$

where $|\Omega|$ denotes the number of elements in set Ω. Let $S_x = \{(i,j) : m_{ij} = x\}$ denote the set of all node pairs with the similarity measure $m_{ij} = x$. Hence $\rho(S_x)$ denotes the proportion of true edges in the S_x, which can be considered as the probability of existence of a link between node pair (i,j) in S_x. Next, we empirically show how $\rho(S_x)$ varies with x in real social networks.

Figure 1 shows how the proportions of true edges in S_x are varied with similarity measure values x in terms of four measures (Common neighbors, Katz, Adamic/Adar, and Commute time) in the US political books network (polbooks). The polbooks network[2] contains 105 nodes and 441 edges, and nodes represent books about US politics sold by the online bookseller Amazon.com while edges represent frequent co-purchasing of books by the same buyers on Amazon. We can observe that $\rho(S_x)$ increases with x. In other words, the probability that $a_{ij} = 1$ is highly correlated with similarity measure m_{ij}: the larger m_{ij} is, the more likely a_{ij} is equal to 1.

We then perturbed the polbooks network by adding 200 false edges and deleting 200 true edges. From the perturbed graph $\widetilde{G}$, we define $\widetilde{S}_x = \{(i,j) : \tilde{m}_{ij} = x\}$ as the set of node pairs with similarity measure $\tilde{m}_{ij} = x$. Figure 2 shows how the proportions of true edges in $\widetilde{S}_x$ (i.e., the probability of existence of a link) are varied with similarity measure values x in terms of four measures in the randomized polbooks network. We

[2] (`http://www-personal.umich.edu/~mejn/netdata/`)



(a) Common neighbors (b) Katz (c) Adamic/Adar (d) Commute time

Fig. 2. Similarity measure vs. the prob. of true edges in the randomized graph ($\rho(\widetilde{S}_x)$) for polbooks

can observe that the same pattern still holds even if the randomized graph itself is quite different from the original one (200 false edges out of 441 edges). In the next section, we will show how attackers exploit $\tilde{m}_{ij}$ in the perturbed graph $\widetilde{G}$ to improve their posterior belief on existence of a true link between nodes (i, j) in the original graph.

3.2 Link Prediction by Exploiting Similarity Measure

In this section, we quantify how much the posterior belief can be enhanced by exploiting similarity measure between two node (i, j) in the randomized graph. We present our quantification in a series of results and leave detailed proofs in Appendix.

Recall the randomization strategy is to randomly add k false edges followed by deleting k true edges. In other words, every true link is to be deleted independently with probability p_1 and every non-existing link is to be added independently with probability p_2. We can easily derive $p_1 = k/m$ and $p_2 = k/[\binom{n}{2} - m]$.

Let $\tilde{m}_{ij}$ denote the similarity measure of node i and j in $\widetilde{G}$. We define $\widetilde{S}_x = \{(i, j) : \tilde{m}_{ij} = x\}$ as the set of node pairs with $\tilde{m}_{ij} = x$ in the perturbed graph. Then we have $P(a_{ij} = 1|\tilde{m}_{ij} = x) = \rho(\widetilde{S}_x)$, and $P(a_{ij} = 0|\tilde{m}_{ij} = x) = 1 - \rho(\widetilde{S}_x)$. Recall that $\rho(\widetilde{S}_x)$ denotes the proportion of true edges in the set $\widetilde{S}_x$ derived from the perturbed graph. Also notice that $P(\tilde{a}_{ij} = 1|a_{ij} = 1) = 1 - p_1$ and $P(\tilde{a}_{ij} = 1|a_{ij} = 0) = p_2$. With the Bayes' theorem, the posterior belief is then given by

$$P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij} = x) = \frac{(1 - p_1)\rho(\widetilde{S}_x)}{(1 - p_1)\rho(\widetilde{S}_x) + p_2[1 - \rho(\widetilde{S}_x)]}, \tag{3}$$

$$P(a_{ij} = 1|\tilde{a}_{ij} = 0, \tilde{m}_{ij} = x) = \frac{p_1\rho(\widetilde{S}_x)}{p_1\rho(\widetilde{S}_x) + (1 - p_2)[1 - \rho(\widetilde{S}_x)]}. \tag{4}$$

Equation 3 (Equation 4) shows the enhanced posterior belief that an observed (missing) edge (i, j) in the $\tilde{G}$ is a true edge in G. The following property shows that the event of an observed link $\tilde{a}_{ij} = 1$ usually has more indications to be a true link than that of $\tilde{a}_{ij} = 0$.

Property 1. *Let r denote the sparse ratio of the graph, $r = m/\binom{n}{2}$. If $k \leq (1 - r)m$, given a fixed x, we have the following inequality stands:*

$$P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij} = x) \geq P(a_{ij} = 1|\tilde{a}_{ij} = 0, \tilde{m}_{ij} = x). \tag{5}$$

Many real-world social networks are very sparse ($r \approx 0$). Hence $k \leq (1-r)m$ is usually satisfied. We thus focus on the risk of the released links, $P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij} = x)$.

One issue here is that attackers cannot know the proportion of true edges in $\widetilde{S}_x$ from the perturbed graph. What they can know actually is the proportion of observed edges in $\widetilde{S}_x$. Our next result shows the maximum likelihood estimate of $\rho(\widetilde{S}_x)$ can be derived from the proportion of observed edges in $\widetilde{S}_x$.

Result 1. *Given the perturbed graph and a fixed x, define $\widetilde{S}_x^1 = \widetilde{S}_x \cap \widetilde{E} = \{(i,j) : \tilde{a}_{ij} = 1, \tilde{m}_{ij} = x\}$. Assume $p_1 + p_2 \neq 1$, then the maximum likelihood estimator (MLE) of $\rho(\widetilde{S}_x)$ is given by*

$$\hat{\rho}\left(\widetilde{S}_x\right) = \frac{|\widetilde{S}_x^1|/|\widetilde{S}_x| - p_2}{1 - p_1 - p_2}, \tag{6}$$

and the MLE is unbiased.

By replacing $\rho(\widetilde{S}_x)$ in Equation 3 with $\hat{\rho}(\widetilde{S}_x)$ (shown in Equation 6), we have derived our enhanced posterior belief $P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij} = x)$. Attackers may simply calculate the posterior belief of all node pairs in the perturbed graph and choose top-t node pairs as predicted candidate links.

For those similarity measures with continuous ranges (e.g., commute time), the number of node pairs with similarity measure equal exactly to x is usually small. In practice, we can apply histogram approximation or use the kernel estimator to smooth the estimation.

We would emphasize that our enhanced posterior belief $P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij} = x)$ more accurately reflect the existence of a true link than the posterior belief $P(a_{ij} = 1|\tilde{a}_{ij} = 1)$ without exploiting the similarity measure derived in previous work [8]. We can see that $P(a_{ij} = 1|\tilde{a}_{ij} = 1)$ (shown in Equation 2) is the same for all observed links. On the contrary, our enhanced posterior belief $P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij} = x)$ tends to be larger for those observed links with higher similarity values, and tends to be smaller for links with lower similarity values. Hence, it can more accurately reflect the existence of true links. We show our theoretical explanations in Results 2 and 3 and will compare the precisions of top-t predicted links derived from these two posterior beliefs in our empirical evaluations.

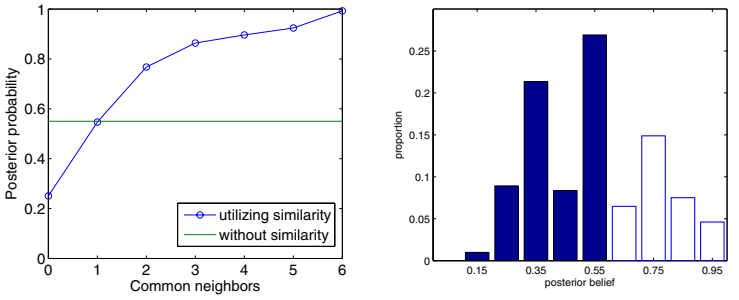
Result 2. *$P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij} = x)$ is an increasing function of $\rho(\widetilde{S}_x)$, and when $\rho(\widetilde{S}_x) \geq \frac{p_2}{p_1+p_2}$, we have the following inequality stands:*

$$P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij} = x) \geq P(a_{ij} = 1|\tilde{a}_{ij} = 1). \tag{7}$$

Our next result shows more clearly the relationship between a-priori belief (Equation 1), posterior belief without exploiting similarity measures (Equation 2), and our enhanced posterior belief with exploiting similarity measures (Equations 3 4).

Result 3. *Both the sum of a-priori belief over all node pairs and the sum of posterior belief (without exploiting similarity measures) overall all node pairs are equal to the number of edges:*

$$\textstyle\sum_{i<j} P(a_{ij} = 1) = \sum_{i<j} P(a_{ij} = 1|\tilde{a}_{ij}) = m.$$



(a) Posterior belief vs. Common neighbors

(b) Posterior belief distribution

Fig. 3. Posterior belief for polbooks network

The sum of our enhanced posterior belief (with exploiting similarity measures) also approaches to the number of edges:

$$\sum_{i<j} P(a_{ij} = 1|\tilde{a}_{ij}, \tilde{m}_{ij}) \to m \quad as \quad n \to \infty.$$

Figure 3 shows the relationship between the two posterior beliefs and the common neighbors for the polbooks data. We set $k = 200$. We can observe that the posterior belief without exploiting the similarity measure, $P(a_{ij} = 1|\tilde{a}_{ij} = 1)$, is 0.55 for all observed links. However, our enhanced posterior belief $P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij})$ are greater than 0.55 for those links with more than 2 common neighbors as shown in Figure 3(a). Figure 3(b) shows the distribution of the calculated posterior belief values. We can observe that 33.5% of released links have their posterior beliefs enhanced with similarity measures.

3.3 Privacy Protection Measure

In the privacy preserving data mining, one natural question from data owner is how many perturbations we need such that we can guarantee the protection for all sensitive individual edges are above some tolerated threshold. When attackers utilize the similarity measure, the absolute measure of protection for an individual link (i, j) can be defined as

$$\tau_a(i, j) = 1 - \max_x \left\{ \max_{t=0,1} P(a_{ij} = 1|\tilde{a}_{ij} = t, \tilde{m}_{ij} = x) \right\} \tag{8}$$

where the second term denotes the maximal suspicion of existing $a_{ij} = 1$. Compared with the protection under the attack without exploiting similarity measures, we define the relative measure of protection as

$$\tau_r(i, j) = \frac{\tau_a(i, j)}{1 - \max_{t=0,1} P(a_{ij} = 1|\tilde{a}_{ij} = t)}.$$

The measures of protection (τ_a and τ_r) are defined in terms of one individual edge. In the privacy preserving data mining, one natural question is how many perturbations we need such that we can guarantee the protection for all individual edges are above the threshold. Our next result shows the formula of the minimum number of perturbations to achieve the protection of all individual links. It is of great importance to evaluate the relationship between the required minimum number of perturbations and the utility loss of the perturbed graph. Due to space limitations, we leave this as our future work.

Result 4. *In the original graph, let* $S_x = \{(i,j) : m_{ij} = x\}$, $\rho_{\max} = \max_x \rho(S_x)$, *and sparse ratio* $r = m/\binom{n}{2}$. *When the protection threshold* $\epsilon < \frac{1-\rho_{\max}}{1-r}$, *there exists the minimum* k *such that* $\tau_r(i,j) \geq \epsilon$ *stands for all the node pair* (i,j) *is given by:*

$$k_{\min} = \frac{[(1-r)\epsilon\rho_{\max} - r(1-\rho_{\max})]m}{\epsilon(\rho_{\max} - r)}. \tag{9}$$

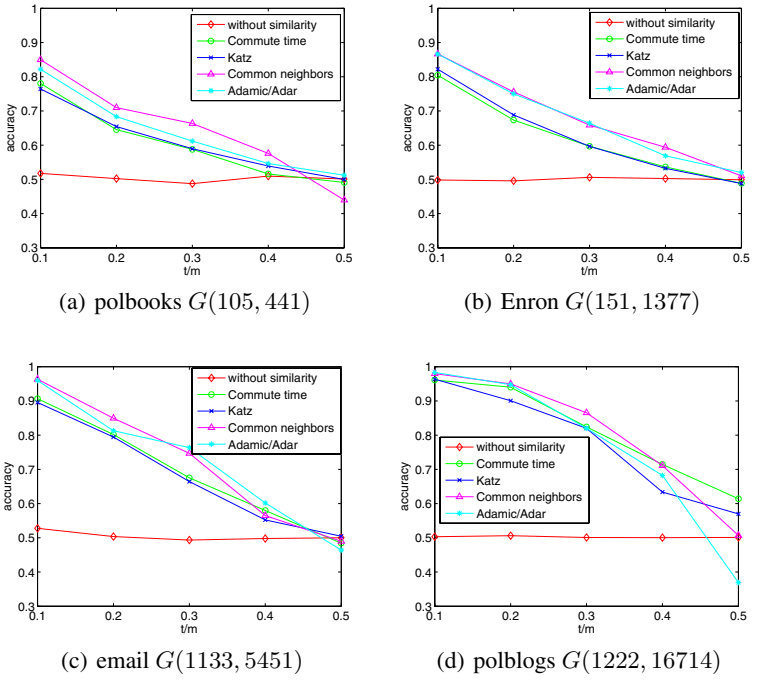
4 Empirical Evaluation

We used four network data sets (*polbooks, Enron, email, polblogs*) in our evaluation. The *Enron* network was built from email corpus of a real organization over the course covering a 3 years period. We used a pre-processed version of the dataset provided by [7]. This dataset contains 252,759 emails from 151 Enron employees, mainly senior managers. The *email* graph is the network of e-mail interchanges between members of the Univeristy Rovira i Virgili (Tarragona)[3]. The *polblogs* compiles the data on the links among US political blogs, containing over 1,000 vertices and 15,000 edges, which is based on incoming and outgoing links and posts around the time of the 2004 presidential election[4].

For each graph G, we randomly add k false edges and delete k true edges. We set $k = 0.5m$ in this paper, which corresponds to a relatively large perturbation. We also conducted evaluations with other k values and skip their results due to space limitations. We applied four similarity measures (Common neighbors, Katz, Adamic/Adar, Commute time) to predict top-t candidate links. We varied t values from $0.1m$ to $0.5m$ for all four data sets.

For each t, we calculated the precision of prediction links with different similarity measures. We also calculated the precision of prediction links using the posterior belief without exploiting the similarity measure. Figure 4 plots our results on four data sets. We can observe that for all four data sets we can achieve very high accuracy (greater than 0.8) by using our enhanced posterior belief for a subset (top $0.1m$) of released links, which indicates severe privacy disclosures for those sensitive links. We can also see that our enhanced posterior belief achieve higher precisions than the previous posterior belief without exploiting similarity measures for most links ($0.5m$) with high similarity measure values, indicating that the network topology does indeed contain latent information from which to infer interactions. From Figure 4, we can also observe that we achieve different precisions using different similarity measures: one measure which

[3] `http://deim.urv.cat/~aarenas/data/welcome.htm`
[4] `http://www-personal.umich.edu/~mejn/netdata/`



(a) polbooks $G(105, 441)$ (b) Enron $G(151, 1377)$

(c) email $G(1133, 5451)$ (d) polblogs $G(1222, 16714)$

Fig. 4. Precision of top t predictions by the posterior belief w/o similarity measures for four data sets

achieves the highest precision for one data set is not necessarily the one for another data set. It is of great significance to explore what similarity measures can be exploited by attackers to achieve the highest privacy disclosure for a given social network. We will investigate this in our future work.

5 Conclusion and Future Work

In this paper, we have investigated how well the edge randomization approach via addition/deletion can protect privacy of sensitive links. We have conducted theoretical analysis and empirical evaluations to show that node proximity measures can be exploited by attackers to enhance the posterior belief and prediction accuracy of the existence of sensitive links among nodes with high similarity values.

There are some other aspects of this work that merit further research. Among them, we will continue the line of this research by investigating other edge randomization approaches (e.g., edge switches). We will also investigate how the edge based randomization affects the graph's utility. We are interested in comparing theoretically and empirically the edge based randomization with the k-degree anonymization approaches [6, 11] in terms of the privacy vs. utility tradeoff.

References

1. Backstrom, L., Dwork, C., Kleinberg, J.: Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In: WWW 2007: Proceedings of the 16th international conference on World Wide Web, pp. 181–190. ACM Press, New York (2007)
2. Campan, A., Truta, T. M.: A clustering approach for data and structural anonymity in social networks. In: PinKDD (2008)
3. Hay, M., Miklau, G., Jensen, D., Towsely, D., Weis, P.: Resisting structural re-identification in anonymized social networks. In: VLDB (2008)
4. Hay, M., Miklau, G., Jensen, D., Weis, P., Srivastava, S.: Anonymizing social networks. University of Massachusetts Technical Report, 07-19 (2007)
5. Liben-Nowell, D., Kleinberg, J.: The link prediction problem for social networks. In: CIKM 2003: Proceedings of the twelfth international conference on Information and knowledge management, pp. 556–559. ACM, New York (2003)
6. Liu, K., Terzi, E.: Towards identity anonymization on graphs. In: Proceedings of the ACM SIGMOD Conference, Vancouver, Canada. ACM Press, New York (2008)
7. Shetty, J., Adibi, J.: The Enron email dataset database schema and brief statistical report. Information Sciences Institute Technical Report, University of Southern California (2004)
8. Ying, X., Wu, X.: Randomizing social networks: a spectrum preserving approach. In: Proc. of the 8th SIAM Conference on Data Mining (April 2008)
9. Ying, X., Wu, X.: Graph generation with prescribed feature constraints. In: Proc. of the 9th SIAM Conference on Data Mining (2009)
10. Zheleva, E., Getoor, L.: Preserving the privacy of sensitive relationships in graph data. In: Bonchi, F., Ferrari, E., Malin, B., Saygın, Y. (eds.) PInKDD 2007. LNCS, vol. 4890, pp. 153–171. Springer, Heidelberg (2008)
11. Zhou, B., Pei, J.: Preserving Privacy in Social Networks Against Neighborhood Attacks. Data Engineering, 2008. In: ICDE 2008. IEEE 24th International Conference on, pp. 506–515 (2008)

A Proofs

Proof of Property 1
It is easy to verify that inequality 5 stands if and only if $(1-p_1-p_2)[1-\rho(\widetilde{S}_x)] \geq 0$. We need only guarantee that $1-p_1-p_2 \geq 0$. Notice that $p_1 = k/m$ and $p_2 = k/[\binom{n}{2} - m]$, then $1 - p_1 - p_2 \geq 0$ if and only if $k \leq \left[1 - m/\binom{n}{2}\right] m = (1-r)m$. □

Proof of Result 1
Let $N = |\widetilde{S}_x|$, $N_1 = |\widetilde{S}_x^1|$ and $\rho = \rho(\widetilde{S}_x)$. Then, for a randomly selected node pair (i, j), $\tilde{a}_{ij}$ is a Bernoulli random variable:

$$P(\tilde{a}_{ij} = 1 | \tilde{m}_{ij} = x) = (1 - p_1)\rho + p_2(1 - \rho)$$
$$P(\tilde{a}_{ij} = 0 | \tilde{m}_{ij} = x) = p_1\rho + (1 - p_2)(1 - \rho)$$

Then the likelihood function of $\widetilde{S}_x$ is

$$L = [(1 - p_1)\rho + p_2(1 - \rho)]^{N_1} [p_1\rho + (1 - p_2)(1 - \rho)]^{N - N_1} .$$

Take derivative to $\ln L$ with respect of ρ, we have

$$\frac{d\ln L}{d\rho} = \frac{N_1(1-p_1-p_2)}{(1-p_1)\rho + p_2(1-\rho)} - \frac{(N-N_1)(1-p_1-p_2)}{p_1\rho + (1-p_2)(1-\rho)}.$$

Set $\frac{d\ln L}{d\rho} = 0$, we have $\hat{\rho} = \frac{N_1/N - p_2}{1-p_1-p_2}$, and the unbiasedness is then obvious. □

Proof of Result 2
Notice that $P(a_{ij} = 1|\tilde{a}_{ij} = 1) = \frac{m-k}{m} = 1 - p_1$, and with Equation 3, it is easy to verify this result. □

Proof of Result 3
$\sum_{i<j} P(a_{ij} = 1) = m$ is obvious. Notice that the number of edges does not change along the perturbation, then we have

$$\begin{aligned}\sum_{i<j} P(a_{ij} = 1|\tilde{a}_{ij}) &= \sum_{(i,j)\in\widetilde{E}} P(a_{ij} = 1|\tilde{a}_{ij} = 1) + \sum_{(i,j)\notin\widetilde{E}} P(a_{ij} = 1|\tilde{a}_{ij} = 0) \\ &= m \cdot \tfrac{m-k}{m} + \left[\tbinom{n}{2} - m\right] \cdot k/\left[\tbinom{n}{2} - m\right] = m. \end{aligned} \tag{10}$$

When attackers utilize the similarity measures with MLE, we first show

$$E[\textstyle\sum_{i<j} P(a_{ij} = 1|\tilde{a}_{ij}, \tilde{m}_{ij})] = m.$$

$$\begin{aligned} E\left[\sum_{i<j} P(a_{ij} = 1|\tilde{a}_{ij}, \tilde{m}_{ij})\right] = \sum_x \Bigg\{ &\sum_{(i,j)\in\widetilde{S}_x^1} E[P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij} = x)] \\ &+ \sum_{(i,j)\in\widetilde{S}_x - \widetilde{S}_x^1} E[P(a_{ij} = 1|\tilde{a}_{ij} = 0, \tilde{m}_{ij} = x)] \Bigg\} \end{aligned} \tag{11}$$

With the MLE in Equation 6, we have

$$\begin{aligned} &\textstyle\sum_{(i,j)\in\widetilde{S}_x^1} E[P(a_{ij} = 1|\tilde{a}_{ij} = 1, \tilde{m}_{ij} = x)] \\ =&\frac{(1-p_1)\hat{\rho}_1(\widetilde{S}_x)}{(1-p_1)\hat{\rho}_1(\widetilde{S}_x) + p_2[1-\hat{\rho}_1(\widetilde{S}_x)]}|\widetilde{S}_x^1| \\ =&(1-p_1)|\widetilde{S}_x|E[\hat{\rho}_1(\widetilde{S}_x)] \quad \text{(substitute Equation 6)} \\ =&(1-p_1)|\widetilde{S}_x|\rho(\widetilde{S}_x) \\ =&(1-p_1)\textstyle\sum_{(i,j)\in\widetilde{S}_x} a_{ij} \quad \text{(by the definition of } \rho(\cdot)) \end{aligned} \tag{12}$$

Similarly, we have

$$\textstyle\sum_{(i,j)\in\widetilde{S}_x - \widetilde{S}_x^1} E[P(a_{ij} = 1|\tilde{a}_{ij} = 0, \tilde{m}_{ij} = x)] = p_1 \sum_{(i,j)\in\widetilde{S}_x} a_{ij} \tag{13}$$

Combining Equation 11, 12 and 13 together, we have

$$E\left[\sum_{i<j} P(a_{ij}=1|\tilde{a}_{ij},\tilde{m}_{ij})\right] = \sum_x \sum_{(i,j)\in\widetilde{S}_x} a_{ij} = \sum_{i,j} a_{ij} = m.$$

Then, due to the law of large number, we can conclude that

$$\sum_{i<j} P(a_{ij}=1|\tilde{a}_{ij},\tilde{m}_{ij}) \to m \quad \text{as} \quad n\to\infty,$$

and we prove the result. □

Proof of Result 4

When $k \le (1-r)m$, with Result 1 and 2, we have that

$$\max_x \{\max_{t=0,1} P(a_{ij}=1|\tilde{a}_{ij}=t,\tilde{m}_{ij}=x)\} = P(a_{ij}=1|\tilde{a}_{ij}=1,\tilde{m}_{ij}=x_0),$$

where x_0 is the value such that $\rho(\widetilde{S}_x)$ is maximized: $\rho(\widetilde{S}_{x_0}) = \max_x \rho(\widetilde{S}_x)$. Let $\tilde{\rho}_{\max} = \rho(\widetilde{S}_{x_0})$. Meanwhile,we can also conclude

$$\max_{t=0,1} P(a_{ij}=1|\tilde{a}_{ij}=t) = P(a_{ij}=1|\tilde{a}_{ij}=1).$$

Then we have

$$\tau_r(i,j) = \frac{p_2[1-\tilde{\rho}_{\max}]}{p_1[(1-p_1)\tilde{\rho}_{\max} + p_2(1-\tilde{\rho}_{\max})]}. \tag{14}$$

Substitute $p_1 = \frac{k}{m} = \frac{k}{rN}$ and $p_2 = \frac{k}{N-m} = \frac{k}{(1-r)N}$ into Equation 14, we can verify that $\tau_r(i,j)$ is an increasing function of k, and the maximum value is $\frac{1-\tilde{\rho}_{\max}}{1-r}$ when $k=(1-r)m$.

When $k \ge (1-r)m$, we similarly have the following:

$$\max_x \{\max_{t=0,1} P(a_{ij}=1|\tilde{a}_{ij}=t,\tilde{m}_{ij}=x)\} = P(a_{ij}=1|\tilde{a}_{ij}=0,\tilde{m}_{ij}=x_0),$$
$$\max_{t=0,1} P(a_{ij}=1|\tilde{a}_{ij}=t) = P(a_{ij}=1|\tilde{a}_{ij}=0).$$

In this case, $\tau_r(i,j)$ is a decreasing function of k, and the maximum is also $\frac{1-\tilde{\rho}_{\max}}{1-r}$ when $k=(1-r)m$.

Therefore, $k_{\min}$ exists if and only if $\epsilon \le \frac{1-\tilde{\rho}_{\max}}{1-r}$, and $k_{\min} < (1-r)m$. Then, $\tau_r(i,j)$ is given by Equation 14. Solving the inequality $\tau_r(i,j) \ge \epsilon$, we have that

$$k \ge \frac{[(1-r)\epsilon\tilde{\rho}_{\max} - r(1-\tilde{\rho}_{\max})]m}{\epsilon(\tilde{\rho}_{\max}-r)}.$$

However, $\tilde{\rho}_{\max} = \max_x \rho(\widetilde{S}_x)$ varies from time to time due to the perturbation, and data owner can substitute it with the true maximum value $\rho_{\max} = \max_x \rho(S_x)$, then we get the result. □

Sentence-Level Novelty Detection in English and Malay

Agus T. Kwee, Flora S. Tsai, and Wenyin Tang

Nanyang Technological University,
School of Electrical & Electronic Engineering,
Singapore
atkwee@ntu.edu.sg,fst1@columbia.edu,wenyintang@ntu.edu.sg

Abstract. Novelty detection (ND) is a process for identifying information from an incoming stream of documents. Although there are many studies of ND on English language documents, however, to the best of our knowledge, none has been reported on Malay documents. This issue is important because there are many documents with a mixture of both English and Malay languages. This paper examines multilingual sentence-level ND in English and Malay documents using TREC 2003 and TREC 2004 Novelty Track data. We describe the text processing for multilingual ND, which consists of language translation, stop words removal, automatic stemming, and novel sentence detection. We compare the results for sentence-level ND on English and Malay documents and find that the results are fairly similar. Therefore, after preprocessing is performed on Malay documents, our ND algorithm appears to be robust in detecting novel sentences, and can possibly be extended to other alphabet-based languages.

Keywords: Novelty detection, sentence novelty, multilingual, automatic stemming, stop words removal, Malay.

1 Introduction

Nowadays, it becomes more and more convenient to find relevant information regarding a certain topic. With the development of Internet and information technology, lots of information sources are now available such as news articles, scientific papers, blogs, forums, etc. The bloom of information brings us rich useful information as well as many redundant information. People who are interested in a certain topic may read similar or even duplicate information over and over again. To solve this problem, a novelty detection system can be designed to retrieve only novel information regarding a certain topic. Novel information refers to documents or sentences that contain fresh content, and novelty detection (ND) is the process of singling out novel information from a given set of text documents. An automatic ND system consists of three main modules: (i) preprocessing, (ii) categorization, and (iii) novelty detection. The first module preprocesses the text documents by removing stop words and performing automatic stemming. Categorization classifies each incoming document or sentence

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 40–51, 2009.
© Springer-Verlag Berlin Heidelberg 2009

into its relevant topic bin. Then, within each topic bin containing a group of relevant documents or sentences, the ND module searches through the time sequence of documents or sentences and retrieves only those with "novel" information.

The pioneer work for ND was contributed by [16]. This paper gave the definition of "novelty" which was the opposite to "redundancy". A document that was less similar to its history documents was regarded as more "novel". [16] introduced ND at the document level. To serve users better, it could be more helpful to further highlight novel information at the sentence level [2]. Later studies focused on sentence-level ND, such as those reported in TREC 2002-2004 Novelty Tracks [12][14][13], which compared various novelty metrics [2][17], and integrated various natural language techniques [15][9][6]. This paper also focuses on sentence-level ND, which is the Task 2 of the TREC 2003/2004 Novelty Track. In this task, one needs to identify all novel sentences from groups of relevant sentences in all documents.

Although previous ND studies have been conducted on the English language, to best of our knowledge, no ND studies have been reported for Malay. This issue is important because there are many documents with a mixture of both English and Malay languages. The Malay language is spoken by more than 20 million people residing in the Malay Peninsula, southern Thailand, Philippines, Singapore, and Indonesia [7]. If a ND algorithm is tuned for the English language, is it necessary to develop a different ND algorithm to process documents of other languages such as Malay? This paper attempts to answer this question by investigating multilingual ND on both English and Malay documents.

If the documents were written in English, preprocessing steps are needed before applying the ND algorithm on Malay documents. Preprocessing steps includes language translation to Malay, Malay stop words removal, and Malay word stemming. Each of these steps will be discussed in detail in this paper.

For this paper, a list of Malay stop words is proposed and an adapted Malay stemming algorithm is introduced. After employing the Malay stop words and integrating the Malay stemming algorithm with the existing system, our ND system, which was originally developed for English sentences, can work well in detecting novel sentences in Malay.

This paper is organized as follows. A novelty detection algorithm for the English language is described in Section 2. A Malay ND algorithm, including Malay stop words and Malay stemming algorithm is presented and explained in Section 3. In Section 4, experiments and results are presented and discussed. Section 5 concludes the paper.

2 Novelty Detection in the English Language

Novelty detection in English involves the following steps:

2.1 Given all relevant sentences in all documents, first all stop words are removed. Stop words are words that need to be filtered out before doing the ND process as it affects the novelty prediction. Words like 'a', 'an', 'the', 'before', etc are considered stop words.

2.2 Next, remaining words are stemmed using Porter stemming algorithm [10]. Stemming [4], by definition, is a process of reducing the inflected (or sometimes derived) words to their root forms. Words like 'gives' and 'given' become 'give' after undergone the stemming process.

2.3 Finally, the novelty of each incoming sentence is determined using cosine similarity. The cosine similarity novelty metric calculates the similarities between the current sentence s_t and each of its history sentences s_i ($1 \leq i \leq t-1$), which determines the novelty score (S) for s_t as shown in Eq. (1).

$$S_{cos}(s_t) = \min_{1 \leq i \leq t-1} S_{cos}(s_t, s_i) \tag{1}$$

$$S_{cos}(s_t, s_i) = -\frac{\sum_{k=1}^{n} w_k(s_t) \cdot w_k(s_i)}{\|s_t\| \cdot \|s_i\|}$$

where $S_{cos}(s)$ denotes the cosine novelty score of sentence s and $w_k(s)$ is the weight of k^{th} element in sentence weighted vector s. The weighting function used in our work is tf.isf (term frequency multiply inverse sentence frequency) weighting function as defined below.

$$w_k(s_i) = tf_{w_k,s_i} log\left(\frac{n+1}{sf_{w_k} + 0.5}\right) \tag{2}$$

tf_{w_k,s_i} is the frequency of the word w_k in sentence s_i; sf_{w_k} is the number of sentences, in which the word w_k appears in the collection; n is the number of sentences in the collection.

The summary of novelty detection process in English is shown in Fig. 1.

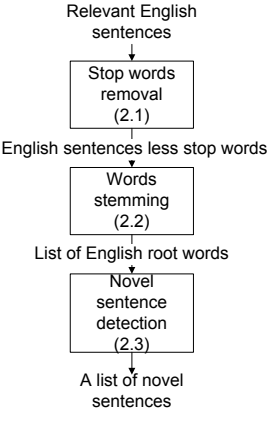


Fig. 1. English Novelty Detection Process

3 Novelty Detection in the Malay Language

"Is it necessary to develop a new ND algorithm for non-English documents?"

This paper addresses the question by adapting the existing ND algorithm on non-English documents, such as Malay. To tackle this issue, all sentences in English documents are first translated into Malay. Then, stop words are removed and the remaining words are stemmed. Lastly, novelty detection is carried out to retrieve all novel sentences. The overall process of ND in Malay can be seen in Fig. 2.

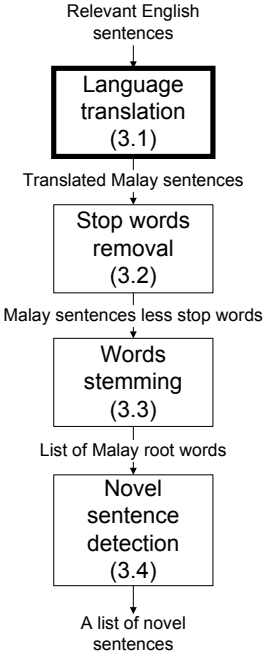


Fig. 2. Novelty Detection Process in Malay

3.1 Language Translation

In this first step, all English sentences are translated to Malay. During this process, we investigated issues in machine (automatic) vs. manual translation. Testing was conducted using TREC 2003 Novelty Track data on topics N1, N2, and N3. These topics were translated using two methods. First, we used only machine translation EBMT (Example-Based Machine Translation) [5]. Next, we manually corrected the machine translation. The F-score, precision and recall were calculated for each of 3 topics, and averaged across the topics. We compared the ND results of the two translations, which are shown in Table 1.

$$\text{Precision} = \frac{M}{S} \qquad \text{Recall} = \frac{M}{A} \tag{3}$$

Table 1. Comparison between machine translation and corrected translation

	Machine Translation	Corrected Translation	% difference
Average Precision	0.7599	0.7557	0.5527
Average Recall	0.9542	0.9564	0.2306
Average F-Score	0.8348	0.8330	0.2156

where S is the number of novel sentences selected by the novelty detection system; A is the number of novel sentences selected by the NIST assessor; and M is the number of matched novel sentences (number of novel sentences selected by both novelty detection system and NIST assessor). In TREC Novelty Track, the novel sentences marked by the NIST assessor is regarded as the truth.

$$F\text{-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

where F-Score represents the harmonic mean of precision and recall.

As shown in Table 1, the average results differ only slightly (<1%) between machine and corrected translations. Since the results differed only slightly, we used machine translation for the remaining documents.

3.2 Stop Words Removal

Since stop words are language dependent, different languages have different set of stop words. For this paper, a list of Malay stop words was generated. Some stop words were directly translated from English stop words, such as 'each' becomes 'setiap', 'after' becomes 'sesudah' and 'therefore' becomes 'maka'. Some others were gathered from Malay documents, such as 'ayuh', 'alamak', and 'amboi'. A list of 216 Malay stop words used in this paper is shown in Table 2.

3.3 Malay Word Stemming

Similar to stop words removal, word stemming is also language dependent. Therefore, each language has its own rules for stemming. In this section, a Malay word stemming algorithm is introduced.

Algorithm

The Malay stemming algorithm used in this paper is adapted from Abdullah [1] and Mangalam [8]. After some modifications, the flowchart is shown in Fig. 3.

In this flowchart, the difference between backward and forward stemming is the affixes (either prefixes or suffixes) which are stemmed first. In the backward stemming, suffixes are stemmed first, while in the forward stemming prefixes are stemmed first.

Table 2. List of Malay stop words

kah	arkian	di	kemudian	nun	semua
lah	atau	dia	kenapa	oh	seperti
tah	au	ee	kendatipun	oleh	serba
acap	auh	eh	kepada	oleh itu	serta
acap kali	ayuh	ehem	kerana	pabila	sesungguhnya
adakala	ayuhai	enggan	ketika	pada	setelah
adakalanya	bagai	entah	kian	padahal	setiap
adalah	bagaimana	entahkan	kita	paling	sewaktu
adapun	bagaimanapun	gamaknya	lagi	para	siapa
adoi	bagi	ha	lagikan	pelbagai	sila
aduh	bahawa	haah	laksana	perlu	sini
aduhai	bahawasanya	hanya	laksana	pernah	situ
agak	bahkan	harapnya	lalu	pun	sudah
agaknya	banyak	harus	macam	sahaja	sungguh
agar	barangkali	hatta	maha	saja	sungguhpun
ah	beberapa	hei	mahu	sambil	supaya
ahai	belum	helo	mahupun	sampai	syabas
ahem	benar	hendak	maka	sangat	syahadan
ai	berapa	hingga	malah	saya	tatkala
aja	betul	ia	malahan	sebab	telah
akan	bila	ialah	mana	sebagai	tentang
alah	bilamana	ini	manakala	sebagaimana	terhadap
alahai	boleh	itu	manalagi	sebermula	terlalu
alamak	boleh jadi	jangan	masih	sedang	tetapi
alhasil	buat	jemput	masing-masing	sedikit	tiap-tiap
alkisah	bukan	jika	memang	segala	tidak
amat	celaka	jikalau	mengapa	sejak	tolong
amboi	cis	jua	meskipun	sekali	umpama
andai	dalam	juga	mesti	sekali peristiwa	umpama
andai kata	dan	kadang	minta	sekalian	untuk
aneka	dapat	kadangkala	misal	sekiranya	usah
antara	dari	kalakian	mungkin	selalu	wah
apa	darihal	kalau	nampaknya	seluruh	wahai
apabila	daripada	kami	namun	semasa	walau
apakala	demi	ke	nan	sementara	walaupun
apalagi	dengan	kecuali	nian	semoga	yang

Malay Affixes

Mangalam [8] points out the affixes in Malay that need to be stemmed:
Prefixes: me-, pe-, be-, ke-, se-, te-, di-
Suffixes: -wati, -wan, -man, -nya, -lah, -kah, -kan, -pun, -ita, -ku, -mu, -an, -i

Rules for Stemming Malay Words

Ranaivo [11] explains the rules for stemming Malay words.

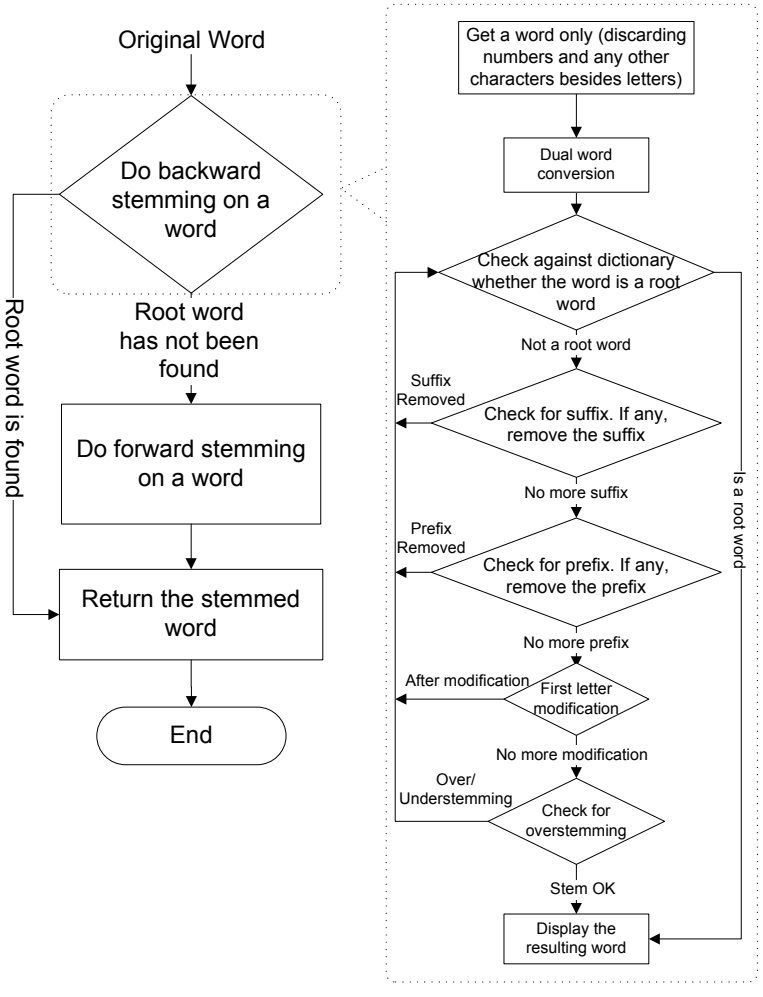


Fig. 3. Malay Word Stemming Flowchart with details on Backward Stemming Flowchart

Dual Word Conversion
In Malay, plural forms of words are represented by their dual word forms, such as 'anak - anak' (kids), 'murid - murid' (students), etc. In this step, all plural words are converted to their singular forms. For example, 'houses' is represented as 'rumah - rumah' in Malay. Having undergone this step, the word 'rumah - rumah' becomes 'rumah' (house), which is the singular form of 'rumah - rumah'.

Prefix
Rule 1: same prefix cannot occur more than once consecutively
If this happens; then, the second prefix must be part of the root word

Rule 2: after prefix te-, or pe-, no more prefixes are possible
Rule 3: prefix di- must appear first
Rule 4: prefix di- and me- can never coexist
Rule 5: prefix be- and te- can never coexist
Rule 6: prefix me- and be- can never coexist
Rule 7: prefix te- must appear after prefix me-
Rule 8: prefix pe- must appear after prefix me-
Rule 9: prefix be-, te-, me- and pe-
If the next two letters after truncating the above prefixes are
-ng, then remove 'ng'
-nC (C: consonant), then remove 'n'
-mC (C: consonant), then remove 'm'
-rC (C: consonant), then remove 'r'
-lV (V: vowel), then remove 'l'

Suffix
Rule 1: suffix -i and -kan can never coexist
Rule 2: suffix -wati, -wan, -man, and -ita can never coexist with any prefixes
Rule 3: suffix -kah and -kan can never coexist
Rule 4: suffix -lah and -kan can never coexist
Rule 5: suffix -an and -kan can never coexist
Rule 6: suffix -i and -an can never coexist

First Letter Modification
After taking out all prefixes and suffixes, each resulting word needs to be checked for its first letter.
Rule 1: if the first letter is 'm', then change it to 'p' or 'f'
Rule 2: if the first letter is 'n', then change it to 't'
Rule 3: if the first letter is 'y', then change it to 's'
Rule 4: if the first letter is vowel ('a', 'i', 'e', 'u', 'o'), then add 'k' at the beginning

Check for Overstemming
Rule 1: if the last prefix truncated is me- and first letter of the resulting word is 's', then check the word against dictionary by changing the first letter 's' with 'ny'. If the word is found in the dictionary, then overstemming has occurred.

Rule 2: if the last suffix truncated is -kan, then check the word against the dictionary by adding 'k' at the end. If by adding the letter 'k' results in a root word, then overstemming has occurred.

Drawback
One drawback of this algorithm is that the algorithm relies heavily on the dictionary to check for root words. Therefore, in order to improve the performance we need to compile a 'complete' list of Malay root words. Currently, the dictionary consists of 5300 Malay root words. Those words were taken from Bhanot's Malay - English dictionary [3].

At the end of stemming process, a list of root words for each sentence is generated.

3.4 Novel Sentence Detection

The last step of the ND process in Malay is novel sentence detection. This step is exactly the same as Section 2.3 of the ND process in the English language. After this step, a list of novel sentences is retrieved and the novelty detection process in Malay ends.

4 Experiments and Discussions

This paper used TREC 2003 and TREC 2004 Novelty Track data as the experimental data. TREC 2003 Novelty Track data consists of 50 topics (N1 - N50) and TREC 2004 Novelty Track data also consists of 50 topics (N51 - N100). The summary of these data is given in Table. 3.

Table 3. Data Statistics of TREC 2003 and TREC 2004 Novelty Track

Statistics	TREC 2003	TREC 2004
# topics	50 (N1-N50)	50 (N51-N100)
# relevant sentences	15557	8343
# novel sentences	10226	3454

In this experimental study, the focus was on novelty detection rather than relevant sentence categorization. Therefore, our experiments started with given all relevant sentences, from which the novel sentences were identified.

All sentences were originally written in English. We followed the novelty detection steps as described in Section 3 on the datasets to retrieve a list of Malay novel sentences. The cosine similarity novelty metric was employed and the threshold of novelty scores was predefined in a range from 0.05 to 0.95 with a step value of 0.10. The total of 40 runs were performed for both English and Malay on TREC 2003/2004 Novelty Track data. For each run (or each threshold), precision and recall (Eq. (3)) for each topic were calculated and the average precision and recall over 50 topics were plotted.

In information retrieval, precision-recall (PR) curves are always used to compare algorithms, where the algorithm with larger area under the curve is regarded as a better algorithm. Fig. 4 and Fig. 5 shows the PR curves of the English version vs. Malay version ND system on both TREC 2003 (Fig. 4) and TREC 2004 (Fig. 5) Novelty Track data. From these figures, we observe that the results of Malay ND is only slightly lower than English ND in the TREC 2003 Novelty Track data (with the average precision loss of 0.77% and the average recall loss of 0.96%), and is very similar in the TREC 2004 Novelty Track data (with the average precision loss of 0.09% and the average of recall loss of 0.16%). The

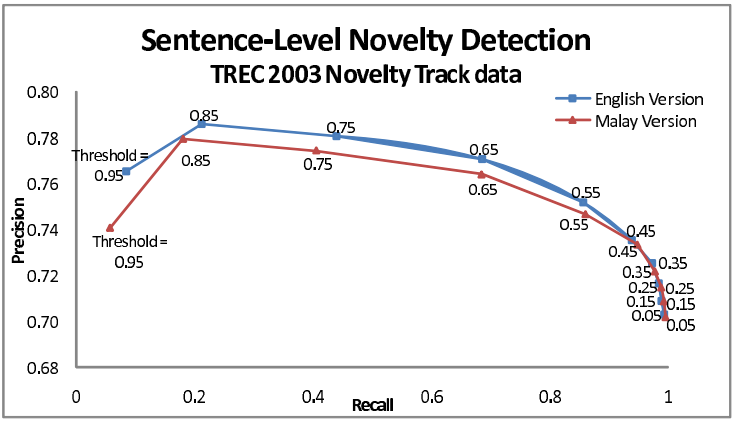


Fig. 4. Comparison between English and Malay version on TREC 2003 Novelty Track data

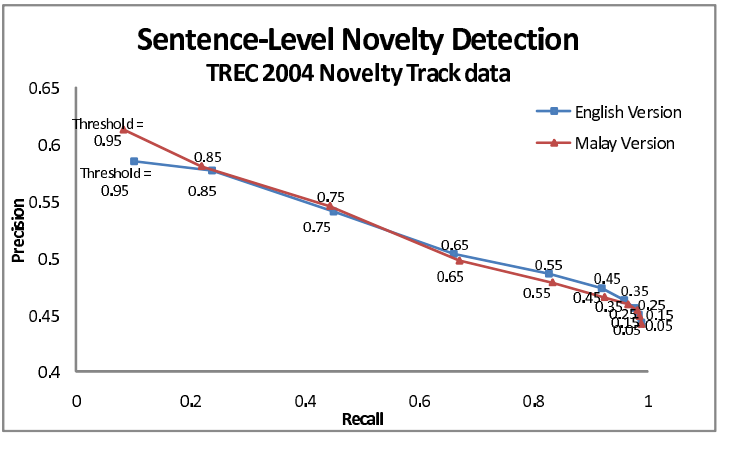


Fig. 5. Comparison between English and Malay version on TREC 2004 Novelty Track data

average precision and recall loss is the percentage difference of average precision and recall between English and Malay over all thresholds.

These experimental results indicate that, except for using different stop word lists and word stemming algorithms, the ND algorithm designed for the English language also performs well in detecting novel sentences in Malay.

5 Conclusion and Future Works

Sentence-level novelty detection (ND) is a process of retrieving novel sentences from a stream of incoming documents, based on a user's history and preferences.

In this paper, we focused on Task 2 of the TREC 2003/2004 Novelty Track to identify all novel sentences, given the relevant sentences in all documents. The document sets in the Novelty Track were all in the English language; thus, the corresponding ND algorithms were all designed for the English language. This paper addressed the issues involved in adapting existing ND algorithms for non-English languages such as Malay, which, to best of our knowledge, had not been previously studied.

Since the existing Novelty Track data is in English, in order to try the existing algorithms for documents in Malay, we first translated all relevant sentences using the EBMT (Example-Based Machine Translation) tool. Next, all Malay stop words from the translated sentences were removed, and the remaining words were converted into their basic forms using automatic Malay word stemming. After these preprocessing steps, the existing ND algorithm was performed on the resulting Malay sentences, and compared the results with original English sentences. The results show that by replacing English stop words with our proposed Malay stop words and English stemming algorithm with our adapted Malay stemming algorithm, existing ND algorithms that were originally developed for English can also be used for Malay sentences, without much loss in precision (<1%) and recall (<1%). Therefore, this paper shows that by replacing the language-specific processes (stop words and stemming), existing ND algorithms appear to be robust in detecting novel sentences in the Malay language, and can possibly be extended to other alphabet-based languages.

References

1. Abdullah, M.T., Ahmad, F., Mahmod, R., Sembok, T.M.T.: A stemming algorithm for Malay language. In: CITA, pp. 181–186 (2005)
2. Allan, J., Wade, C., Bolivar, A.: Retrieval and novelty detection at sentence level. In: SIGIR 2003, Toronto, Canada, pp. 314–321. ACM, New York (2003)
3. Bhanot, D.K.: The first online Malay — English dictionary (June 2008), http://dictionary.bhanot.net/
4. Datasegment.com online dictionary (August 2008), http://onlinedictionary.datasegment.com/word/stemming
5. U. R. Group. Example-Based Machine Translation (EBMT) prototype, Universiti Sains Malaysia (July 2008), http://utmk.cs.us.my:8080/ebmt-controller/index.jsp
6. Li, X., Croft, W.B.: An information-pattern-based approach to novelty detection. Information Processing and Management 44(3), 1159–1188 (2008)
7. Malay language. In: Wikipedia, The Free Encyclopedia (2008), http://en.wikipedia.org/wiki/Malay_language
8. Mangalam, S.S.V.: Malay-language stemmer. Sunway Academic Journal 3, 147–153 (2006)
9. Ng, K.W., Tsai, F.S., Goh, K.C., Chen, L.: Novelty detection for text documents using named entity recognition. In: 2007 6th International Conference on Information, Communications and Signal Processing, pp. 1–5 (2007)
10. Porter, M.F.: An algorithm for suffix stripping. In: Readings in information retrieval, pp. 313–316 (1997)

11. Ranaivo, B.M.: Computational analysis of affixed words in Malay language. In: 8th International Symposium on Malay/Indonesian Linguistics (ISMIL) (2004)
12. Robertson, S., Soboroff, I.: The TREC 2002 Filtering Track report. In: TREC 2002 - the 11th Text REtrieval Conference (2002)
13. Soboroff, I.: Overview of the TREC 2004 Novelty Track. In: TREC 2004 - the 13th Text REtrieval Conference (2004)
14. Soboroff, I., Harman, D.: Overview of the TREC 2003 Novelty Track. In: TREC 2003 - the 12th Text REtrieval Conference (2003)
15. Zhang, H.-P., Sun, J., Wang, B., Bai, S.: Computation on sentence semantic distance for novelty detection. Journal of Computer Science and Technology 20(3), 331–337 (2005)
16. Zhang, Y., Callan, J., Minka, T.: Novelty and redundancy detection in adaptive filter. In: SIGIR 2002, Tampere, Finland, pp. 81–88. ACM, New York (2002)
17. Zhao, L., Zheng, M., Ma, S.: The nature of novelty detection. Information Retrieval 9, 527–541 (2006)

Text Categorization Using Fuzzy Proximal SVM and Distributional Clustering of Words

Mani Arun Kumar and Madan Gopal

Control Group, Department of Electrical Engineering, Indian Institute of Technology Delhi, Hauz Khas, New Delhi 110016, India
sendtoarun@rediffmail.com

Abstract. Text Categorization (TC) remains as a potential application area for linear support vector machines (SVMs). Among the numerous linear SVM formulations, we bring forward linear PSVM together with recently proposed distributional clustering (DC) of words to realize its potential in TC realm. DC has been presented as an efficient alternative to conventionally used feature selection in TC. It has been shown that, DC together with linear SVM drastically brings down the dimensionality of text documents without any compromise in classification performance. In this paper we use linear PSVM and its extension Fuzzy PSVM (FPSVM) together with DC for TC. We present experimental results comparing PSVM/FPSVM with linear SVM^{light} and SVM_{lin} on popular WebKB text corpus. Through numerous experiments on subsets of WebKB, we reveal the merits of PSVM and FPSVM over other linear SVMs.

Keywords: Text categorization, support vector machines, proximal SVM, fuzzy proximal SVM, distributional clustering.

1 Introduction

Text categorization is the task of assigning a given text document to one or more predefined categories. It is gaining popularity due to the increased availability of documents in digital form and the following need to access them in flexible ways. A variety of statistical and machine learning techniques have been applied to automate the TC system in recent past [1]. Among the entire machine learning methods, linear SVM achieved impressive performance for TC tasks [2-3]. In its simplest form, a linear SVM is a hyperplane that separates a set of positive examples from negative examples with maximum margin. The approach is systematic and motivated by statistical learning theory (SLT) and Bayesian arguments.

Numerous linear SVM formulations and decomposition algorithms have been reported in literature. Among them linear PSVM proposed by Fung and Mangasarian [4] is interesting because of its simplicity and ease of implementation. It is easy to code and just requires only the solution of a single system of linear equations. PSVM is obtained by changing inequality constraints to equality constraints in conventional SVM. This change, though simple, changes the nature of the problem such that the two bounding hyperplanes of SVM become proximal planes around which the

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 52–61, 2009.
© Springer-Verlag Berlin Heidelberg 2009

datapoints of corresponding class are clustered. The Lagrangian function for the optimization problem is constructed and solved with KKT conditions to get the solution of PSVM. Using Sherman-Morrison-Woodbury (SMW) formula, Fung and Mangasarian [4] showed that the solution of linear PSVM can be obtained by just finding an inverse of size equal to the dimension of the classification problem.

While critics argue linear PSVM may not be suitable for TC because of inherent high dimensionality of text documents [5], we bring forward linear PSVM together with recently proposed distributional clustering (DC) of words [6] to realize its potential in TC realm. DC is a word clustering technique that drastically brings down the dimensionality of text documents. Bekkerman et al. [7] showed improved performance on 20newsgroups corpus by just using 300 word clusters over using 15,000 word features selected using mutual information (MI) feature selection measure. They used word clustering in a more general Information Bottleneck (IB) framework [7]. Thus word clusters greatly reduce the dimension of text documents without any significant compromise in classification performance. Hence, linear PSVM which can take advantage of this reduced dimension, is a ready choice for TC among other linear SVM formulations. In this paper, we present experimental results of linear PSVM and its fuzzy extension FPSVM proposed by Jayadeva et al. [8] with word clusters based document representation. We present experimental results comparing PSVM/FPSVM with linear SVM^{light} [9] and SVM$_{\text{lin}}$ [10] on popular WebKB [11] text corpus. Through numerous experiments on subsets of WebKB, we reveal the merits of PSVM/FPSVM over other linear SVMs, like fast training, independence of training time with respect to category for which training is done, and independence of training time with respect to penalty parameter of SVM. We study the classification performance of all these algorithms in terms of standard information retrieval metrics like F1 measure, break even point (BEP) and testing accuracy.

The rest of the paper is organized as follows: In Section 2 we describe text categorization preliminaries and document representation issues. Section 3 gives a brief overview of linear SVM, PSVM and FPSVM. Experimental results are explained and discussed in Section 4. Finally Section 5 discusses the conclusion and future work.

Notations used in this paper: All vectors will be column vectors unless transformed to a row vector by a prime $'$. A column vector of ones in real space of arbitrary dimension will be denoted by e. A column vector of zeros in a real space of arbitrary dimension will be denoted by $0e$. For a matrix $A \in \Re^{l \times n}$, A_i is the i^{th} row of A which is a row vector in $\Re^n$. For a vector $x \in \Re^n$, x_* denotes the vector in $\Re^n$ with components $(x_*)_i = 1$ if $x_i > 0$ and 0 otherwise, $i = 1,...,n$. In other words x_* is the result of applying step function component-wise to x.

2 Text Categorization Preliminaries

Automatic TC is supervised learning problem which involves training a classifier with some labeled documents and then using the classifier to predict the labels of unlabeled documents. Each document may belong to multiple labels or single label or no label at

all. Hence a binary classifier is learned for each label or category to form a complete text categorization system. During testing phase, each binary classifier will classify whether an incoming document belongs to its category or not. Below we will discuss the document representation issues.

2.1 Document Representation

Documents which typically are string of characters, have to be transformed into a representation suitable for the learning algorithm of the classifier. This transformation involves several steps like preprocessing, dimensionality reduction, feature selection/extraction and term weighting. We used '*bag of words*' representation in all our experiments. We removed stop words using a stop word dictionary [12] consisting of 319 words to reduce the dimension. In addition to stop words, we removed words that occurred in only one document of the text corpus. We also performed word stemming using Porter stemmer algorithm [13]. After stemming and stop word removal, we obtained document representation using IB based distributional clustering of words [7], while conventional way is to obtain significant word features using feature selection measures like document frequency (DF), MI etc… IB based DC of words has been used in [7], and is suggested as an effective alternative to traditional feature selection measures. We used publicly available distributional clustering implementation with default parameter setting [14]. Following [7], after generating clusters we used term weighting based on word cluster count to represent each document. All these document representation steps were performed using MATLAB 7.3.0 (R2006b) [15] software.

3 Linear SVM Formulations

SVMs represent novel learning techniques that have been introduced in the framework of Structural Risk Minimization (SRM) and in the theory of VC bounds. In the simplest binary pattern recognition tasks, SVMs use a linear separating hyperplane to create a classifier with a maximal margin. Consider the problem of binary classification wherein a linearly inseparable dataset A of l datapoints in real n-dimensional space of features is represented by a matrix $A \in \Re^{l \times n}$. The corresponding target or class of each datapoint A_i, $i = 1, 2, ..., l$ is represented by a diagonal matrix $D \in \Re^{l \times l}$ with entries D_{ii} as $+1$ or -1. Given the above problem, SVM's linear soft margin problem is to solve the following primal Quadratic Programming Problem (QPP):

$$\begin{aligned} &\text{Minimize} \quad \frac{1}{2} w'w + Ce'y \\ &\text{such that } D(Aw - eb) + y \geq e, \; y \geq 0e \end{aligned} \tag{1}$$

where C is the penalty parameter and y is a vector of nonnegative slack variables. The optimal separating hyperplane can be expressed as

$$d(x) = w'x - b, \tag{2}$$

where $x \in \Re^n$ is any (new) datapoint, $w \in \Re^n$ and $b \in \Re$ are parameters. Since the number of constraints in (1) is large, the dual of (1) is usually solved. The decision boundary described as $d(x) = 0$ lies midway between the two bounding hyperplanes given by:

$$w'x - b = 1 \text{ and } w'x - b = -1, \tag{3}$$

and separates the two classes from each other with a margin of $\frac{2}{\|w\|_2}$. A datapoint is classified as class $+1$ or -1 according to whether the decision function $(d(x))_*$ yields 1 or 0 respectively. An important characteristic of SVM is that it can be extended in a relatively straightforward manner to create nonlinear decision boundaries [16]. However, we will restrict our discussion to linear SVM as it has been shown that linear SVM performs better than nonlinear SVM in TC [3]. Conventional training algorithms for SVMs are complex and slow. Standard quadratic optimizers cannot easily solve the immense size of dual of the QPP (1) that arises from SVMs. Many decomposition methods and new formulations have been proposed to address this issue. As a representative from the family of such decomposition methods, we will use SVM^{light} [9] in our experiments for comparison. Further, special SVM formulations and algorithms exclusively for applications like TC that work with linear kernel have been proposed; As a representative we will use SVM_{lin} [10] algorithm that uses modified Newton method along with conjugate gradient solver for fast training of linear SVMs.

3.1 Linear Proximal SVM

PSVM [4] does binary classification by assigning data depending on its proximity to the two parallel hyperplanes that are pushed apart as far as possible. PSVM uses the modified primal QPP given by:

$$\begin{aligned} &\text{Minimize } \frac{1}{2}\left(w'w + b^2\right) + \frac{C}{2} y'y \\ &\text{such that } D(Aw - eb) + y = e \end{aligned} \tag{4}$$

The difference between primal QPPs (1) and (4) is that, QPP (4) minimizes the square of 2-norm of slack variables as opposed to 1-norm in QPP (1) and measures the margin in $(n+1)$-dimensional space of $(w,b) \in \Re^{n+1}$. Also the inequality constraints of QPP (1) are changed to equality constraints in QPP (4). Note that the constraint $y \geq 0e$ doesn't appear in (4) as the inclusion of $\|y\|_2^2$ makes it redundant. This simple change completely changes the nature of the formulation so that the bounding planes (3) now become proximal planes around which the datapoints of each class are clustered. By constructing the Lagrangian function and solving for its KKT conditions, Fung and Mangasarian [4] showed the solution of (4) to be:

$$w = A'D\alpha \text{ and } b = -e'D\alpha$$
$$\text{where } \alpha = C\left(I - H\left(\frac{I}{C} + H'H \right)^{-1} H' \right) e,\ H = D[A \quad -e] \tag{5}$$

They used the popular Sherman-Morrison-Woodbury (SMW) formula to obtain this result. Thus PSVM algorithm is extremely simple and it needs only the solution of a single system of linear equations. Even a classification problem with millions of datapoints can be solved by just inverting a matrix of the order of input space dimension (plus one). In general, machine learning problems have input space dimension typically of the order 100 or less. However text documents will have very high dimension of the order of 10,000 or more. Because of this reason, TC researchers argue that PSVM may not be suitable for TC as it will need inversion of a very large matrix [5]. It is very true when we work with text documents without any dimensionality reduction. But, dimensionality reduction has always been an inherent constituent in text categorization; in fact researchers argue that dimensionality reduction often improves the classification performance. Hence, we take advantage of the recently proposed DC to achieve drastic reduction in dimension of the text documents, which will also aid PSVM. We used publicly available PSVM implementation [17] in all our experiments.

3.2 Linear Fuzzy Proximal SVM

Often in real-world classification problems, some datapoints are more important than other datapoints. That is, datapoints no more exactly belong to one or more classes and can be treated as if there is uncertainity with regard to the class to which they belong. Therefore a fuzzy membership, $0 \le s_i \le 1$, can be associated with each datapoint A_i, indicating the extent to which the datapoint may be said to belong to its target class. In linear FPSVM [8], these membership values are used to assign relatively different penalty parameter for slack variables, instead of fixed penalty parameter in PSVM. FPSVM is obtained by solving the following optimization problem.

$$\begin{aligned} &\text{Minimize } \frac{1}{2}\left(w'w + b^2 \right) + \frac{C}{2}\|Y\|^2 \\ &\text{such that } \quad D\left(Aw - eb \right) + y = e \end{aligned} \tag{6}$$

where $Y = Sy$ and S denotes a diagonal matrix, $S = diag\left(s_1, s_2, ..., s_l \right)$, whose diagonal entries are membership values of datapoints. Using SMW formula the solution of FPSVM has been shown to be:

$$w = A'DS\alpha \text{ and } b = -e'DS\alpha$$
$$\text{where } \alpha = C\left(I - SH\left(\frac{I}{C} + H'S'SH \right)^{-1} (SH)' \right) Se,\ H = D[A \quad -e] \tag{7}$$

Thus the solution of FPSVM can also be obtained by solving a system of simultaneous linear equations similar to PSVM. We modified the PSVM implementation available at [8] to FPSVM and used in all our experiments.

4 Experimental Results and Discussion

We performed all our experiments on WebKB corpus. Also called as four universities dataset, WebKB is a collection of Web pages from large number of university computer science departments [11]. The corpus contains 8,282 Web pages, manually classified into seven categories. Following previous studies [18], we used top four categories: course, faculty, project, and student, in all our experiments. This led us to a corpus with 4,199 documents and 11,521 word features. We generated 200 word clusters for document representation. It is to be noted that, all document representation steps like, preprocessing, dimensionality reduction and word clustering were done on the entire dataset before splitting them into training/testing sets. We created training and testing splits only during classification phase. We created many subsets of WebKB to carry out our experiments. A training set of size X means it will contain randomly selected X documents per category in training set and the remaining documents will be used for testing. We compared four linear SVM algorithms SVM^{light}, SVM$_{\text{lin}}$, PSVM and FPSVM in terms of computing time and standard information retrieval metrics F1 measure/BEP/accuracy measured on testing set. We calculated these performance measures separately for each of the four categories and then averaged (micro/macro) them to get final performance measures. As far as computing time is concerned we report total computing time over all four categories. For both SVM^{light} and SVM$_{\text{lin}}$ the tolerance of stopping criteria was set to default value of 0.001. We used publicly available implementations of both these algorithms for our experiments. Default values for all parameters of these implementations were used unless otherwise stated. We conducted a grid search for best penalty parameter C in the range $\left(2^{-13}, 2^{11}\right)$ to maximize micro BEP of the test set. For FPSVM, following [8], we present experimental results for a simple case wherein datapoints belonging to class $+1$ are assigned a membership value s^+ and datapoints belonging to class -1 are assigned a membership value s^-. For each category, we maximized micro BEP of FPSVM using best values of s^+ and s^- obtained through search in the range 0.1 to 1 in steps of 0.1. For all implementations, after training SVM for a category, we fit a two parameter sigmoid trained with regularized binomial maximum likelihood [19], so that SVM outputs posterior probabilities. Using these posterior probabilities, a threshold is learnt for each category to maximize micro BEP. All experiments were implemented using MATLAB 7.3.0 (R2006b) [15] software on a PC with Intel Core2Duo processor (2.13GHz), 1GB RAM.

Table 1 shows the micro BEP performance of all four algorithms on numerous subsets of WebKB corpus. Table 2 shows testing accuracy of all four algorithms on the same subsets. All these experimental results are average of three random trials on each subset. These two tables show that, PSVM performs better than SVM^{light} and is comparable with SVM$_{\text{lin}}$. However, FPSVM emerges as the best performer among all algorithms in terms of both micro BEP and testing accuracy. These observations are also reflected in other performance measures like, micro F1, macro F1, and macro BEP as shown in Table 3. Figure 1 shows the variation of total training time of all four

Table 1. Micro BEP performance of SVM^{light}, SVM_{lin}, PSVM and FPSVM on different subsets of WebKB

Training set	SVM^{light}	SVM_{lin}	PSVM	FPSVM
10	66.89	67.63	66.96	67.72
20	71.57	72.60	73.77	74.30
30	73.37	74.97	75.02	76.33
40	77.01	78.69	78.66	79.46
50	76.90	77.96	78.31	78.85
60	77.40	78.66	78.30	78.98
70	79.15	80.59	80.46	80.93
80	80.08	81.42	81.16	81.48
90	79.51	80.51	80.74	81.32
100	80.74	81.80	81.93	82.27
125	81.35	82.56	82.39	82.79
150	81.89	82.57	82.50	82.96
175	82.96	83.78	83.22	83.74
200	82.88	83.99	83.81	84.17
250	83.71	84.65	84.30	84.75
300	84.41	85.09	84.79	85.37
Average	**78.74**	**79.84**	**79.77**	**80.34**

Table 2. Testing accuracy of SVM^{light}, SVM_{lin}, PSVM and FPSVM on different subsets of WebKB

Training set	SVM^{light}	SVM_{lin}	PSVM	FPSVM
10	69.44	74.32	71.67	73.85
20	79.86	83.63	84.71	84.70
30	83.85	85.38	85.36	86.38
40	87.47	88.83	88.79	89.16
50	87.28	87.67	88.23	88.50
60	87.51	89.02	88.11	88.92
70	88.76	90.06	89.78	90.17
80	89.68	90.72	90.53	90.59
90	88.96	89.64	89.92	90.42
100	90.06	90.80	90.90	91.00
125	90.30	91.25	90.97	91.14
150	90.84	91.18	91.09	91.36
175	91.11	91.86	91.40	91.61
200	91.21	91.92	91.86	92.01
250	91.64	92.32	92.11	92.32
300	91.90	92.53	92.30	92.58
Average	**87.49**	**88.82**	**88.61**	**89.05**

Table 3. Average results of macro BEP, micro F1 and macro F1 of SVM^{light}, SVM$_{lin}$, PSVM and FPSVM on different subsets of WebKB

	Macro BEP	Micro F1	Macro F1
SVM^{light}	**77.55**	**77.86**	**76.26**
SVM$_{lin}$	**78.76**	**79.34**	**77.90**
PSVM	**78.65**	**79.21**	**77.75**
FPSVM	**79.26**	**79.84**	**78.45**



Fig. 1. Training time variation of SVM^{light}, SVM$_{lin}$, PSVM and FPSVM on different subsets of WebKB

algorithms with respect to the training size used. Again these total training times are average over three random trials in each training size. It could be observed that, the training times of PSVM and FPSVM are the least and the growth rate is not as fast as SVM^{light}/SVM$_{lin}$. This is because, in PSVM and FPSVM, the training times are dominated by dimensionality of the corpus with which we are working, thus making them ideal choice to combine with powerful dimensionality reduction methods. Figure 2 shows how these algorithms scale with respect to penalty parameter C. These results are average of three trials with training size 300. It could be observed that the training time shoots up drastically for higher values of C in SVM^{light} and slightly for SVM$_{lin}$; whereas the training time of PSVM/FPSVM are constant and are independent of penalty parameter as dimension size remains fixed. In fact we observed on large unbalanced datasets that the training time of PSVM/FPSVM are independent of the category for which training is done as dimension size remains fixed.

Though fuzzy extensions are possible for other versions of SVMs as well, in view of fast training and constant training time with respect to penalty parameter variations/category, FPSVM seems to be more attractive.

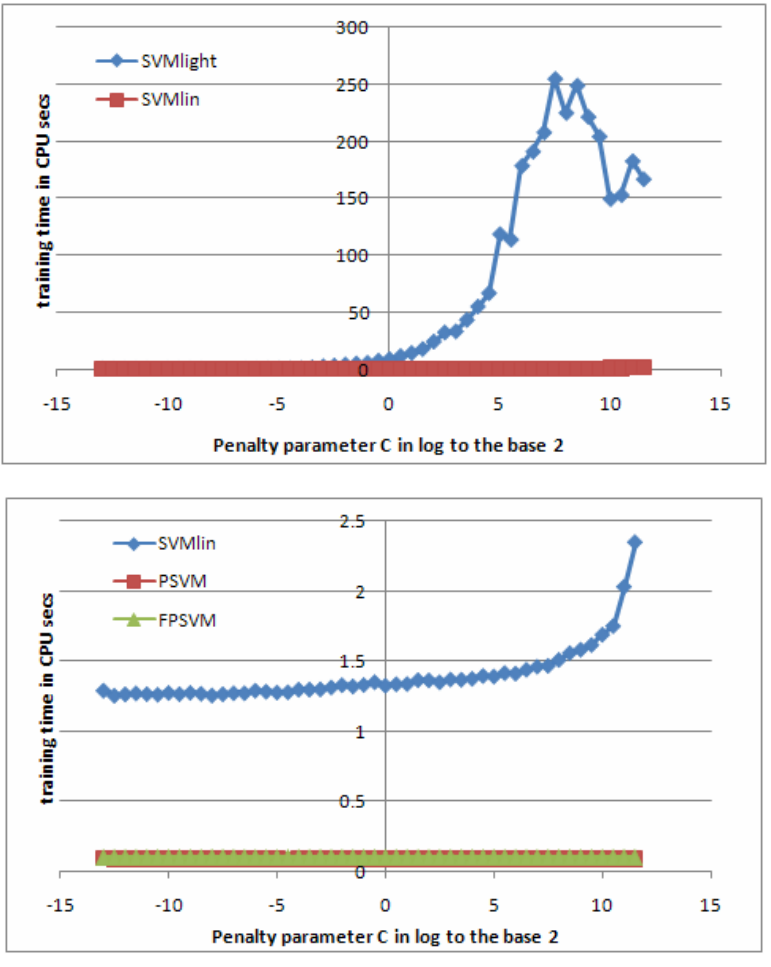


Fig. 2. Training time vs. penalty parameter C comparison of SVM^{light} and SVM_{lin} (top), SVM_{lin}, PSVM and FPSVM (bottom) on WebKB with training size 300

5 Conclusion

In this paper we have investigated PSVM/FPSVM algorithms with distributional clustering of words for text categorization. Distributional clustering of words, being a powerful dimensionality reduction technique, drastically brings down the dimension of text documents to work with. PSVM/FPSVM algorithms take advantage of this reduced dimension leading to fast training with competitive results. The training time of PSVM/FPSVM is independent of the penalty parameter or the category for which training is done. We have compared PSVM/FPSVM with popular SVM^{light} and SVM_{lin} algorithms on numerous subsets of WebKB corpus. While PSVM gives comparable results, FPSVM outperforms all other algorithms in terms of BEP, F1 measure and testing accuracy. Thus our experiments demonstrate that, FPSVM

combined with distributional clustering of words is a better choice for text categorization. Our future work is to explore the use of rough sets based dimensionality reduction techniques along with PSVM/FPSVM in text categorization.

References

1. Yang, Y.: An evaluation of statistical approaches to text categorization. Technical Report CMU-CS-97-127, Carnegie Mellon University (1997)
2. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)
3. Dumais, S., Platt, J., Heckerman, D., Sahami, M.: Inductive learning algorithms and representations for text categorization. In: Proc. of Seventh International Conf. on Information and Knowledge Management, pp. 148–155 (1998)
4. Fung, G., Mangasarian, O.L.: Proximal support vector machine classifiers. In: Proc. Of Seventh International Conf. on Knowledge and Data Discovery, pp. 77–86 (2001)
5. Joachims, T.: Training linear SVMs in linear time. In: Proc. of the ACM conf. on Knowledge Discovery and Data Mining, pp. 217–226 (2006)
6. Baker, L.D., McCallum, A.K.: Distributional Clustering of words for Text Classification. In: Proc. of (SIGIR) 1998, 21st International Conf. on Research and Development in Information Retrieval, pp. 96–103 (1998)
7. Bekkerman, R., El-Yaniv, R., Tishby, N., Winter, Y.: Distributional word clusters vs words for text categorization. Journal of Machine Learning Research 3, 1183–1208 (2003)
8. Jayadeva, Khemchandani, R., Chandra, S.: Fast and robust learning through fuzzy linear proximal support vector machines. Neurocomputing 61, 401–411 (2004)
9. Joachims, T.: Making large-scale SVM learning practical. In: Advances in kernel methods – support vector learning. MIT Press, Cambridge (1998)
10. Sindhwani, V., Keerthi, S.S.: Newton methods for fast solution of semi-supervised linear SVMs. In: Large Scale Kernel Machines, pp. 155–174. MIT Press, Cambridge (2005)
11. Craven, M., DiPasquo, D., Freitag, D., McCallum, A.K., Mitchell, T.M., Nigam, K., Slattery, S.: Learning to extract symbolic knowledge from the World Wide Web. In: Proc. of National Conf. Artificial Intelligence (AAAI 1998) (1998)
12. Stop words list, `http://www.dcs.gla.ac.uk/idom/ir_resources/linguisticutils/`
13. Porter, M.: An algorithm for suffix stripping. Program (Automated Library and Information Systems) 14(3), 130–137 (1980)
14. DC of words software, `http://www.cs.technion.ac.il/~ronb/thesis.html`
15. MATLAB (2008), `http://www.mathworks.com`
16. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)
17. PSVM codes, `http://www.cs.wisc.edu/dmi/svm/psvm/`
18. Al-Mubaid, H., Umair, S.A.: A new text categorization technique using distributional clustering and learning logic. IEEE Transactions on Knowledge and Data Engineering 18(9), 1156–1165 (2006)
19. Platt, J.C.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Advances in Large Margin Classifiers, pp. 61–74. MIT Press, Cambridge (1999)

Cool Blog Classification from Positive and Unlabeled Examples

Kritsada Sriphaew*, Hiroya Takamura, and Manabu Okumura

Precision and Intelligence Laboratory
Tokyo Institute of Technology
4259 Nagatsuta Midori-ku Yokohama 226-8503 Japan
kong@lr.pi.titech.ac.jp,{takamura,oku}@pi.titech.ac.jp
http://www.lr.pi.titech.ac.jp

Abstract. We address the problem of cool blog classification using only positive and unlabeled examples. We propose an algorithm, called PUB, that exploits the information of unlabeled data together with the positive examples to predict whether the unseen blogs are cool or not. The algorithm uses the weighting technique to assign a weight to each unlabeled example which is assumed to be negative in the training set, and the bagging technique to obtain several weak classifiers, each of which is learned on a small training set generated by randomly sampling some positive examples and some unlabeled examples, which are assumed to be negative. Each of the weak classifiers must achieve admissible performance measure evaluated based on the whole labeled positive examples or has the best performance measure within iteration limit. The majority voting function on all weak classifiers is employed to predict the class of a test instance. The experimental results show that PUB can correctly predict the classes of unseen blogs where this situation cannot be handled by the traditional learning from positive and negative examples. The results also show that PUB outperforms other algorithms for learning from positive and unlabeled examples in the task of cool blog classification.

Keywords: Cool blog, PU-learning, weighting examples, bagging.

1 Introduction

For text mining, blog is an interesting textual resource because of its unique characteristic and public availability. It is a recent form of online content that has become popular among the web users. The contents of each blog are frequently updated by the postings of blog entries in reverse chronological order. Nowadays, a large number of blogs are published over the blogosphere but only some of them are interesting from the viewpoint of the readers.

The problem to identify a set of blogs that can interest an individual reader from a huge amount of blogs is not trivial. In this work, we consider the interestingness of blogs in an aspect of their contents but not their design or appearance,

* The author is supported by Japan Soceity for the Promotion of Science (JSPS).

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 62–73, 2009.
© Springer-Verlag Berlin Heidelberg 2009

named coolness. Finding the cool blogs with interesting contents can be useful, but a more challenging task is how to overcome the real-life scenario when the labeled data is rare and only positive examples are preferred.

The criteria to judge the coolness of blogs are fairly subjective based on the readers' intuition. For example, the readers whose interests are gourmet may prefer those blogs that recommend good restaurants more than ones that relate to the modern fashions. Some of the readers may have interest in both kinds of blogs. We can simply represent the cool blog classification as a basic binary classification where we suppose that the reader is interested in one specific group of blogs and seeks to distinguish between the interesting instances (i.e., positive examples) and the other instances (i.e., negative examples). In the real world case, positive examples can be collected from the blogs that have been visited many times by the user, but this is not applicable for the negative examples. Therefore, it often happens that we lack the labeled negative examples although there are a large number of unlabeled examples which we can employ to learn the classifier. This scenario is called the learning from positive and unlabeled examples (*PU-learning*). Moreover, the labeled positive examples may not cover all target concepts that the reader is interested in. This effects the difficulty of predicting unseen instances if only a small set of labeled examples is available. Especially to the labeled cool blogs, it is highly possible that we have less number of positive examples. Those labeled examples are only a portion of positive concepts that can interest the reader. There are other target and non-target concepts available in the unlabeled examples which can be used to learn the classification model.

To overcome these problems, we propose a new algorithm for PU-learning that employs two techniques for learning. First, the weighting technique is used for assigning a weight to each unlabeled example and all of unlabeled examples are assumed to be negative in the training set. With the training set that contains positive and assumed negative examples, the bagging technique is applied to obtain a smaller set of training used for learning the classifier by random sampling with replacement from the training set. After several iterations, a number of classifiers will be generated. Each of them must achieve admissible performance measure evaluated based on the whole labeled positive examples or has the best performance measure within iteration limit, but it does not overfit to the training positive examples. This enables the generated classifiers to predict the classes of unseen instances. By aggregating these classifiers, the variance error from various training sets generated by the samplings can be lessen.

The contributions of this work are as follows. We are the first to address the problem of cool blog classification in the scenario of PU-learning that is prevalent in the real world application. We propose an algorithm for PU-learning which needs only positive and unlabeled examples as the inputs without separated validation sets. Although it is difficult to predict the class of unseen instances when only a small number of labeled examples is available, the proposed algorithm can perform well in such case by employing the bagging technique to randomly sample the data for training.

2 Related Work

We divide this section into two parts. The first part discusses the problem of cool blog classification. The second part reviews existing approaches of PU-learning.

The problem of cool blog classification was first introduced in [1]. They introduced some assumptions on the topics of blogs to indicate the coolness. A set of features corresponding to their assumptions was proposed for encoding both positive and negative examples and they are learned by traditional SVM. However, their scenario is absolutely different from our work since we deal with a more practical scenario where only positive and unlabeled examples are available for learning. Another related research on blog data is the credibility assessment of blogs [2,3]. They proposed a set of factors which can be used to indicate the credibility of blogs. The objective of credibility assessment is totally different from cool blog classification in that they want to retrieve a set of highly credible blog entries with regard to the specified query while our objective is to identify cool blogs as a whole without prerequisite information. Although there are some studies on the interestingness of web contents [4,5] in the information retrieval area, they are not equivalent to this problem because their targets are to find interesting web pages based on the information of reader's profile.

For the problem of PU-learning, several algorithms have been published after the first work in [6]. As stated in [7], the algorithms for PU-learning can be divided into three families. The first family of algorithms is a two-step strategy where the heuristics to identify unlabeled examples that are likely to be negative are applied in the first step, and the standard learning method learns the identified examples and positive examples in the second step [8,9,10,11].

The second family of algorithms focuses on estimating statistical queries over positive and unlabeled examples in order to construct the model of negative examples [12]. This idea was also extended to the Co-Training scenario for which the data has two or more redundant views that are compatible but conditionally independent [13]. The shortcoming of this family of algorithms is the prerequisite knowledge of the prior probability of the positive examples which are usually not available in practice.

The third family of algorithms concerns with assigning weights to the examples, assuming unlabeled examples as negative and applying standard learning method. Lee and Liu [14] applied the logistic regression to learn the linear function from the weighted examples. The positive examples are assigned a specific weight while the unlabeled examples, that are assumed to be negative, are assigned another weight. Those two weights are derived from a validation set by optimizing the model to handle noise rates of greater than a half. Liu et al. [9] presented the biased SVM that allows noise for both positive and negative examples in the SVM formulation by giving two different weights for positive and negative errors, respectively, via regularization parameters. Zhang and Lee [15] gives a specific weight to the positive examples in the classification function. The classification function is a simple sign function of the difference between the weighted probability of being positive and the probability of being unlabeled

for a given example, where a weight is estimated from a validation set and the probabilities are estimated based on tf-idf of the words in the training set.

The most recent successive approach was presented by Elkan and Noto in [16]. When the assumption that positive training examples are selected completely at random holds, they proved that a classifier trained on the positive and unlabeled examples predict probabilities that differ by only a constant factor from the true conditional probabilities of being positive. This assumption is coincident with a general situation when we prepare the data for cool blog classification. However, there are two major shortcomings of their approach. First, the high bias of the generated classifier that is trained from the whole positive and unlabeled examples may result to the overfit on the training set. Second, their method to duplicate the same unlabeled examples as both positive and negative will effect badly to the SVM.

Most of the existing algorithms for PU-learning have been evaluated by the modified versions of standard data sets, e.g., Reuters-21578 and 20Newsgroup. These modified datasets are generated by regarding some arbitrarily-selected examples as unlabeled. In contrast, our dataset for cool blog classification is not artificial in the sense that the unlabeled examples are not generated from the labeled ones. It may be the case that users would try to avoid annotating examples which are hard to make the decision on. In this work, we address the problem of PU-learning in this practical way using cool blog data and compare our approach with some existing PU-learning algorithms for performance study.

3 PUB: An Algorithm for PU-Learning Using Weighting and Bagging Techniques

In this work, we propose an algorithm that employs weighting and bagging technique for PU-learning, called PUB. The algorithm can be divided into two steps. In the first step, the weighting technique is used to assign individual weight to each unlabeled example. We follow the approach in [16]. In the second step, the bagging technique is applied to obtain several weak classifiers, each of which is learned on a small training set generated by randomly sampling some positive examples and some unlabeled examples, which are assumed to be negative. Each of the weak classifiers must achieve admissible performance measure evaluated based on the whole labeled positive examples or has the best performance measure within iteration limit. The majority voting function on all weak classifiers is employed to predict the class of a test instance.

3.1 Weighting Technique

Before giving an explanation about weighting technique, we first formally define the problem of cool blog classification using positive and unlabeled examples.

Let D denote a training set, $D = \{(x_n, y_n, s_n)\}$ where $n = 1, ..., N$. The x's are examples in the training set and y's are the class labels where $y_n = 0$ if x_n is a negative example and $y_n = 1$ if x_n is a positive example. At the beginning, y_n

is left unknown if x_n is an unlabeled example. The s's are other binary attributes of examples, where $s_n = 1$ if x_n is labeled and $s_n = 0$ if x_n is unlabeled. x, y and s can be viewed as random variables where there is some unknown probability distribution $p(x, y, s)$ over triples $< x, y, s >$. In our case, only positive examples are labeled. Therefore, no negative example is labeled, i.e., $p(s = 1|x, y = 0) = 0$.

Our goal is to learn a function $f(x)$ so that $f(x)$ becomes close to $p(y = 1|x)$ as much as possible, but we need an assumption about which positive examples are labeled. By following the same assumption as in [16], the labeled positive examples are chosen completely at random from all positive examples. Therefore, this assumption can be formally stated as $p(s = 1|x, y = 1) = p(s = 1|y = 1)$.

Suppose we want to learn a function $g(x)$ to classify whether a given document is labeled or not, such that $g(x) = p(s = 1|x)$. In [16], it is stated that we can obtain a goal function $f(x)$ from $g(x)$ by using the following lemma.

Lemma 1. *Suppose the "selected completely at random" assumption holds, then* $p(y = 1|x) = p(s = 1|x)/c$ *where* $c = p(s = 1|y = 1)$.

In other words, c is the constant probability that a positive example is labeled. In their work, c is estimated as $\frac{1}{n} \sum_{x \in P} g(x)$ where P is a set of labeled positive examples and n is the cardinality of P. This means we can calculate c from the probabilities of positive examples produced as output of the classifier $g(x)$. The proof can be found in [16].

In this scenario, it is obvious that $p(y = 1|x, s = 1) = 1$ for any labeled positive example, and $p(y = 1|x, s = 0)$ can be obtained from

$$p(y = 1|x, s = 0) = \frac{(1 - c)}{c} \frac{p(s = 1|x)}{1 - p(s = 1|x)}. \tag{1}$$

Here, $p(y = 1|x, s = 0)$ is the probability that an unlabeled example is positive, and we denote $w(x) = p(y = 1|x, s = 0)$. In other words, $1 - w(x)$ is the probability that an unlabeled example is negative. The original approach in [16] employs these probabilities by assigning different weights to examples for constructing a new training set. Each positive example is assigned with unit weight $p(y = 1|x, s = 1)$, while the unlabeled example x is duplicated; one copy is assumed to be positive with weight $w(x)$ and the other copy is assumed to be negative with weight $1 - w(x)$. This is the main shortcoming of the original approach since the duplication of examples with different labels effects to the failure of the generated classifier.

In this work, we propose to assign unit weight to the positive examples, and assume all unlabeled examples to be negative with their weights being $1 - w(x)$. These weighted examples are later used as a training set for the second step.

3.2 Bagging Technique

The original idea of bagging for classification is to generate m weak classifiers trained from different training sets that are sampled from the whole training set

uniformly and with replacement [17]. Although it is possible to directly apply the original idea for PU-learning, there are three differences between the scenario of the original bagging technique and PU-learning. First, all examples in the training set must be labeled in the original scenario of bagging approach. Second, the training set must have both positive and negative examples. Third, there is no guarantee that each weak classifier trained from the sampled training set will achieve high performance on the target concepts, especially for the case that only positive examples are labeled.

We can simply solve the first and second problems by assuming all unlabeled examples as either positive or negative. However, we still need to solve the third problem simultaneously since our assumed classes of some unlabeled examples may be incorrect and effect to the poor performance of the generated classifiers. A modified version of bagging, BaggTaming [18], can be applied to solve the third problem by setting a condition that each classifier should achieve admissible performance measure when it is evaluated based on the target concepts. However, such work has different scenario to our problem since the nature of data used in their approach is different from PU-learning and we do not have labeled negative examples. Moreover, the third problem still remains if the original algorithm is applied because not all generated classifiers achieve admissible performance measure. Therefore, we proposed a new bagging technique to solve those three problems as shown by the pseudo code in Figure 1.

1: Construct D' from D by assigning $y_n = 0$ if x_n is an unlabeled example
2: $\mathcal{W} = \emptyset$;
3: **while** $|\mathcal{W}| < m$ **do**
4: $i = 1$
5: $maxperf = 0$
6: **while** $i \leq$ LIMIT **do**
7: Generate D_i by random sampling with replacement from D' and balancing the number of examples in each class
8: Learn $f_i(x) = p(y = 1|x)$ from $x \in D_i$
9: **if** $eval_{x' \in P}(f_i(x')) > maxperf$ **then**
10: $f_{BEST} \leftarrow f_i(x)$
11: $maxperf = eval_{x' \in P}(f_i(x'))$
12: **end if**
13: **if** $eval_{x' \in P}(f_i(x')) > \alpha$ **then**
14: goto line 18
15: **end if**
16: $i \leftarrow i + 1$
17: **end while**
18: $\mathcal{W} = \mathcal{W} \cup f_{BEST}$
19: **end while**
20: $\hat{f}(x) = arg \max_{y \in \{-1,1\}} \sum_{f_j(x) \in \mathcal{W}} eval_{x' \in P}(f_j(x')) \times I(y == f_j(x))$

Fig. 1. Pseudo code of Bagging Technique in *PUB*

With the given training set D that contains positive and unlabeled examples, D' is constructed from D, where the positive examples are unchanged but the unlabeled examples are assumed to be negative. The algorithm has two repetition loops, where the outer loop (line 3) concerns with the construction of a set of weak classifiers W, while the inner loop (line 6) generates the candidate classifiers $f_i(x)$. Any candidate classifier that has the best performance (*maxperf*) in the inner loop (line 9-12) or achieves admissible performance measure α (line 13-15) is included to a set of weak classifiers $\mathcal{W}$ (line 18). In line 7, to generate a candidate classifier, we construct a new training set from D' by random sampling with replacement and balancing the number of examples in both classes. This is one difference of our algorithm to the original bagging approach. We propose to balance the number of examples since the number of negative examples (assumed from the unlabeled examples in line 1) is usually much larger than that of positive examples. This simple proposed solution can prevent skewed decision boundaries problem occurred in an unbalanced classification [19]. Furthermore, the approach of generating a new training set by random sampling with replacement can help solve the problem that the number of positive examples is small. This issue was rarely addressed by other PU-learning algorithms.

A function $eval_{x \in P}(f_i(x))$ (line 9) is the performance measure function of the classifier $f_i(x)$ evaluated based on $x \in P$. Since the scenario of PU-learning has only positive examples as target concept, we cannot use F-measure, which is commonly used in the classification. We turn to a similar performance measure proposed in [14], which can be calculated only from the positive examples. Given $precision = p(y = 1|f(x) = 1)$ and $recall = p(f(x) = 1|y = 1)$, the performance measure is calculated as

$$\frac{precision \times recall}{p(y = 1)} = \frac{recall^2}{p(f(x) = 1)}. \tag{2}$$

This measure behaves similarly to the F-measure in that it is large only when both *precision* and *recall* are large. Therefore, we use only recall and $p(f(x) = 1)$, which can be calculated from only positive examples as a performance measure.

The algorithm iteratively runs until we get m weak classifiers as conditioned in line 3. In the last step, majority voting function $\hat{f}(x)$ on all m weak classifiers is used to predict the class of a test instance, but each vote is weighted by the performance measure of each weak classifier as shown in line 20. The identity function $I[cond]$ produces 1 if the *cond* is true, otherwise 0. Note that a classifier $f_i(x)$ in this case will output 1 or -1.

Our algorithm employs the weighting technique in the first step to construct the training examples with their corresponding weights and applies the bagging technique to learn the classification function.

4 Experiments

To study the performance of the proposed algorithm, we conduct several experiments on the real data for cool blog classification in various settings. We

also compare the performance of our algorithm with some other best performing algorithms for PU-learning.

4.1 Benchmark Corpus

We use the same corpus as presented in [1] for a task of cool blog classification. The data was provided by the blogWatcher Inc., a Japanese blog search company, for the purpose of research. There are 838,341 blog entries on distinct 60,736 blogs. Their issued dates are varied from Feb 2005 to Dec 2007. Manual labeling of cool blogs was done by an engineer in that company, who is a native Japanese speaker, without any given assumptions. The annotator selected the blogs for labeling in random, and judged the coolness of the whole blog instead of each individual blog entry. Finally, the corpus contains 270 positive blogs (with 9,827 blog entries), 270 negative blogs (with 3,836 blog entries) and the rest are left as unlabeled examples. We did not interview the annotator of his notion to label the cool blogs. This can help avoid the bias of our method to identify cool blogs. We encode each blog consisting of multiple blog entries as a simple bag-of-words vector.

Although our data is a collection of Japanese blogs, we did not use any language-specific techniques in the task of classification. In other words, the contribution of this work can be applied for identifying cool blogs in any languages if the concept of cool blog is similar to this work. Since the main objective of this work is to solve the problem of cool blog classification rather than the problem of PU-learning, the performance study of PUB against other PU-learning algorithms on standard corpus (e.g., Reuters-21578, 20Newsgroup) is out of scope.

4.2 Implementations and Parameter Settings

Our proposed algorithm is flexible for any classification method. In this work, we use SVM where we selected one implementation presented by M.W. Chang and H.T. Lin[1] that allows different weights (via regularization parameter C) for different examples. However, SVM is not a learning method that directly produces the correct probabilities. The output of SVM can be post-processed into calibrated probabilities by applying Platt scaling [20]. We apply Platt scaling only in the first step of an algorithm for weighting the examples. Through all experiments, the linear kernel function, which is commonly used for text classification, was employed with the optimized C.

Four parameters are needed to be defined in *PUB* algorithm, i.e., the number of weak classifiers (m), the admissible performance measure (α) of a weak classifier, the iteration limit of generating candidate classifiers (LIMIT) and the size of sampled training set ($|D_i|$). With the larger m, it tends to be that the more reliable result from the majority voting function will be obtained. However, the larger m also causes an algorithm to take more computational time. We can simply set m to be a specific number that has acceptable trade-off between

[1] http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/#15

performance and computational time. In this work, we set the default m to be 30. For the α, we propose to get this threshold by learning a classifier from the whole positive examples and assuming all unlabeled examples to be negative. In our experiment, we found out that this value is close to the optimized value. For our case, the default α is 0.6. The parameter LIMIT performs in the same way as parameter m. If we set smaller LIMIT, most of the weak classifiers may not achieve α performance. In the preliminary experiment, we found that the number of iterations in the inner loop is less than ten if we set the value of α not too high. We set the default value of LIMIT to be 100. Since it is not straightforward to find the optimized value of $|D_i|$, we then investigate it by the experiment. By default, we set $|D_i|$ to be 50% of the number of positive examples.

4.3 Experimental Results

All experiments were conducted in three-fold cross validation on the labeled examples where two folds of positive examples and the whole unlabeled examples are used for training. The remaining positive examples with one fold of negative examples are used for testing. Note that the other two folds of negative examples are used for training only in the case of PN-learning (described below). We show the performance by averaging the results from those three validations.

Comparison with PN-learning: We conducted this experiment to compare the performance of PUB with the learning from positive and negative examples (PN-learning). The results are shown in Table 1. We used the default setting as shown in Section 4.2 for PUB. There are two alternative ways to exploit PN-learning. The first way is to provide actual negative examples for learning denoted by "PN-learning1". This is possible for this corpus since the annotator also labeled the negative examples but it is not practical in real world application. The second way is to assume all unlabeled examples to be negative denoted by"PN-learning2". The performance of PUB is surprisingly better than both cases of PN-learning. This is possible since the given negative examples for PN-learning may not be the good representatives for negative class, and this circumstance is also the same for the case of positive examples. Therefore, only one discriminative model may not be able to predict the classes of all instances that have the target concepts. We can say that PUB is a less-biased model in the sense that it combines several outputs from the discriminative models that are guaranteed to achieve admissible performance measure on the target concepts.

Table 1. Performance of PUB and PN-learnings

Method	Accuracy	Precision	Recall	F-measure
PN-learning1	0.787	0.819	0.741	0.778
PN-learning2	0.748	0.935	0.533	0.679
PUB	0.828	0.842	0.807	0.824

Table 2. Performance of PUB and existing PU-learning algorithms

Method	Accuracy	Precision	Recall	F-measure
PEBL	0.723	0.696	0.795	0.742
Biased SVM	0.748	0.935	0.533	0.679
W-LR	0.758	0.832	0.659	0.735
WUE	0.737	0.887	0.552	0.680
PUB	0.828	0.842	0.807	0.824

Table 3. Variation of Parameters for PUB

Parameter	Value	Accuracy	Precision	Recall	F-measure
m	5	0.813	0.850	0.759	0.802
	10	0.813	0.833	0.781	0.807
	30	0.828	0.842	0.807	0.824
	50	0.830	0.850	0.800	0.824
α	0.5	0.822	0.863	0.767	0.812
	0.6	0.828	0.842	0.807	0.824
	0.7	0.802	0.793	0.819	0.806
	0.8	0.776	0.762	0.837	0.798
$\|D_i\|$	45	0.815	0.813	0.819	0.816
	90	0.828	0.842	0.807	0.824
	180	0.798	0.862	0.711	0.779
	360	0.794	0.867	0.696	0.772
size of positive	20	0.769	0.727	0.859	0.788
examples for	45	0.811	0.810	0.815	0.812
training	90	0.817	0.830	0.796	0.813
	180	0.828	0.842	0.807	0.824

Comparison with existing PU-learning algorithms. We selected four best performing algorithms (mentioned in their papers) from the families of algorithms for PU-learning to study the relative performance of PUB against the existing algorithms. PEBL [11], Biased SVM [9], Weighted Logistic Regression (W-LR, for short) [14] and weighting unlabeled examples (WUE, for short) [16] are selected to investigate on this corpus. Since there is no source code available for such algorithms except W-LR[2], we carefully implement them following their papers. The optimized value of parameters in each algorithm is manually tuned and the best result of each algorithm is shown in Table 2. The results show that PUB is superior to the other PU-learning algorithms, and those previous PU-learning algorithms cannot achieve higher performance than the PN-learning.

Variations of PUB. We also varied the setting of parameters to show the best optimized values of our algorithm. The results are shown in Table 3. Here, we can achieve higher performance when we increase the value of m. However, with higher m, more computational time is taken while the performance becomes saturated

[2] http://www.comp.nus.edu.sg/∼leews/publications/logistic.tar.gz

Table 4. Variation of method in PUB algorithm

Method	Accuracy	Precision	Recall	F-measure
original PUB	0.828	0.842	0.807	0.824
PUB without weighting technique	0.822	0.840	0.796	0.817

because of the identity function used for aggregating the classifiers. The optimized value of parameter α in this case is 0.6 since it provides the classifiers that can predict the target concept but not overfit to the training positive examples. However, this optimized value is close to the performance measure of the model that is trained from all positive examples and unlabeled examples assumed as negative, i.e., 0.53. Therefore, we can tune this parameter in this way. For D_i, our default setting for 50% of training positive examples performs the best. We also study the situation that a training set has an extremely small number of labeled examples. Although the performance of PUB algorithm becomes lower when the smaller number of positive examples is available, that performance is still better than the PN-learning approaches and other PU-learning algorithms.

We conducted the last experiment to study the significance of two techniques used in PUB algorithm by removing the weighting technique applied in the first step. The results in Table 4 show that the performance of this modified PUB becomes slightly lower than the original PUB. It means the proposed bagging technique plays an important role to improve the performance of PU-learning more than the weighting technique.

5 Conclusions and Future Work

This work addressed the problem of cool blog classification in the scenario of PU-learning which is prevalent to the real world application. We proposed an algorithm for PU-learning using weighting and bagging techniques. By investigation in several experimental settings, the proposed algorithm surprisingly showed that it can predict unseen cool blogs. This situation cannot be handled by the traditional PN-learning. The results show that the proposed algorithm is superior to other existing algorithms for PU-learning in the task of cool blog classification, and it can also deal with the case that has an extremely small number of labeled examples. For future work, we plan to extend our algorithm on other real world applications such as learning to annotate the NLP corpus using only a small set of labeled examples.

References

1. Sriphaew, K., Takamura, H., Okumura, M.: Cool blog identification using topic-based models. In: Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence, Sydney, Australia. IEEE, Los Alamitos (2008)
2. Rubin, V.L., Liddy, E.D.: Assessing credibility of weblogs. In: Proceedings of the AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs (CAAW) (2006)

3. Weerkamp, W., de Rijke, M.: Credibility improves topical blog post retrieval. In: Proceedings of ACL 2008: HLT, Columbus, Ohio, Association for Computational Linguistics, pp. 923–931 (2008)
4. Billsus, D., Pazzani, M.J.: Learning and revising user profiles: The identification of interesting web sites. Machine Learning 27, 313–331 (1997)
5. Pazzani, M.J., Muramatsu, J., Billsus, D.: Syskill & webert: Identifying interesting web sites. In: 13th National Conference on Artificial Intelligence, Portland, OR, US, vol. 1, pp. 54–61 (1996)
6. Denis, F.: Pac learning from positive statistical queries. In: Richter, M.M., Smith, C.H., Wiehagen, R., Zeugmann, T. (eds.) ALT 1998. LNCS, vol. 1501, pp. 112–126. Springer, Heidelberg (1998)
7. Zhang, B., Zuo, W.: Learning from positive and unlabeled examples: A survey. In: Proceedings of the International Symposiums on Information Processing, pp. 650–654 (2008)
8. Liu, B., Lee, W.S., Yu, P.S., Li, X.: Partially supervised classification of text documents. In: ICML, pp. 387–394 (2002)
9. Liu, B., Dai, Y., Li, X., Lee, W.S., Yu, P.S.: Building text classifiers using positive and unlabeled examples. In: ICDM 2003: Proceedings of the Third IEEE International Conference on Data Mining, p. 179. IEEE Computer Society Press, Washington (2003)
10. Li, X., Liu, B.: Learning to classify texts using positive and unlabeled data. In: IJCAI 2003: Proceedings of Eighteenth International Joint Conference on Artifical Intelligence, pp. 587–594 (2003)
11. Yu, H., Han, J., Chang, K.C.: Pebl: Web page classification without negative examples. IEEE Trans. on Knowl. and Data Eng. 16(1), 70–81 (2004)
12. Denis, F., Gilleron, R., Tommasi, M.: Text classification from positive and unlabeled examples. In: IPMU 2002, 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, pp. 1927–1934 (2002)
13. Denis, F., Gilleron, R., Laurent, A., Tommasi, M.: Text classification and cotraining from positive and unlabeled examples. In: Proceedings of the ICML Workshop: the Continuum from Labeled Data to Unlabeled Data in Machine Learning and Data Mining, pp. 80–87 (2003)
14. Lee, W.S., Liu, B.: Learning with positive and unlabeled examples using weighted logistic regression. In: ICML 2003: Proceedings of the Twentieth International Conference on Machine Learning (2003)
15. Zhang, D., Lee, W.S.: A simple probabilistic approach to learning from positive and unlabeled examples. In: Proceedings of the 5th Annual UK Workshop on Computational Intelligence (2005)
16. Elkan, C., Noto, K.: Learning classifiers from only positive and unlabeled data. In: SIGKDD (2008)
17. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
18. Kamishima, T., Hamasaki, M., Akaho, S.: Baggtaming – learning from wild and tame data. In: Proceedings of the Workshop on Web 2.0 Mining at ECML PKDD 2008 (2008)
19. Morik, K., Brockhausen, P., Joachims, T.: Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In: ICML, pp. 268–277 (1999)
20. Platt, J.C.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Smola, A., Bartlett, P., Scholkopf, B., Schuurmans, D. (eds.) Advances in Large Margin Classifiers, pp. 61–74 (1999)

Thai Word Segmentation with Hidden Markov Model and Decision Tree

Poramin Bheganan, Richi Nayak, and Yue Xu

Faculty of Science and Technology, Queensland University of Technology, Australia
poramin.bheganan@student.qut.edu.au, {r.nayak,y.xu}@qut.edu.au

Abstract. The Thai written language is one of the languages that does not have word boundaries. In order to discover the meaning of the document, all texts must be separated into syllables, words, sentences, and paragraphs. This paper develops a novel method to segment the Thai text by combining a non-dictionary based technique with a dictionary-based technique. This method first applies the Thai language grammar rules to the text for identifying syllables. The hidden Markov model is then used for merging possible syllables into words. The identified words are verified with a lexical dictionary and a decision tree is employed to discover the words unidentified by the lexical dictionary. Documents used in the litigation process of Thai court proceedings have been used in experiments. The results which are segmented words, obtained by the proposed method outperform the results obtained by other existing methods.

Keywords: Hidden Markov Model, Thai Word segmentation, Decision tree.

1 Introduction

In civil court proceedings, judges need to read the documents submitted for a litigation case by litigants to settle the points in dispute by finding facts and relevant information. These documents, called plaint and defend, lodged by the litigants, referring to plaintiff and defendant respectively are approximately 20 – 45 pages long. Judges read and analyse these documents to find points in dispute for which the plaintiff and defendant do not agree and take action. There are approximately 13,000 cases in the civil court of Thailand to be proceeded in each year and there are only 100 judges responsible for these cases [4]. It is an overwhelming task for a judge to read the documents, identify the points in dispute, and reach a decision. With the recent advancements in computing resources and data mining techniques, it is feasible to automate some of these tasks for judges. However, the first step is computationally processing Thai documents.

The computational processing of Thai written language is different and more complicated than the English language. Thai text does not have word boundaries and there is no delimiter to separate two sentences. The spaces found in documents are for making the readers comfortable with reading the passages rather than separating words or sentences. The task includes identification of syllables, words and then sentences. Some research has been done in Thai word segmentation [1, 9, 12] but a

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 74–85, 2009.
© Springer-Verlag Berlin Heidelberg 2009

higher accuracy can still be achieved. Most of the conventional Thai word segmentation methods use a dictionary to segment the texts [1, 5, 9]. If the dictionary does not contain the matching word, it leads to error. There is some research that uses a non-dictionary word segmentation method using decision trees [12]. The decision tree is trained to learn the basic grammar rules and to represent the principles or general rules about the interested information [12]. Similarly, the work of [9] used the decision tree to extract Thai words from a number of Thai corpora to build a corpus-based Thai dictionary. However, no grammar rules were used.

To increase the accuracy of the dictionary-based methods, there must be a sophisticated process to accurately segment the Thai texts into words. There can be ambiguous words formed by combining possible syllables. There are some shortcomings with the sole use of the decision tree for Thai language processing as the tree needs to use the grammar rules to construct it. If the grammar-rules are incorrectly defined, especially for the unknown words or words derived from other languages, the tree cannot generate suitable output. On the other hand, a dictionary-based word segmentation method needs no grammar rules but it needs all words to exist in the dictionary. If the potential words are not in the dictionary, the method cannot generate the output either.

This paper proposes a novel method of Thai word segmentation by combining a probability-based model using the six-state left-right Hidden Markov model (HMM) with the dictionary-based model using the decision tree. The HMM model is used to extract possible words combining several possibilities of syllables. If the probability-based model is used alone, the quality of the word segmentation heavily relies on the quality and quantity of the training data set. After words are formed as the output of HMM, the TCL's lexicon dictionary [11] is used to verify those words. The identified words are tagged according to their part of speech using this dictionary. If a word formulated from HMM cannot be identified by the dictionary, it is sent as a group of syllables to the decision tree to be identified as a possible word. The decision tree is trained to learn the patterns of combining syllables into words, by using the words that exist in the lexicon dictionary as a training set.

In this paper, we study the use of the combination of two different word segmentation techniques. We compared the results obtained by the proposed method with the results obtained by using the decision tree only, by using the six-state left-right HMM only, by using the three-state left-right HMM only, and by using the lexicon dictionary only. The proposed method significantly outperforms all other methods. The results show improvement in precision and recall for Thai language processing when applying the proposed method to a real-world data set.

The novel contribution of this paper is to develop a Thai word segmentation and processing method which combines the benefit of probabilistic and dictionary-based word segmentation together.

2 Problem Definition: Automatic Thai Language Processing

Thai characters are members of the Brahmic family of characters descended and modified from Brahmi [13]. The Thai written language does not use the conjunct

consonant mechanism and independent vowel letters that are found in most Brahmic writing [13]. This makes Thai language processing more difficult [1, 9, 10, 12].

There are 44 consonants and 32 vowel forms which can be written in front of, on top of, at the bottom of, and at the end of the consonants. In addition, four different tonal marks and special characters are used to form a Thai word. The smallest unit of Thai word is defined as a syllable which consists of at least one consonant with or without vowel(s). There is no space to separate among words or even sentences. Figure 1 shows an example of Thai text containing three paragraphs, four sentences, 211 words, and 282 syllables.

การนำรถยนต์คันที่เช่าซื้อไปขาย ถ้าโจทก์แจ้งให้จำเลยทราบก่อน จำเลยก็จะซื้อคืนหรือหาผู้ซื้อ ได้ราคาสูงกว่าที่โจทก์ขาย เพราะรถยนต์คันที่โจทก์ยึดคืน อยู่ในสภาพเรียบร้อย ใช้การได้ดีและมีอุปกรณ์ตกแต่งครบถ้วนอยู่ในสภาพใหม่ การขายของโจทก์ที่ขายได้ราคาเพียง ๑๘๖,๐๐๐ บาท จึงไม่สุจริตและไม่ถูกต้อง

ตามฟ้องของโจทก์ โจทก์เรียกค่าขาดประโยชน์ โดยกล่าวอ้างว่าสามารถนำรถยนต์ที่เช่าซื้อให้บุคคลอื่นเช่าได้ไม่ต่ำกว่าเดือนละ ๗,๐๐๐ บาท ซึ่งโจทก์ขอคิดนับแต่วันที่ ๑๔ สิงหาคม ๒๕๓๑ อันเป็นวันผิดสัญญาถึงวันยึดรถคืน เป็นเวลา ๑๓ เดือน คิดเป็นค่าเสียหาย ๙๑,๐๐๐ บาท

จำเลยที่ ๒ ขอกราบเรียนว่า การคิดคำนวณค่าเสียหาย ค่าขาดประโยชน์ของโจทก์ ดังกล่าวในข้อนี้ เป็นการคิดโดยความคาดหมายของโจทก์เองทั้งสิ้น ตามวิสัยและพฤติการณ์ของพ่อค้า ซึ่งเป็นการเอาเปรียบผู้บริโภค ความจริงแล้ว รถยนต์คันที่เช่าซื้อไม่สามารถนำออกให้เช่าได้ในราคา ๗,๐๐๐ บาทต่อเดือน ตามที่โจทก์กล่าวอ้าง แต่จะได้ราคาต่ำกว่าที่โจทก์เรียกร้อง ซึ่งอย่างมากก็ไม่เกินเดือนละ ๒,๐๐๐ บาท แต่อย่างไรก็ตามจำเลยที่ ๒ ก็ไม่ต้องรับผิดในส่วนนี้

Fig. 1. An example of Thai paragraphs

Information processing of documents in Thai language is considered more difficult in comparison to English language. The reasons are manifold: (1) there is no simple way to determine where a syllable starts and ends; (2) once a symbol is recognised as a syllable, the problem becomes apparent as to where a word starts and ends; (3) the next problems is how words in a sentence can be segmented correctly according to their context; (4) what words make a sentence due to the absence of word boundary in a sentence; and (5) what sentences form a paragraph.

3 The Proposed Method Combining the HMM and Decision Tree Models

The proposed method of Thai word segmentation, as depicted in Figure 2, combines the techniques of probabilistic (that is, non-dictionary based) and decision-tree (that is, dictionary based) modeling. The integration of these two techniques allows overcoming of the shortcomings of each individual technique. Both are different in detail or at the level of the analysis which makes the integrated process and result accurate. The process starts from syllable segmentation in Phase 1 by applying the standard language rules. The process is then followed by word segmentation in Phase 2. The initial word segmentation phase uses the HMM based non-dictionary word segmentation technique. When the first set of identified words are extracted using the HMM, the dictionary-based word segmentation technique with the lexical dictionary [11] is used in revising and combining the possible list of words before identifying the sentence. Any syllables unidentified by the dictionary are processed by the decision tree to turn into a tagged word.

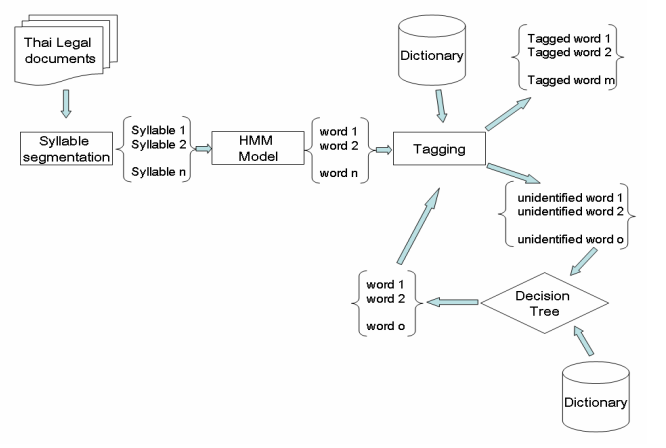


Fig. 2. Thai word segmentation process using HMM and Decision Tree

3.1 Phase 1: Syllable Segmentation

This task includes the analysis of input strings in a document to separate syllables. The process uses the standard Thai syllable formation rules to extract possible syllables, using the necessary grammar rules to suit all possible Thai syllables. The input string is analysed by applying the grammar rules for identifying the possible pattern of the syllables. The intermediate result may have some ambiguous syllables. Due to this, the segmented syllables are then combined and formulated for the possible words using the Hidden Markov model in the next phase.

This research implemented the syllable rules to formulate syllables from the Thai strings existing in a document. The syllable rules include fifteen (15) Thai consonant rules and Thirty Three (33) syllable structure rules. Before applying these syllable rules, a string is parsed though the Thai syllable structure to identify the syllables present in the string. This formula is able to recognise a syllable formed by the combinations of 44 consonants, 21 vowels, and four different tonal marks. It can also recognise the Thai vowels located in three levels: upper, middle and lower levels. The Thai syllable structure is shown in the equation (1):

$$S = [Vi] + Ci + [Cm] + [Vc] + [Tn] + [Vf] + [Cm] + [Cf] \quad (1)$$

S is a syllable. *Vi* is an initial vowel. *Ci* is initial consonant. *Cm* is initial-compound consonant. *Vc* is centre vowel. *Tn* is tone. *Vf* is final vowel. *Cf* is final consonant. Terms in brackets *[ ]* are optional.

The above formula stands true for every syllable formation. For example, a string "ซื้อขาย" consists of two syllables which are "ซื้อ" and "ขาย". The first syllable contains value for the following variables Ci ="ซ", Vc="ื", Tn="้", and Vf = "อ". When the next consonant "ข"is detected in the next position of string, the next possible syllable is started. The second syllable consists of Ci="ข", Vc="า", and Cf="ย".

In order to form a syllable, a string is parsed using the above formula. Then, the syllable rules are applied to verify a syllable. The composition of a syllable in Thai language is unambiguous and can be defined by a set of fifteen (15) Thai consonant rules and thirty three (33) syllable structure rules. These are some examples of rules that are implemented to process a string:

- The initial vowel must not be in the last position of a syllable.
- The initial compound consonant must not be in the last position of a syllable.
- A final vowel and the previous character have to be grouped into a same unit.
- A Final vowel must be in the last position of a syllable.
- A Garun (a particular vowel) must not be in the initial vowel position.
- An Upper vowel must not be in the initial vowel.
- An Upper vowel must not be in the final vowel.
- A Lower vowel must not be in the initial vowel.
- A Lower vowel must not be in the final vowel.
- There must be only one Tonal mark in a syllable.

3.2 Phase 2: Word Segmentation

The previous phase merges and groups the characters in the input string into syllables. Once the syllables are extracted, the next task is to identify combinations and form the possible words.

3.2.1 Word Segmentation Using HMM

The theory of a Hidden Markov Model (HMM) was derived from the Markov chains theory [2]. In an HMM, a state corresponds to each possibility and is not directly observable (i.e. it is hidden). Given a sequence of observation symbols, the HMM will be able to predict the probability of the sequence based on previous observations used to build the HMM. Given two sequences of observation symbols, the model communicates the probabilities of these two sequences. Therefore, it is possible to decide which one is better. In terms of applying HMM to the Thai word segmentation problem, if each state corresponds to a character position and the character values are the observation symbols, the HMM will be able to tell the best sequence of characters among a number of sequences given to the model.

In this paper, a left-right Hidden Markov model of six states is developed for Thai word segmentation. A six-state HMM model has been chosen after finding that the majority of words are the size of one to six syllables in the Thai TCL's lexicon dictionary developed by the Thai Computational Linguistics Laboratory [11]. In addition, the left-right model has been chosen because the Thai language is written from left to right [5]. In this model, a one-syllable word is discovered when the loop transition is at state 1, whereas, a multi-syllable word is discovered when the transition is from state 4 to state 1. A one-syllable word will pass only state 1, while a two-syllable word passes states 1 and 6. A three-syllable word passes states 1, 5, and 6, whereas, a four-syllable word passes states 1, 2, 5, and 6. A five-syllable long word passes states 1, 2, 3, 5, and 6, while, a six-syllable word passes states 1, 2, 3, 4, 5, 6.

In order to determine the most suitable words in the document, a Hidden Markov Model can be represented by equation 2:

$$\lambda = (A, B, \pi) \tag{2}$$

A is the state transition probability distribution matrix where a_{ij} is the transition probability from state i to state j. B is a matrix in which each row contains the probability distribution of observation symbols in particular state. π is the initial state distribution in the vector of i column, which indicates how likely the whole sequence of the trials start from state i.

With the model in Figure 3, the transition probability for the corresponding HMM model can be formulated as matrix A. This matrix shows the possibility of each word when in transit from one state to another state. Each word will pass a different group of states. The value will be calculated according to matrix A, B, and π.

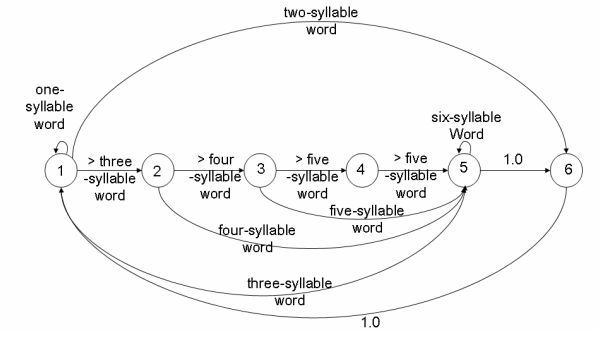


Fig. 3. A six-state left-right Hidden Markov model for the Thai word segmentation problem

$$A = \begin{bmatrix} a_{11} & a_{12} & 0.0 & 0.0 & a_{15} & a_{16} \\ a_{11} & a_{12} & 0.0 & 0.0 & a_{15} & a_{16} \\ 0.0 & 0.0 & 0.0 & a_{34} & a_{35} & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & a_{45} & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & a_{55} & 1.0 \\ 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

Where a_{11}, a_{12}, a_{15}, a_{16}, a_{23}, a_{25}, a_{34}, a_{35}, a_{45}, and a_{55} are the ratios between the number of words within one syllable, with more than three syllables, with three syllables, with two syllables, with more than four syllables, with four syllables, with more than five syllables, with five syllables, with more than five syllables, and with more than six syllables respectively over the number of words in the training data set.

For initial state distribution π, every word must start from state 1. Therefore,

$$\pi = \begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

To formulate B as the matrix of 6X6, an observation symbol will be counted in a particular state when each possible word is parsed into HMM. The counted values will be divided by the number of times the specific state is passed through [8].

Once the matrices A, B and π are obtained, the HMM for the given data can be executed. If the HMM is used alone, the quality of the word segmentation heavily relies on the quality and quantity of the training data set. However, many words are inaccurately formed (as shown in experiments) and there remains a need to parse through the words with a dictionary to ensure the accuracy of the word and tag the words with its part-of-speech.

3.2.2 Dictionary-Based Word Segmentation Using Decision Tree

After each word is identified by the HMM, the lexicon dictionary is used to tag the part-of-speech to each word. If a word is not identified by the lexicon, it is processed by the decision tree. The decision tree is trained with various information that a word in the lexicon dictionary has. All the words that exist in the lexicon dictionary form the training set for decision tree modeling. During the iteration, the tree is branched according to the frequency of the occurrence of the words. The attributes in the decision tree include the following information gathered from each word:

1 Left Mutual information and Right Mutual Information

The mutual information of the co-occurrence of the word [3] is calculated as one of the attributes in the tree. The mutual information on the left and the right [9] will be calculated as followed in equations (3) and (4):

$$L_m(xyz) = \frac{p(xyz)}{p(x)\,p(yz)} \tag{3}$$

$$R_m(xyz) = \frac{p(yz)}{p(xy)\,p(z)} \tag{4}$$

where

- Lm is the left mutual information of the noticing string
- Rm is the right mutual information of the noticing string
- x is the leftmost syllable of xyz
- y is the middle syllable of xyz
- z is the rightmost syllable of xyz
- $p(\ )$ is the probability of the occurrence

2 Length of word

The number of syllables to form a word is counted as an attribute in the decision tree. The possible lengths of common Thai words are between one to six syllables.

3 Functional words

In the Thai language, there are many functional words such as “then” and “will”. The appearance of these functional words may cause the grouping of syllables to form a word wrongly [9]. Therefore, a function *Func(s)* is considered as an attribute to eliminate these functional words from the string pattern.

$$Func(s) = \begin{cases} 1 & \text{if string contains a functional word,} \\ 0 & \text{if otherwise.} \end{cases}$$

These attributes values are prepared for a total of 30,000 words that exist in the lexicon dictionary. The C4.5 learning algorithm [7] is used to train the decision tree with this data set. The unidentified words from the previous phase are tested with this trained decision tree using the patterns of combining syllables that exist in the words of training data.

4 The Proposed Method Combining the HMM and Decision Tree Models Empirical Analysis and Discussion

The proposed method starts by applying the Thai language grammar rules to identify the syllables. Syllables are combined into possible words with HMM. The lexicon dictionary is then used in verifying the words and the words that are not verified by the dictionary are checked in the decision tree for identification as suitable words.

Data Set: There are two sets of input data used in the experiment. The first set of data is the legal documents that need to be processed for word segmentation. These legal documents are collected from the archive centre of the Court of Justice, Thailand [4]. All documents are civil cases which were pleaded between the years 2000 to 2003. There are a total of 300 cases collected that are used in experiments. In each case, there are about 5 – 15 pages of the plaint document and about 10 - 20 pages of the defend document. Each case also contains about 5 – 10 pages of judgment verdict.

The second set of data is the TCL's Thai lexicon dictionary [11]. As stated earlier, there are 30000 unique words in the lexicon. The lexicon dictionary is used to verify and tag words from HMM. It is also used for training the decision tree.

Evaluation Criteria: Precision, recall, and F-measure are used as the evaluation criteria in experiments. Precision is the ratio of words retrieved compared to all of the words in the legal documents. Recall is the ratio of the relevant words retrieved compared to all relevant words in the legal documents. F-measure is the weighted harmonic mean of precision and recall.

10-fold CV experiments are used throughout the experiments and the average performance is reported along with standard deviation.

4.1 Empirical Analysis and Discussion

Results obtained by the proposed method are compared with the results obtained by the decision tree only, by the six-state left-right HMM, by the three-state left-right HMM and by lexicon only.

Five (5) sets of comparative experiments are conducted to evaluate the proposed method. They are as follows:

- The six-state left-right HMM in the proposed method is replaced with the three-state left-right HMM to check whether the six-state left-right HMM is suitable for Thai processing. Results of the three-state left-right HMM are integrated with the Lexicon and C 4.5 decision tree as described in our method.

- The six-state left-right HMM is used alone for the word segmentation problem. It means that the decision tree and dictionary are not used to further enhance the quality of word segmentation. The output words produced from the HMM become the final word set.
- The three-state left-right HMM is also used alone for the word segmentation problem. It means that the decision tree and dictionary are not used to further enhance the quality of word segmentation.
- The lexicon dictionary is used alone to identify the words. The process of this experiment is as follows. The Thai language grammar rules are used to identify the syllables, then syllables are segmented. The first combination a group of syllables form becomes the word for lexicon if this word exists. Even though this may not be the correct interpretation.
- The decision tree is used alone to identify the words. The process of this experiment is as follows.

The training dataset from each fold of 270 legal documents consists of an average of 534330 words for HMM. The decision tree training dataset consists of 30,000 records, each record representing a word in the lexicon dictionary. Each record is made of four (4) input attributes, namely Left mutual information, Right mutual information, length of words, and functional words, and one (1) binary output attribute that is a word formed or not. The test dataset in our experiments from each fold of 30 legal cases consists of an average of 57360 words.

Table 1 shows the numbers of words and syllables found in the data set. This table also includes the number of words identified by the proposed method and other methods used for comparison. The numbers of words identified by the lexicon are the highest, 677846 for training dataset, and 62413 in test dataset. It can also be observed that the numbers of words identified by the lexicon are even more than the number of words in the documents (that are 534330 in training dataset and 57360 in test dataset).

The reason is that when a group of syllables in a string is presented to the lexicon dictionary, it accepts the first possible combination of syllables found in the lexicon as a word. It ignores any other combination that may be formed as a word by adding more syllables next in the string. Consequently, most of the syllables in the input set matched with one-syllable words in the lexicon. It can also be observed from Table 2 that it has the least accuracy as a result.

Table 1. Numbers of words and syllables in the experiments

Method/ Information	Dataset	
	Training (270 documents)	Test (30 documents)
Total No. of words	534330	57360
Total No. of syllables	1870155	216821
No. of words identified by 6-state left-right HMM only	341139	37159
No. of words identified by 6-state left-right HMM + Lexicon + Decision Tree (the proposed method)	348623	38862
No. of words identified by 3- state left-right HMM only	316445	36881
No. of words identified by 3-state left-right HMM + Lexicon + Decision Tree	306285	31285
No. of words identified by Lexicon Only	677846	62413
No. of words identified by Decision Tree Only	356173	39458

Table 2. Precision, Recall, and F-score from the experiments

Method	Precision		Recall		F-Score		Elapsed time	
	Training	Test	Training	Test	Training	Test	Training	Test
6-state HMM + Lexicon + DT	82.65% (0.530)	79.96% (1.272)	53.92% (1.348)	54.17% (0.987)	65.26% (0.959)	64.59% (1.012)	56.51s	55.16s
6-state HMM only	65.51% (0.707)	64.34% (1.254)	41.83% (0.871)	41.68% (0.896)	51.05% (0.775)	50.59% (0.609)	53.80s	52.42s
3-State HMM + Lexicon +DT	59.81% (0.800)	59.72% (0.919)	34.28% (1.413)	32.57% (0.769)	43.58% (1.139)	42.15% (0.576)	35.18s	33.96s
3-state HMM only	50.73% (1.305)	50.06% (1.194)	30.04% (1.429)	32.18% (1.183)	37.74% (1.093)	39.18% (0.908)	32.07s	31.23s
DT only	73.03% (1.009)	72.84% (0.847)	48.68 % (1.335)	50.11% (0.873)	58.42% (0.968)	59.37% (0.721)	10.06s	9.95s
Lexicon only	28.83% (0.736)	24.73% (0.927)	36.58% (0.751)	26.90% (1.007)	32.25% (0.501)	25.77% (0.862)	10.15s	10.03s

* Values in parentheses are Standard Deviation.

The number of words identified by the six-state left-right HMM only is the lowest (34119 in training dataset and 37159 in test dataset). As HMM is a statistic-based method, the quality of the output depends upon the quality of training dataset. The number of identified words can be improved if there are more training datasets with good quality. There are approximately 20,000 words that HMM is unable to correctly identify. It can be seen that when we included the lexicon dictionary and decision tree model to further identify the words produced from HMM, the number is increased to 348623 for the training data set and 38862 for the test data set.

Besides, the experiment results also reveal that numbers of identified words by decision tree (356173 in training and 39458 in test dataset) are more in quantity than when only using HMM. This is because the DT is trained on the entire lexicon words. However, it can be seen that their accuracy is lower in comparison to our results.

Table 2 reports the precision, recall, F-score measures, and elapsed times of the experiments. The precision, recall, F-Score values gained by our proposed method are the highest among other methods, whereas those of the lexicon dictionary only method give the lowest performance among other methods. This is because, when the syllables are compared with the lexicon, most syllables were matched with one-syllable words in the dictionary. The improvement is 55.23% for precision value and 27.27% for recall value for the test data set.

The precision and recall ratio on the test set is 63.34% and 41.68% respectively when the six-state left-right HMM alone is used to identify the words. The improvement is 15.62% for precision value and 12.49% for recall value when the outputs by HMM are combined with lexicon and DT.

The experiment also reports that if we change the HMM from six-state to be three-state left-right HMM [6] without using the decision tree, the precision ratio is dropped to 50.06% and the recall ratio is dropped to 32.18% in the test dataset. This number reveals that Thai words tend to be better segmented by using the six-state left-right HMM rather than using the three-state left-right HMM. When we combine the outputs

of the three-state left-right HMM with the lexicon and the decision tree part, the precision ratio is increased to 59.72% and the recall ratio is slightly increased to 32.57%. It is, however, significantly lower than the proposed method.

Considering elapsed times in experiments, the proposed method consumed the maximum time of 56.51 seconds on training and 55.16 seconds on testing whereas the decision tree took the least time. The training dataset used only 10.06 seconds and the testing consumed 9.95 seconds. This is because the proposed method consumes the first period of time to retrieve the intermediate results using the six-state left-right HMM. The remaining time consumed are counted when unknown words from the HMM is parsed to the lexicon and the rest of unsegmented words parsed to the decision tree. The time consumed in the experiment on the decision tree only is the minimal since there is no intermediate result to pass to another process.

It can be ascertained based on the results reported in Table 1 and Table 2 that the proposed method produces the best results as it utilizes the strength of each individual method used. The words unidentified by the HMM only are further processed using the decision tree and the overall accuracy is enhanced.

5 Conclusion

In this paper, we combine the techniques of probabilistic method word segmentation as well as dictionary-based word segmentation together. Both of the techniques provide different advantages and drawbacks. The integration of these two techniques allows overcoming the shortcomings of each individual technique. Most are different in detail or the level of the analysis, which makes the results different. The initial word segmentation technique is developed by using the non-dictionary word segmentation. When the first set of words is extracted, the dictionary-based word segmentation is used to help revise and combine the possible list of words before identifying the sentence.

The proposed method is extensively evaluated against other methods. Results show that the proposed method significantly outperforms all other methods. The word segmentation accuracy (in terms of precision and recall ratio) can be further improved. Firstly, more training and testing data will help with improving the accuracy of the experiment. The data set used in this paper was collected from real legal cases. The number of cases lodged was limited in the specified period. If there were more cases, the accuracy would have been increased. Secondly, the nature of the words in these data sets is different from the nature of the words that exist in the lexicon dictionary used. The domain of collected data is in the legal domain, while, the dictionary used in refining the words is the general lexicon. Therefore, some legal- specific terms might not be present in the dictionary.

Future work includes experimentation with some more datasets and the construction of the domain-specific vocabulary that can be used with the dictionary, to help with segmenting the more suitable words according to the domain. We also plan to use the tagged words to discover the relationships of particular paragraphs in the plaint and the defend.

References

1. Aroonmanakul, W.: Collocation and Thai Word Segmentation. In: Joint International Conference of SNLP-Oriental COCOSDA, Thailand, pp. 68–75 (2002)
2. Christen, P., Churches, T., Hegland, M., Lim, K., Nielsen, O.M., Roberts, S., Zhu, J.: High-Performance Computing Techniques for Record Linkage. In: Australian Health Outcomes Conference, Canberra, Australia, pp.1–14 (2002)
3. Church, K.W., Robert, L., Mark, L.Y.: A Status Report on ACL/DCL. In: 7th Annual Conference of the UW Centre New OED and Text Research: Using Corpora, Canada, pp. 84—91 (1991)
4. Civil court of Thailand, http://www.cvcourt.com
5. Kawtrakul, A., Thumkanon, C., Poovarawan, Y., Varasrai, P., Suktarachan, M.: Automatic Thai Unknown Word Recognition. In: Natural Language Processing Pacific Rim Symposium, Phuket, Thailand, pp. 341–346 (1997)
6. Nagata, M.: Context-based spelling correction for Japanese OCR. In: 16th conference on Computational linguistics, New Jersey, USA, pp. 806–811 (1996)
7. Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufman, USA (1993)
8. Rabiner, L.R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. IEEE 77(2), 257–285 (1989)
9. Sornlertlamvanich, V., Potipiti, T., Charoenporn, T.: Automatic corpus-based Thai word extraction with the C4.5 learning algorithm. In: 18th conference on Computational linguistics. Saarbrücken, Germany, pp. 802–807 (2000)
10. Sudprasert, S., Kawtrakul, A.: Thai word segmentation based on Global and Local Unsupervised learning. In: NCSEC, Chonburi, Thailand (2003)
11. Thai Computational Linguistics Laboratory.: TCL's Computational Lexicon, http://www.tcllab.org/tcllex/
12. Theeramunkong, T., Usanavasin, S.: Non-dictionary-based Thai word segmentation using decision trees. In: The first international conference on Human language technology research, New Jersey, USA, pp. 1–5 (2001)
13. Unicode Consortium.: The Unicode Standard 4.0: Southeast Asian Scripts. Addison Westley, California (2004)

An Efficient Method for Generating, Storing and Matching Features for Text Mining

Shing-Kit Chan and Wai Lam

The Chinese University of Hong Kong
{skchan,wlam}@se.cuhk.edu.hk

Abstract. Log-linear models have been widely used in text mining tasks because it can incorporate a large number of possibly correlated features. In text mining, these possibly correlated features are generated by conjunction of features. They are usually used with log-linear models to estimate robust conditional distributions. To avoid manual construction of conjunction of features, we propose a new algorithmic framework called F-tree for automatically generating and storing conjunctions of features in text mining tasks. This compact graph-based data structure allows fast one-vs-all matching of features in the feature space which is crucial for many text mining tasks. Based on this hierarchical data structure, we propose a systematic method for removing redundant features to further reduce memory usage and improve performance. We do large-scale experiments on three publicly-available datasets and show that this automatic method can get state-of-the-art performance achieved by manual construction of features.

1 Introduction

Log-linear models have become popular in text mining because of its capability to incorporate a vast number of possibly correlated features, which is common in text mining tasks. Given an instance $\mathbf{x}$ and its corresponding output $\mathbf{y} \in \mathcal{Y}$, we present a feature vector $\mathbf{F}(\mathbf{x}, \mathbf{y})$ to a learning system for training. In log-linear models, we aim to learn the conditional distribution:

$$\Pr(\mathbf{y}|\mathbf{x}; \mathbf{\Lambda}) = \frac{\exp[\mathbf{\Lambda} \cdot \mathbf{F}(\mathbf{x}, \mathbf{y})]}{Z(\mathbf{x}; \mathbf{\Lambda})} \tag{1}$$

where $\mathbf{\Lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ is the training model parameter, and $Z(\mathbf{x}; \mathbf{\Lambda}) = \sum_{\mathbf{y}' \in \mathcal{Y}} \exp[\mathbf{\Lambda} \cdot \mathbf{F}(\mathbf{x}, \mathbf{y}')]$ is the normalization factor.

To help our discussion, we assume $\mathbf{x}$ is a sentence and $\mathbf{y}$ is a sequence of tags for each word in $\mathbf{x}$. To keep the size of the feature vector $\mathbf{F}(\mathbf{x}, \mathbf{y})$ tractable, usually we assume first order independence on the output $\mathbf{y}$ and write $\mathbf{F}(\mathbf{x}, \mathbf{y})$ as the sum of features at all positions of $\mathbf{x}$.

$$\mathbf{F}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{|\mathbf{x}|} \mathbf{f}(\mathbf{x}, y_{i-1}, y_i, i) \tag{2}$$

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 86–97, 2009.
© Springer-Verlag Berlin Heidelberg 2009

where $|\mathbf{x}|$ is the length of sentence $\mathbf{x}$, and $\mathbf{f} = (f_1, f_2, \ldots, f_m)$. Each f_k is a feature function defined at position i of sentence $\mathbf{x}$. More precisely, it is a binary function of the whole sentence $\mathbf{x}$, the previous label y_{i-1}, the current (i.e. the i-th) label, and the position i.

To see the importance of conjunction of features in text mining, let us consider the following example. Suppose we have instances $\mathbf{p}$ = "Morgan", $\mathbf{q}$ = "Stanley", and $\mathbf{r}$ = "Morgan Stanley". Usually we tend to think that $\mathbf{p}$ and $\mathbf{q}$ are two people. If we blindly extend the inference without considering the two words together as "Morgan Stanley", we will conclude that $\mathbf{r}$ is also a person (but it is an investment bank). This shows that we need to consider "Morgan Stanley" as one object in our learning algorithm. However, features are usually constructed manually and there is little research done on the construction of *conjunctions of features* for text mining tasks. For example, in a noun-phrase chunking task, we observe a sentence $\mathbf{x}$ where each word x_i has a label $y_i \in \{\text{B, I, O}\}$ to indicate that a word is the "beginning", "inside", or "out" of a noun-phrase respectively. Thus a feature $(x_i = \text{boy}, y_i = \text{B})$ may be defined as

$$f(\mathbf{x}, \mathbf{y}, i) = \begin{cases} 1 & \text{if } x_i \text{ is the word "boy",} \\ & \text{and } y_i \text{ is the label "B"} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Manually writing all these features is not feasible. A possible solution to this problem is to define a set of feature templates and let the computer generate the features automatically. For example, consider two feature templates: $A1 : (x_i, y_i)$, and $A2 : (x_{i-1}, x_i, y_i)$ When these two feature templates work on the following labeled sentence,

```
x: I would like to thank my advisor .
y: B O     O    O  O     B  I       O
```

the features generated by each template are:

$$\begin{aligned} A1 :& (x_i = \texttt{I}, y_i = \texttt{B}), (x_i = \texttt{would}, y_i = \texttt{O}), (x_i = \texttt{like}, y_i = \texttt{O}), \ldots \\ A2 :& (x_{i-1} = \texttt{START}, x_i = \texttt{I}, y_i = \texttt{B}), (x_{i-1} = \texttt{I}, x_i = \texttt{would}, y_i = \texttt{O}), \\ & (x_{i-1} = \texttt{would}, x_i = \texttt{like}, y_i = \texttt{O}), \ldots \end{aligned}$$

where `START` denotes the start of a sentence. In this way, the problem of exactly defining a set of features is reduced to that of defining a set of feature templates. In many cases, users may want to define a threshold δ to remove a feature f that occurs rarely, i.e. $\text{Count}(f) < \delta$. An alternative to using a threshold is using a statistical test on the features as in [1]. This feature template technique is used in programs such as [2], [3], and [4]. However, researchers often resort to intuition or a trial-and-error method to determine whether a feature should be used.

As long as computation power is not a concern, it is expected that incorporating more features in a mining system increases the performance. In [5], a noun-phrase chunking system is developed by using the above feature template technique. On a complete feature set, the system achieve 94.38% F1. On a reduced feature set where the features are *supported* in the training set, the

system performance is 94.19% F1. However, it is not true in all cases. In [1], the model with a complete feature set (where a semantic lexicon is added) gives a lower accuracy. Using orthographic features only, the biomedical named entity recognition performance is 69.8% F1 on the evaluation set (and 88.9% F1 on the training set). The performance decreases to 69.5% F1 on the evaluation set (and 86.7% F1 on the training set) when both orthographic features and semantic lexicons are used. A more interesting study is presented in [6]. The performance of a named entity recognition system with hand-engineered features is 73% F1. However, without the tedious and elusive manual search in the feature space, a simple model with only word features can achieve 80% F1. The above results suggest that adding more features does not necessarily increase the performance.

To investigate the above problem, we need an efficient method to generate and store the enormous number of features. We also need to efficiently get all the features activated in our feature space, given an instance. In this paper, we propose a new algorithmic framework called F-tree for automatically generating and storing conjunctions of features in text mining tasks. Instead of a complete set of feature templates, our approach requires only a set of *atomic* feature templates. These atomic feature templates generate the atomic features on which high-level *molecular* features are based. An instance may activate many high-level molecular features and more features can be considered than that of a hand-engineered approach. It is infeasible to find all the molecular features activated by an instance by using a one-vs-one test, i.e. test if an instance can activate a feature and then the next feature. To efficiently retrieve these activated features, we develop a prefix tree algorithm to store all features and do a one-to-all test. We further improve our framework by proposing a systematic method for removing redundant features based on a hierarchical data structure, which is helpful in representing the relationships between the low-level features and the high-level features. In practice, the hierarchy can be integrated with the prefix tree to save storage space.

2 Preliminaries

In many text mining tasks, a set of training data is given and the problem can be cast as a supervised learning problem. We follow [7] to describe the general setting of a learning problem. Assume that $\mathbf{x}$ is an observation, $\mathbf{y}$ is the answer for $\mathbf{x}$, the feature function $\mathbf{f}(\mathbf{x}, \mathbf{y})$, which is used to describe the pair $(\mathbf{x}, \mathbf{y})$, is available, and $P(\mathbf{f}(\mathbf{x}, \mathbf{y}))$ is the unknown probability distribution of $\mathbf{f}(\mathbf{x}, \mathbf{y})$.

Definition. The goal of the general learning problem is to minimize the following risk function $R(\mathbf{w})$.

$$R(\mathbf{w}) = \int Q(\mathbf{f}(\mathbf{x}, \mathbf{y}); \mathbf{w}) \, \mathrm{d}P(\mathbf{f}(\mathbf{x}, \mathbf{y})) \tag{4}$$

where

$Q(\mathbf{f}(\mathbf{x}, \mathbf{y}); \mathbf{w})$ is a loss function on $\mathbf{f}(\mathbf{x}, \mathbf{y})$,

$\mathbf{w}$ is the parameter for $Q(\cdot)$

Our goal is to find the parameter $\mathbf{w}$ that can minimize the risk $R(\mathbf{w})$, i.e. to minimize the number of prediction mistakes on $\mathbf{f}(\mathbf{x}, \mathbf{y})$ from the distribution $P(\mathbf{f}(\mathbf{x}, \mathbf{y}))$, under the condition that $P(\mathbf{f}(\mathbf{x}, \mathbf{y}))$ is unknown but a set of samples $\{\mathbf{f}(\mathbf{x_i}, \mathbf{y_i})\}$ is given. Usually $\mathbf{w}$ parametrizes and enables a model to predict the answer for $\mathbf{x}$, thus the loss function $Q(\cdot)$ can be calculated by comparing the predicted answer with the true answer $\mathbf{y}$. That is why $Q(\cdot)$ is also parametrized by $\mathbf{w}$. However, in many real-world applications, there are uncertainties in choosing $\mathbf{f}(\mathbf{x}, \mathbf{y})$. In general, the feature selection problem affects the learning performance in two ways: linear combination of features and model selection. Linear combinations of features are widely used in many learning algorithms, e.g. support vector machines (SVMs), maximum entropy models, and their variants. In these algorithms, part of the models involves a linear combination of features $w_1 f_1 + w_2 f_2 + \cdots + w_n f_n$ where f_i represents the presence of feature i. Model selection is closely related to the over-fitting problem [7,8]. In general, a family of models $\mathcal{M}$ can be specified by its structure with a scale parameter k. Our goal is to select the model with the best k^* so as to minimize the generalization error, given that we can observe only the fitting error. There are many approaches to choosing k^*, including regularization, cross-validation, minimum description length (MDL) principle, Akaike's information criterion, structural risk minimization (SRM), etc..

The above issues become more complicated in text mining problems because of the rich feature sets available for text. These features may or may not be fully derived from the given text documents, e.g. part-of-speech (POS) information, prefix, suffix, lemma form, base type, and domain lexicons.

2.1 Discriminative Training and Log-Linear Models

Log-linear models are widely used in maximum entropy modelling. The parameters $\mathbf{w}$ are optimised such that $E_{\mathbf{w}}[f(\mathbf{x}, \mathbf{y})] = [f(\mathbf{x}, \mathbf{y})]$, where $E_{\mathbf{w}}[\cdot]$ is the expectation parametrized by $\mathbf{w}$ and $[\cdot]$ is the empirical expectation. The log-linear models can be shown to have maximum entropy over the training instances.

In discriminative training, a conditional distribution $P_{\mathbf{w}}(\mathbf{y}|\mathbf{x})$ is modelled directly. This gives more freedom to a model to fit the training instances $\{(\mathbf{x}, \mathbf{y})\}$ by adjusting its parameter $\mathbf{w}$. By comparing the following models

$$P_g(\mathbf{y}, \mathbf{x}; \mathbf{w}) = P_g(\mathbf{y}; \mathbf{w}) P_g(\mathbf{x}|\mathbf{y}; \mathbf{w}) \tag{5}$$

$$P_d(\mathbf{y}, \mathbf{x}; \mathbf{w}) = P_d(\mathbf{x}; \mathbf{w}') P_d(\mathbf{y}|\mathbf{x}; \mathbf{w}) \tag{6}$$

where (5) is used in generative training and (6) is used in discriminative training, it is easy to see that there are fewer constraints for discriminative training as the parameter $\mathbf{w}$ does not need to account for the marginal distribution $P_d(\mathbf{x}; \mathbf{w}')$, which can be parametrized by another parameter $\mathbf{w}'$.

Another advantage of modelling the conditional distribution directly is the possibility of adding a large number of possibly correlated and diversified features without degrading the performance as that of generative training. In this paper, we use conditional random fields (CRFs) to evaluate our automatic method

because it has been shown to perform better on text mining tasks than hidden Markov models (HMMs) which is its generative counterpart.

3 Feature Tree

3.1 Definitions

Definition. A feature a is an *atomic* feature if there exists no other features $f_1, f_2, \ldots, f_n$ such that $a = f_1 \wedge f_2 \wedge \cdots \wedge f_n$, i.e. it is not a conjunction of other features. A feature f is a *molecular* feature if it can be expressed as a conjunction of atomic features, i.e. there exists atomic features $a_1, a_2, \ldots, a_n$ such that $f = a_1 \wedge a_2 \wedge \cdots \wedge a_n$. An atomic feature is also a molecular feature [1]. Furthermore, if x has atomic features $a_1, a_2, \ldots, a_n$, we write $x = \{a_1, a_2, \ldots, a_n\}$. The final representation for x is a set of *molecular* features, denoted by $x \simeq \{f_1, f_2, \ldots, f_j\}$.

Definition. A feature $f = \{b_1, b_2, \ldots, b_m\}$ is said to be *activated* by an instance $x = \{a_1, a_2, \ldots, a_n\}$ if $f \subseteq x$.

3.2 Finding All Activated Features

In text mining, each $\mathbf{x}$ is usually a word with its associated information such as POS, lemma form, prefix, etc.. We can cast the feature generation problem as a frequent itemset mining problem. Then an instance is matched against the generated frequent itemset (the final feature space).

To avoid manual construction of conjunction of features, a frequent itemset mining algorithm can be applied on the atomic features of each instance by treating each atomic feature a_j as an item and each instance $\mathbf{x_i}$ as a transaction. The frequent itemsets $\mathcal{Z}$ will be served as our feature space.

Example 1 (Finding all activated features). Suppose we have an instance $\mathbf{x}$ that possesses the following atomic features: $\mathbf{x} = \{e, c, a, b\}$, and from the whole training set we have the following set of features $\mathcal{Z}$:

$$\mathcal{Z} = \{f_1 = \{c, a, b\}, \quad f_2 = \{a, c\}, \quad f_3 = \{a, e, b\}, \quad f_4 = \{c, d, e, a\}, \quad f_5 = \{a, d\}\}, \quad f_6 = \{b, e, c\}\}$$

The activated features for $\mathbf{x}$ are f_1, f_2, f_3 and f_6. In general, suppose each feature $f \in \mathcal{Z}$ has a length of $|f|$, and each instance $\mathbf{x}$ has a length of $|\mathbf{x}|$. When there are N_x instances and $|\mathcal{Z}|$ features in $\mathcal{Z}$, it takes $O(N_x|f||\mathbf{x}||\mathcal{Z}|)$ comparisons to get all activated features for all instances. Naively matching each instance $\mathbf{x}$ to $\mathcal{Z}$ is infeasible. To efficiently retrieve all activated features from $\mathcal{Z}$, we develop a prefix tree data structure, called *feature tree* (F-tree), to store $\mathcal{Z}$. An auxiliary array of linked list is used to store all nodes of the same atomic feature a_j to allow fast access. Algorithm 1 describes the details. We sort the atomic features a_j in each instance $\mathbf{x}$ in Example 1 according to lexicographic order. All the features are then further sorted according to their feature size $|f|$.

[1] A molecule can show chemical properties different from its constituent atoms. A close analogy to many text mining tasks.

Algorithm 1.

Input: A list of features $\mathcal{Z}$. Each feature $f_i = (a_1, a_2, \ldots, a_n) \in \mathcal{Z}$ is sorted according to certain order $\mathcal{R}$
Output: F-tree T, and array of linked list L
Procedure: F-TREE-CONSTRUCT

1: Create the root node, T, for F-tree
2: Sort the features in $\mathcal{Z}$ in ascending order of feature size
3: **for** $\mathbf{f_i} = (a_1, a_2, \ldots, a_n) \in \mathcal{Z}$ **do**
4: $\theta \leftarrow T$
5: $k = 1$
6: **while** $\exists c$ s.t. $c.id = a_k$ and $c.parent = \theta$ **do**
7: $\theta \leftarrow c$
8: $k \leftarrow k + 1$
9: **end while**
10: Create a node e and $e.id \leftarrow a_k$
11: $e.parent \leftarrow \theta$
12: $L[a_k].append(e)$
13: **if** $k = n$ **then**
14: $e.fid \leftarrow i$
15: F-TREE-APRIORI($\mathbf{f_i}, T, e$)
16: **end if**
17: **end for**
18: **return** T, L

Input: An instance $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ sorted according to an arbitrary order $\mathcal{R}$. A F-tree T, and an array of linked list L for T.
Output: A set of activated features Ω
Procedure: F-TREE-RETRIEVE

1: $T.mark \leftarrow$ **true**
2: **for** $i = 1$ to n **do**
3: $\theta \leftarrow L[x_i]$
4: **while** $\theta \neq$ **NULL do**
5: **if** $\theta.parent.mark =$ **true then**
6: $\theta.mark \leftarrow$ **true**
7: **if** $\theta.fid \neq$ **NULL then**
8: $\Omega \cup \{\theta\}$
9: F-TREE-APRIORI-COUNTS(θ) *//update counts of activated children in Apriori lattice*
10: **end if**
11: **end if**
12: $\theta \leftarrow \theta.next$
13: **end while**
14: **end for**
15: Ω = F-TREE-APRIORI-REMOVE(Ω) *//for abstraction with Apriori lattice*
16: **return** Ω

In Algorithm 1, we only need to mark nodes that have their parents marked. If a node is not marked but its parent is marked, then all the molecular features represented by this node (if its fid is non-null) and downward will not be activated. This can be easily proved by contradiction. Suppose a molecular feature represented by a downward node is activated. This implies that we will need to mark the current node later. However, because the atomic features to be marked is sorted, this node will never be marked.

Figure 1 shows the complete F-Tree T for $\mathcal{Z}$ in Example 1. We can use T to check which molecular features can be activated by $\mathbf{x}$. Algorithm 1 accomplishes this by making use of an array of linked list L for fast access to the list of atomic feature nodes. When marking nodes, we need to check if the current node is marked and has a non-null feature ID (fid). If we find such a node, the corresponding fid will be stored and returned. We first mark the nodes

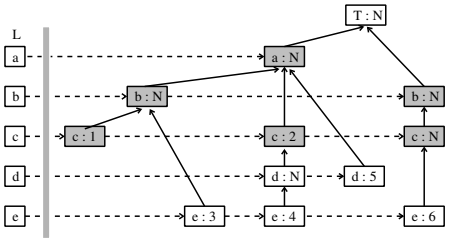


Fig. 1. The marked nodes after marking node c in Example 1

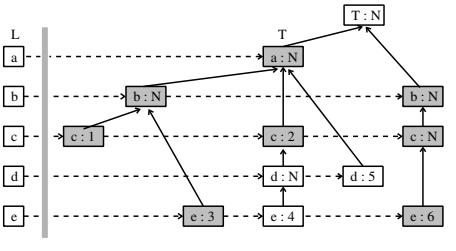


Fig. 2. All the marked nodes for Example 1

representing the first atomic feature a of $\mathbf{x}$ as **true**. We do the same for each of the sorted atomic features in $\mathbf{x}$. After marking the third atomic feature c, the resulting F-tree is shown in Figure 1. At this point, we have retrieved f_1 and f_2, i.e. $\Omega = \{1, 2\}$. The complete marked tree is shown in Figure 3.2. The middle node of e is not marked because its parent is not marked. Finally, $\Omega = \{1, 2, 3, 6\}$. Our experimental results (described in Section 4.1) show that the memory requirement can be reduced by half and the performance is comparable to that of the state-of-the-art systems that use manual construction of features.

3.3 Removing Redundant Features

In order to further reduce memory usage, we propose the following principle to remove redundant features.

Abstraction Principle. If a feature f is activated by $\mathbf{x}$ but all of its immediate supersets in the set of features are also activated by $\mathbf{x}$, i.e. $G : \{g \in \mathcal{Z}, f \subseteq g \text{ and } |g| = |f|+1\}$ and every $g \in G$ is activated by $\mathbf{x}$, then f is abstracted away from the representation for x.

To illustrate this principle, we consider the following example that shows an Apriori lattice.

Example 2 (Abstract away redundant features). Suppose from the whole training set we have the following list of features:

$$\mathcal{Z} = \{\ f_1 = \{a\},\quad f_2 = \{b\},\quad f_3 = \{c\},\quad f_4 = \{d\},\quad f_5 = \{a, b\},\quad f_6 = \{a, c\},$$
$$f_7 = \{b, c\},\quad f_8 = \{c, d\},\quad f_9 = \{b, d\},\quad f_{10} = \{a, b, c\},\quad f_{11} = \{b, c, d\}\}$$

These features can be represented structurally with an Apriori lattice as in Figure 3. Suppose further that we have $\mathbf{x} = \{a, b, c\}$. The activated features are shown in Figure 4. At this point, we need to choose among the activated features to represent $\mathbf{x}$. Suppose we choose abc as one of the features to represent $\mathbf{x}$. For the remaining features, we can see clearly their relationships from the lattice . For feature bc, from the training set we know that it can be used to generate the feature bcd. However, bcd is not activated because cd and bd is not activated. Therefore, bc may contain some information that may not be covered by abc so we also choose it to represent $\mathbf{x}$. The same argument also applies for feature b and c.

For feature ab, it can only be used to generate higher-level feature abc but abc is already chosen to represent $\mathbf{x}$. From this lattice structure (which is generated by the training set), we can expect that the information covered by ab should also be covered by abc. Therefore we do not choose ab to represent $\mathbf{x}$. ac and a are not chosen because of the same argument. Finally, instance $\mathbf{x}$ is represented by features abc, bc, b and c, i.e. $\mathbf{x} \simeq \{abc, bc, b, c\}$.

To make a contrast, we also consider $\mathbf{u} = \{a\}$ whose representation is also a, i.e. $\mathbf{u} \simeq \{a\}$. $\mathbf{u}$ and $\mathbf{x}$ share common atomic features a, and that $\mathbf{u}$ has a as its representation, but the representation for $\mathbf{x}$ does not necessarily contain a. To further illustrate our idea, let us consider a different set of features where

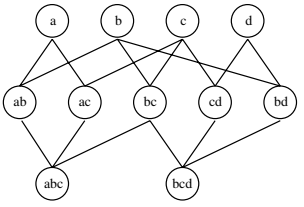


Fig. 3. Apriori lattice for Example 2

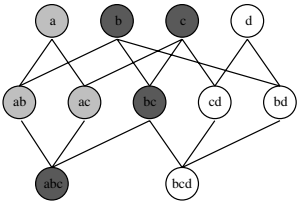


Fig. 4. Activated features for $\mathbf{x} = (a, b, c)$ for Example 2

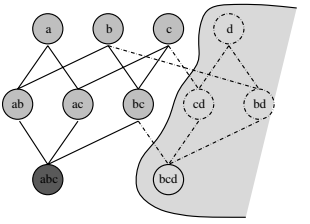


Fig. 5. When node d is absent, abc is chosen

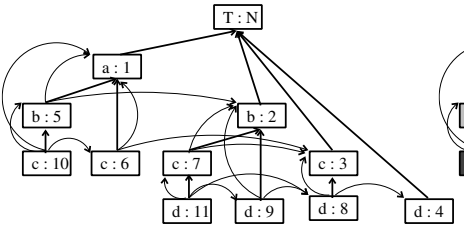


Fig. 6. The F-tree for Example 2 with the Apriori lattice incorporated

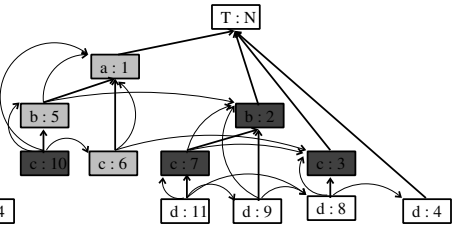


Fig. 7. Activated features and the final representation for Example 2

feature d and its associated features are removed, as shown in Figure 5. In this case, $\mathbf{x} \simeq \{abc\}$. This also shows the interactive characteristics between an instance and the whole feature space in our approach. To implement the above feature abstraction, we need an Apriori lattice to show the relationships between features. We add a procedure F-TREE-APRIORI, which is presented in Algorithm 2, to F-TREE-CONSTRUCT (in Algorithm 1) to create the Apriori lattice.

Figure 6 shows the complete F-tree with the Apriori lattice incorporated. The solid lines point to the original parents in F-tree while the dotted curved lines point to the parents in the Apriori lattice. Starting from Figure 4, we call the procedures F-TREE-APRIORI-COUNTS and F-TREE-APRIORI-REMOVE defined in Algorithm 2 when we retrieve the features from F-tree by procedure F-TREE-RETRIEVE (Algorithm 1). These two procedures check the number of activated children for each activated node. If all children are activated, then according to our abstraction principle it is not included in the representation. the instance being considered. Figure 7 shows the activated features and the features chosen to represent $\mathbf{x}$, i.e. $\mathbf{x} \simeq \{b, c, bc, abc\}$. For example, in Figure 4, the node a has two children in the Apriori lattice. They are indicated by the two dotted curved arrows pointing to $(a : 1)$ in Figure 7. Each of these arrows comes from an activated node. Therefore feature a is not chosen to represent $\mathbf{x}$. As another example, the node $(c : 3)$ has one of its children $(d : 8)$ not activated. Thus feature c is chosen to represent $\mathbf{x}$.

Algorithm 2.

Input: A feature $\mathbf{f} = (a_1, a_2, \ldots, a_n)$, a F-tree T, and the node e for representing $\mathbf{f}$ in T.
Procedure: F-TREE-APRIORI

1: $S \leftarrow$ all subsets of size $n-1$ of $\mathbf{f}$
2: **for** $\mathbf{s} = (s_1, s_2, \ldots, s_{n-1}) \in S$ **do**
3: $\theta \leftarrow T$
4: $k = 1$
5: *//Find the node that represents* $\mathbf{s}$
6: **while** $\exists c$ s.t. $c.id = s_k$ and $c.parent = \theta$ **do**
7: $\theta \leftarrow c$
8: $k \leftarrow k + 1$
9: **end while**
10: *//store the Apriori parents in the set* $e.AParents$
11: $e.AParents \cup \{\theta\}$
12: $\theta.nChild \leftarrow \theta.nChild + 1$
13: **end for**

Input: A node e that represents an activated feature
Procedure: F-TREE-APRIORI-COUNTS

1: *//update the counts of activated children*
2: **for** $p \in e.AParents$ **do**
3: $p.ActChild \leftarrow p.ActChild + 1$
4: **end for**

Input: A set of activated feature Ω
Output: A set of feature Ψ after abstraction
Procedure: F-TREE-APRIORI-REMOVE

1: *//remove features if children are all activated*
2: **for** $e \in \Omega$ **do**
3: **if** $e.nChild = e.ActChild$ **then**
4: $\Psi \cup \{e\}$
5: **end if**
6: **end for**
7: **return** Ψ

4 Experimental Results

Since the conditional random field (CRF) model [9] has shown promising performance in many text mining tasks, we use it to conduct large scale experiments on three publicly-available datasets and compare with other published text mining results on them. In all experiments, we used our automatic method to generate and store features. Then we find the representation for each instance.

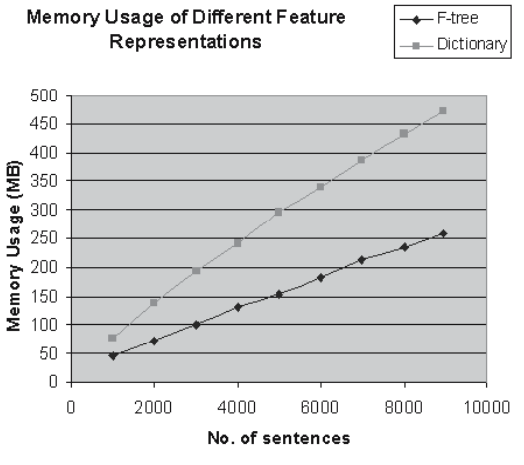


Fig. 8. Memory usage of F-TREE on CoNLL-2000 dataset

4.1 Memory Usage

We compare the memory usage of our F-tree with that of a dictionary which is used in storing the feature space $\mathcal{Z}$ and for matching each $\mathbf{x}$. The experiment was done on a public dataset CoNLL-2000. The details of this dataset will be described in Section 4.2. We tested the memory requirement using different sizes (see Figure 8) and found that F-tree required only approximately half of the memory as that of a dictionary on this dataset.

4.2 Accuracy

Noun-phrase Chunking. We use the WSJ corpus from the CoNLL-2000 shared task to conduct noun-phrase (NP) chunking experiments. Our atomic features and performance, along with other published results, are shown in Tables 1 and 2. Sha and Pereira [5] used a manual template with a second-order CRF. Their results are on par with the best results (Kudo and Matsumoto [10] : 94.39% F1). McCallum [6] used a feature induction method on a first-order CRF to induce features to be used in the CRF. Their feature induction method is specific to CRFs. In both of the work, they did not report on the precision (P) and recall (R), so we can list only the $F1$-measures in the table. Our model is a first-order CRF (same as McCallum). Our precision is 94.65% and recall is 94.37%. Our automatic method gets a better $F1$ than the above methods. It also outperforms Kudo and Matsumoto [10] (94.39% F1) that has the best result in the CoNLL-2000 shared task. We believe that although Sha and Pereira [5] used a second-order CRF, there may not be enough training data to train it. The feature induction method [6] uses a pseudo-likelihood to approximate the effect of adding a new feature. Therefore the new feature is not necessarily the best feature to be induced.

Features	0th order	1st order
Word	(w_{i-2}), (w_{i-1}), (w_i), (w_{i+1}), (w_{i+2})	(w_{i-1}), (w_i), (w_{i+1})
POS	(p_{i-2}), (p_{i-1}), (p_i), (p_{i+1}), (p_{i+2})	(p_{i-1}), (p_i), (p_{i+1})

Table 1. The atomic features used in the noun-phrase chunking task

Experiments	F_1
Automatic: After abstraction	94.51
Automatic: All features	94.45
Manual: All features [5]	94.38
Manual: Supported features [5]	94.19
Feature Induction [6]	93.96

Table 2. Results of the noun-phrase chunking task

Experiments	F_1
Automatic: After abstraction	82.31
Automatic: All features	82.04
Manual: Single word features [6]	80.00
Manual: Hand-engineered features [6]	73.30

Table 3. Results of the general named entity recognition task

General Named Entity Recognition. In this experiment, we used the dataset in the CoNLL-2003 shared task, which was used in the feature induction experiments in [6]. McCallum conducted an experiment with manually-prepared

templates with lexicons, and another one with single word features only. Surprisingly, the manually-prepared templates with lexicons got only 73.3% F1 while in the experiment using simple single word features, the performance was 80% F1. Our automatic method with all features got 82.04% F1. After abstraction, the performance increased slightly to 82.31% F1. The results are listed in Table 3. Both are better than those of the experiment using manually-prepared templates with lexicons and the experiment using the single word features.

Biomedical Named Entity Recognition. In this experiment, we use the GENIA corpus from the COLING workshop (BioNLP/NLPBA 2004). We tried to reproduce the features described in [1] and used them only as atomic features. The POS features are added by the GENIA tagger. Two lexicons for cell lines and genes are drawn from two online public databases: Cell Line Database and BBID. They are listed in Tables 4 and 5.

There are many different types of features involved in this experiments. Thus we used fewer number of first-order features to reduce the computation cost. The final results are listed in Table 6. Settles used two conditional random fields with different sets of features and applied a χ^2 test to select the words for their keyword lexicons. One CRF only used a set of orthographic features and the other used orthographic features and lexicons. The performance of the orthographic features is better than that of complete features. We used our automatic method on the orthographic and lexicon features, resulting in a higher F1 of 71.39 and 71.20. The performance also beats a CRF model used in the NLPBA-2004 shared task.

Features	0th order	1st order
Word (w)	(w_{i-2}), (w_{i-1}), (w_i), (w_{i+1}), (w_{i+2})	(w_i)
POS (p)	(p_{i-2}), (p_{i-1}), (p_i), (p_{i+1}), (p_{i+2})	(p_i)
Word Class (k)	(k_i)	
Brief Word Class (m)	(m_{i-1}), (m_i) (m_{i+1})	(m_i)
Digit (d)	(d_{i-1}), (d_i)	(d_{i+1})

Table 4. The atomic features used in the biomedical named entity recognition task. Only 3 first-order features are used.

Features	0th order
Punctu (q) -ation	(q_{i-1}),, (q_i), (q_{i+1})
Dash (t)	(t_{i-1}), (t_i), (t_{i+1})
Capital Letter (c)	(c_{i-1}), (c_i) (c_{i+1})
Prefix (r)	(r_i)
Suffix (u)	(u_i)
Word (ℓ) Length	(ℓ_i)
Cell line lexicon (v)	(v_i)
Gene lexicon (w)	(w_i)

Table 5. The atomic features used in the biomedical named entity recognition task

Experiments	F_1
Automatic: After abstraction	71.39
Automatic: All features	71.20
Manual: Ortho. Features [1]	69.80
Manual: Ortho. Features + Lexicons [1]	69.50

Table 6. Comparisons of automatic methods with other published results on biomedical named entity recognition task

5 Conclusions and Future Work

We propose an algorithmic framework F-tree for automatically generating and storing conjunction of features for text mining tasks. It only requires the user

to define a set of *atomic* feature templates. An instance's activated features in a feature space can be found by a *one-vs-all* matching process. Our experimental results show that our automatic method reduces the memory usage by half and achieves state-of-the-art performance which is achieved by manual construction of features. To further improve the performance, we propose an *abstraction* principle to remove redundant features, which leads to slight improvements on the top performance. We believe that our abstraction principle can be more aggressive and further improvements can be made by employing more refined criteria.

References

1. Settles, B.: Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets. In: Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA), pp. 104–107 (2004)
2. Kudo, T.: CRF++: Yet another CRF toolkit (2006), `http://chasen.org/~taku/software/CRF++`
3. Phan, X.H., Nguyen, L.M., Nguyen, C.T.: Flexcrfs: Flexible conditional random field toolkit (2005), `http://flexcrfs.sourceforge.net`
4. McCallum, A.K.: Mallet: A machine learning for language toolkit (2002), `http://mallet.cs.umass.edu`
5. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: NAACL 2003: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pp. 134–141 (2003)
6. McCallum, A., Li, W.: Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, Morristown, NJ, USA, pp. 188–191. Association for Computational Linguistics (2003)
7. Vapnik, V.N.: The nature of statistical learning theory. Springer, New York (1995)
8. Xu, L.: A trend on regularization and model selection in statistical learning: A bayesian ying yang learning perspective. Challenges for Computational Intelligence, 365–406 (2007)
9. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. 18th International Conf. on Machine Learning, pp. 282–289 (2001)
10. Kudo, T., Matsumoto, Y.: Chunking with support vector machines. In: NAACL 2001: Second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies 2001, Morristown, NJ, USA, pp. 1–8. Association for Computational Linguistics (2001)

Robust Graph Hyperparameter Learning for Graph Based Semi-supervised Classification

Krikamol Muandet[1], Sanparith Marukatat[2], and Cholwich Nattee[1]

[1] School of Information, Computer and Communication Technology
Sirindhorn International Institute of Technology
Thammasat University, 131 Tiwanont Rd. Bangkadi
Pathum Thani, Thailand 12000
[2] National Electronic and Computer Technology Center
National Science and Technology Development Agency
111 Thailand Science Park
Pathum Thani, Thailand 12120

Abstract. Graph-based semi-supervised learning has attracted much attention in recent years. Many successful methods rely on graph structure to propagate labels from labeled data to unlabeled data. Although graph structure affects the performance of the system, only few works address its construction problem. In this work, the graph structure is constructed by adjusting the hyperparameter controlling the connection weights between nodes. The idea is to learn the hyperparameters for graphs that yields low leave-one-out prediction error on labeled data while retaining high stability of the prediction on unlabeled data. The problem is solved efficiently using the gradient-based optimization technique. Experimental results indicate that the proposed technique yields improvement of generalization error as well as stability of the labeling function.

Keywords: Machine Learning, Graph-Based Semi-Supervised Learning, Stability, Hyperparameter Learning, Model Selection.

1 Introduction

Graph-based semi-supervised learning techniques have attracted much attention in recent years. The general idea of these methods is based on building a graph whose nodes are data points connected with edges that encode similarities between them. The labels of unlabeled data are learned on this graph in such a way that nearby data points will have similar labels. Examples of graph-based semi-supervised algorithms include label propagation [1], graph mincuts [2], randomized mincuts [3], and regularization on graphs [4,5].

Although the graph structure is shown to have much influence in graph-based semi-supervised learning [5], few methods are proposed to deal with its construction problem. In [6] the hyperparameters of the graph are learned using the evidence maximization. Carriera-Perpinan and Zemel [7] propose to construct

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 98–109, 2009.
© Springer-Verlag Berlin Heidelberg 2009

an ensemble of graphs by combining multiple minimum spanning trees, formed from perturbed version of the data. Wang and Zhange [8] propose a method that is similar to locally linear embedding(LLE) to learn the graph weights by minimizing the reconstruction error of the data points from its neighbors. The most influencial works are the algorithms introduced in [9] and [10]. In [9], the authors optimize the hyperparameters using average label entropy on unlabeled data points as a criterion, i.e., they select hyperparameters which yield the most confident labeling on unlabeled data. It is shown by experimental results in [10] that minimizing the entropy on unlabeled data is insufficient. Zhang and Lee instead find the parameters by minimizing leave-one-out prediction error on labeled data.

While there are several strategies to learn graph structure, propagation algorithms on the graph are quite similar. Indeed, most of graph-based semi-supervised techniques rely on an objective function which is a tradeoff between prediction error on labeled data and smoothness of label transition. In [5], the authors have shown that the smoothness term makes the label assignment a "stable" algorithm, thus bounds over the generalization error can be computed. This suggests that a good hyperparameter should not only minimize the leave-one-out prediction error as in [10], but also maximize the stability of the labeling function on the resulting graph.

In this work, we extend the techniques proposed in [9] and [10] by directly incorporating a regularization term which measures the stability of the labeling function on graphs. Involving this regularizer, the hyperparameters of a graph are learned using the gradient-based optimization technique. Using the stability measure, we can achieve a stable labeling function as a consequent of this graph learning method.

2 Graph-Based Semi-supervised Learning

Given a data set X consisting of l labeled examples $x_i \in L = \{x_1, x_2, ..., x_l\}$ and u unlabeled examples $x_i \in U = \{x_{l+1}, x_{l+2}, ..., x_n\}$, where L and U denote sets of labeled and unlabeled examples, respectively. Let $n = l + u$ be the total number of examples and m be the dimensionality of the input data. Associated with each labeled example is labels $y_i \in Y$. In this work, we consider only a binary classification problem in which $Y = \{0, 1\}$.

2.1 Graph-Based Semi-supervised Classification

A general task in graph-based semi-supervised classification is to assign labels to unlabeled nodes $x_i \in U$ through a graph G. The nodes represent both labeled and unlabeled data points. The symmetric weight matrix W which encodes the similarities between nodes is typically computed as follow:

$$w_{ij} = \exp(-\sum_{d=1}^{m}(x_{i,d} - x_{j,d})^2/\sigma_d^2) \ , \tag{1}$$

where $x_{i,d}$ is the d-th component of $x_i \in \mathbb{R}^m$ and σ_d is the bandwidth hyper-parameters for the dimension d. The transition probability matrix between two nodes can be computed by $P = D^{-1}W$. The matrix D is a diagonal matrix whose $D_{ii} = \sum_{j=1}^{n} w_{ij}$ is a degree of node i.

A general strategy is to find a real-valued function f on graph and then assign labels based on f. We want f to take values y_i on labeled data points and varies smoothly on unlabeled data points according to the weight matrix W. This criterion leads to the following choice of consistency of f on graphs [9]:

$$E(f) = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(f(i) - f(j))^2 \tag{2}$$

where $f(i)$ and $f(j)$ are the function values at node i and j, respectively. To find the solution of the above equation, it is convenient to split the matrices W and P into four parts, starting with labeled data points followed by unlabeled data points:

$$W = \begin{pmatrix} W_{LL} & W_{LU} \\ W_{UL} & W_{UU} \end{pmatrix}, P = \begin{pmatrix} P_{LL} & P_{LU} \\ P_{UL} & P_{UU} \end{pmatrix} \ .$$

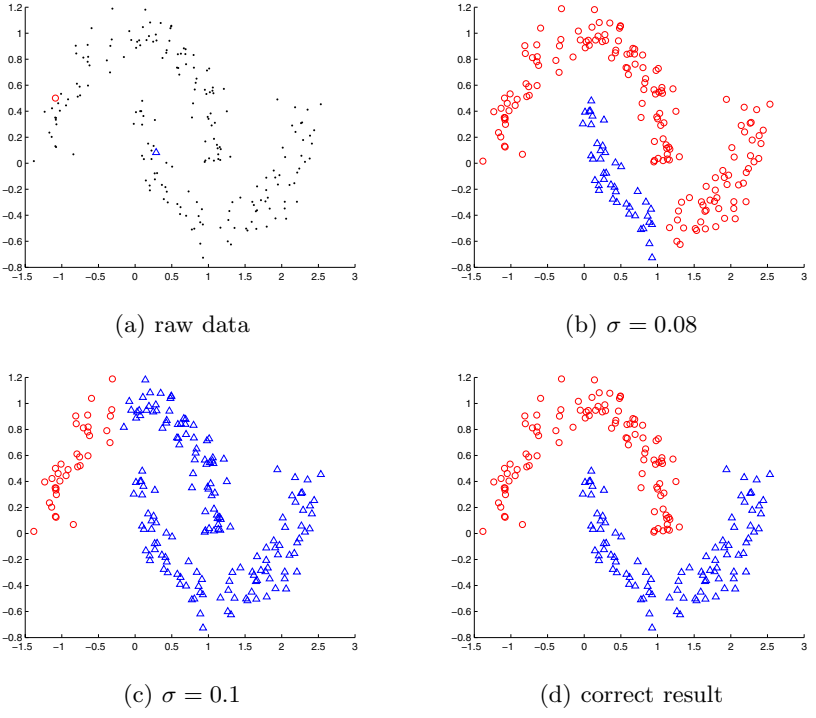
It is not difficult to show that the function that yields a minimum value of (2) is given by:

$$f_U = (D_{UU} - W_{UU})^{-1} W_{UL} f_L = (I - P_{UU})^{-1} P_{UL} f_L \ . \tag{3}$$

This function is called the *soft-label* function since its value does not directly specify the class to which the example belongs, but it gives a possibility of being in each class. The most obvious method to transform soft-label to hard-label is by thresholding, e.g., classify x as being in class 1 if $f(x) > 0.5$, and class 0 otherwise. This method generally works well when the classes are well-separated. However, this is not the case in many practical applications. In such cases, using simple thresholding may result in an unbalanced classification.

Another method to transform the soft-label to hard-label is *Class Mass Normalization*(CMN) introduced in [9]. The class distribution of the data is adjusted to match the class priors, that can be obtained from the labeled examples. For example, if the prior class proportion of class 1 and 0 is p and $1 - p$, then an unlabeled example x is classified as class 1 iff $p \cdot (f(x)/\sum_i f(x_i)) > (1-p)((1-f(x))/\sum_i (1-f(x_i)))$. This method works well when we have sufficient labeled examples to determine the class prior that accurately represents the true class distribution.

As mentioned previously, graph structures do influence the results of classification. Figure 1 illustrates examples of such a situation, which shows two-dimensional data set called `twomoon`. In Fig.1(a) two labeled examples, circle and triangle, are drawn along with unlabeled examples depicted as solid dots. As we can see in Fig.1(b) and (c), a small change in the value of σ used to construct graphs results in significantly different classification results. Moreover the optimal σ, which yields a correct result as shown in Fig.1(d), is difficult to find in practice. Consequently, we need an efficient graph learning algorithm to solve this problem as it plays vital role in graph-based semi-supervised classification.



(a) raw data (b) $\sigma = 0.08$

(c) $\sigma = 0.1$ (d) correct result

Fig. 1. The results of classification using different values of σ to construct graphs; (a) the raw data with 2 labeled examples (b), the result when $\sigma = 0.08$, (c) the result when $\sigma = 0.1$, (d) the result of correct classification

2.2 Graph Learning Algorithms

As mentioned previously, this work extends two previous works namely Minimum Entropy(MinEnt) [9] and Leave-one-out Hyperparameter Learning(LOOHL) [10]. This section gives a brief review of these two techniques successively.

In MinEnt, the weight matrix is obtained by learning all σ_d in (1) from both labeled and unlabeled data. An *average label entropy*, $H(f)$, is used as a heuristic criterion for parameter learning, which is defined as

$$H(f) = \frac{1}{u} \sum_{i=l+1}^{n} H_i(f(i)) \ , \tag{4}$$

where $H_i(f(i)) = -f(i) \log_2 f(i) - (1 - f(i)) \log_2 (1 - f(i))$ is the entropy of the soft-label at the individual unlabeled data point i. The concept of this algorithm is that a good weight W should result in a confident labeling, i.e., the average label entropy is low when soft-label at each unlabeled data point is close to the boundary of the labeling function.

Instead of considering label entropy at unlabeled data, the LOOHL learns the hyperparameters σ_d of the graph by minimizing the leave-one-out prediction error on all labeled examples. The objective function is:

$$Q = \sum_{t=1}^{l} h_t(f_{U,1}^t) \ , \tag{5}$$

where $h_t(x)$ is the cost function for labeled example x that penalizes the difference between the predicted soft-label and the true label of the instance. An efficient algorithm to gradient computation is proposed based on matrix inversion lemma and careful precomputation. Both MinEnt and LOOHL also mention a case of degenerative graphs in which σ_d grow either too large or too small. Thus a simple regularizer is used to prevent this problem. Moreover, P is replaced by $\tilde{P} = \varepsilon\mathcal{U} + (1-\varepsilon)P$, where $\mathcal{U}_{ij} = 1/n$, to prevent the numerical problem.

3 Robust Graph Hyperparameter Learning

In this work, we propose a hyperparameter learning method which considers the leave-one-out prediction error on labeled examples and the stability cost of the labeling function on unlabeled examples. Using the result of (3), at each iteration $t = 1, .., l$ of leave-one-out, the soft-label function can be determined by $f_U^t = (I - \tilde{P}_{UU}^t)^{-1}\tilde{P}_{UL}^t f_L^t$ where $f_L^t = (f_1, .., f_{t-1}, f_{t+1}, .., f_l)^{\mathrm{T}}$. The modified transition matrix is then defined as:

$$P_{UU}^t = \begin{pmatrix} p_{t,t} & p_{t,l+1} & \cdots & p_{t,n} \\ p_{l+1,t} & & & \\ \vdots & & P_{UU} & \\ p_{n,t} & & & \end{pmatrix}, P_{UL}^t = \begin{pmatrix} p_{t,1} & \cdots & p_{t,t-1} & p_{t,t+1} & \cdots & p_{t,l} \\ p_{l+1,1} & \cdots & p_{l+1,t-1} & p_{l+1,t+1} & \cdots & p_{l+1,l} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ p_{n,1} & \cdots & p_{n,t-1} & p_{n,t+1} & \cdots & p_{n,l} \end{pmatrix}$$

The first row and column in P_{UU}^t and the missing column in P_{UL}^t correspond to a labeled example x_t used to evaluate prediction cost $h_t(x_t)$. Let e_k be a vector whose element is 1 at k-th position and 0 elsewhere. The soft label of this example is $e_1^{\mathrm{T}} f_U^t$, the first element of f_U^t. Let $f_{U,k}^t$ for $1 \le k \le u+1$ define the soft-label of x_t together with other unlabeled examples. Then the objective function to be minimized can be written as:

$$R = \gamma \sum_{t=1}^{l} h_t(f_{U,1}^t) + (1-\gamma) \sum_{k=2}^{u+1} s(f_{U,k}) \tag{6}$$

where γ balances accuracy and stability of the soft-label function. The $f_{U,k} = (f_{U,k}^1, f_{U,k}^2, .., f_{U,k}^l)^{\mathrm{T}}$ is the vector whose element is the soft-label of k-th unlabeled example at each $t = 1, .., l$. The cost function $h_t(f_{U,1}^t)$ is defined as in (5). Possible choices for $h_t(x_t)$, if $y_t = 1$, are $1 - x_t$, $(1-x_t)^2$, and $-\log x_t$. If $y_t = 0$, the cost function can be x, x^a, and $a^{x_t - 1}$, for example.

In the second term of (6), the cost function $s(f_{U,k})$ measures the variation of the soft-label over elements of $f_{U,k}$. Thus the sum of this quantity over all

unlabeled examples can be used to determine how stable the labeling function is. A possible choice of $s(f_{U,k})$ is the variance function. Although this function captures the variation of soft-label, it does not take into account the confidence of labeling function. For example, if most of soft-label values lies near 0.5, we assign labels to unlabeled data with low confidence. Instead, the stability cost function used in this work is:

$$s(f_{U,k}) = -\bar{f}_{U,k} \log_2 \bar{f}_{U,k} - (1 - \bar{f}_{U,k}) \log_2 (1 - \bar{f}_{U,k}) \tag{7}$$

which is the entropy of an average of all elements of $f_{U,k}$. This function is chosen because it penalizes the high variation as well as the deviation from the function boundary. By minimizing this cost, labels are assigned with high confidence using the stable function.

To minimize R, we use the gradient-based optimization technique. The gradient is:

$$\begin{aligned} \partial R/\partial \sigma_d = \gamma & \left(\sum_{t=1}^{l} h'_t(f^t_{U,1}) e_1^{\mathrm{T}} \left(\partial f^t_U / \partial \sigma_d \right) \right) \\ & + (1-\gamma) \left(\sum_{k=2}^{u+1} s'(\bar{f}_{U,k}) \sum_{t=1}^{l} e_k^{\mathrm{T}} \left(\partial f^t_U / \partial \sigma_d \right) \right) \end{aligned} \tag{8}$$

where $\partial f^t_U/\partial \sigma_d = (I - \tilde{P}^t_{UU})^{-1} (\partial \tilde{P}^t_{UU}/\partial \sigma_d \cdot f^t_U + \partial \tilde{P}^t_{UL}/\partial \sigma_d \cdot f^t_L)$ using matrix property $dX^{-1} = -X^{-1}(dX)X^{-1}$. The $s'(\bar{f}_{U,k}) = \log_2((1 - \bar{f}_{U,k})/\bar{f}_{U,k})$. Denoting $(\beta^t)^{\mathrm{T}} = (h'_t(f^t_{U,1}) e_1^{\mathrm{T}} (I - \tilde{P}^t_{UU})^{-1})$ and $(\xi^t_k)^{\mathrm{T}} = e_k^{\mathrm{T}} (I - \tilde{P}^t_{UU})^{-1}$ and noting that $\tilde{P} = \varepsilon \mathcal{U} + (1 - \varepsilon) P$. With careful re-arrangement and substitution, the gradient can be written as

$$\begin{aligned} \partial R/\partial \sigma_d = \mu & \sum_{t=1}^{l} \left(\beta^t\right)^{\mathrm{T}} \left(\frac{\partial P^t_{UU}}{\partial \sigma_d} \cdot f^t_U + \frac{\partial P^t_{UL}}{\partial \sigma_d} \cdot f^t_L \right) \\ & + \mu' \left(\sum_{k=2}^{u+1} s'(\bar{f}_{U,k}) \sum_{t=1}^{l} (\xi^t_k)^{\mathrm{T}} \left(\frac{\partial P^t_{UU}}{\partial \sigma_d} \cdot f^t_U + \frac{\partial P^t_{UL}}{\partial \sigma_d} \cdot f^t_L \right) \right) \end{aligned} \tag{9}$$

where $\mu = \gamma(1 - \varepsilon)$ and $\mu' = (1 - \gamma)(1 - \varepsilon)$. From the relation that $P(i,j) = w(i,j)/\sum_{j=1}^{n} w(i,j)$, the partial derivative of transition probability $P(i,j)$ can be computed by

$$\frac{\partial P(i,j)}{\partial \sigma_d} = (\theta_i)^{-1} \left(\frac{\partial w(i,j)}{\partial \sigma_d} - P(i,j) \cdot \sum_{j=1}^{n} \frac{\partial w(i,j)}{\partial \sigma_d} \right) \ , \tag{10}$$

where $\theta_i = \sum_j w(i,j)$ and $\partial w(i,j)/\partial \sigma_d = 2w(i,j)(x_{i,d} - x_{j,d})^2/\sigma_d^3$ by (1). Followed from [10], the computational complexity of the algorithm is very expensive. A possible way to reduce the cost is by applying matrix inversion lemma in computation of $(I - \tilde{P}^t_{UU})^{-1}$ for each iteration t. Moreover, the computational

cost can be further reduced by using an efficient precomputation. Spliting the gradient computation into two parts, $\partial R/\partial\sigma_d = \mu \cdot \partial H/\partial\sigma_d + \mu' \cdot \partial S/\partial\sigma_d$ and substituting (10) into (9), each gradient term can be derived separately as:

$$\begin{aligned}\frac{\partial H}{\partial \sigma_d} &= \sum_{t=1}^{l}\sum_{i=1}^{u+1}\frac{\beta_i^t}{\theta_i^t}\left(\sum_{j=1}^{u+1}\frac{\partial w_{UU}^t(i,j)}{\partial\sigma_d}f_{U,j}^t + \sum_{j=1}^{l-1}\frac{\partial w_{UL}^t(i,j)}{\partial\sigma_d}f_{L,j}^t\right.\\ &\quad\left.-\sum_{k=1}^{n}\frac{\partial w^t(i,k)}{\partial\sigma_d}\left(\sum_{j=1}^{u+1}P_{UU}^t(i,j)\cdot f_{U,j}^t + \sum_{j=1}^{l-1}P_{UL}^t(i,j)\cdot f_{L,j}^t\right)\right)\\ &= \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij}\frac{\partial w(i,j)}{\partial\sigma_d} \qquad (11)\\ \frac{\partial S}{\partial \sigma_d} &= \sum_{k=2}^{u+1}s'(\bar{f}_{U,k})\sum_{t=1}^{l}\sum_{i=1}^{u+1}\frac{\xi_{k,i}^t}{\theta_i^t}\left(\sum_{j=1}^{u+1}\frac{\partial w_{UU}^t(i,j)}{\partial\sigma_d}f_{U,j}^t + \sum_{j=1}^{l-1}\frac{\partial w_{UL}^t(i,j)}{\partial\sigma_d}f_{L,j}^t\right.\\ &\quad\left.-\sum_{p=1}^{n}\frac{\partial w^t(i,p)}{\partial\sigma_d}\left(\sum_{j=1}^{u+1}P_{UU}^t(i,j)\cdot f_{U,j}^t + \sum_{j=1}^{l-1}P_{UL}^t(i,j)\cdot f_{L,j}^t\right)\right)\\ &= \sum_{k=2}^{u+1}s'(\bar{f}_{U,k})\sum_{i=1}^{n}\sum_{j=1}^{n}\rho_{ij}^k\frac{\partial w(i,j)}{\partial\sigma_d}\ . \qquad (12)\end{aligned}$$

Let $\delta(\cdot)$ be *Kronecker* delta. The ρ_{ij}^k can be computed by the following formula:

$$\rho_{ij}^k = \begin{cases}(\theta_i)^{-1}\sum_{t=1}^{l}\xi_{k,i-l+1}^t(f_{U,j-l+1}^t - \sum_{k=l+1}^{n}p_{ik}f_{U,k-l+1}^t & \text{if } i>l, j>l,\\ \quad -p_{it}f_{U,1}^t - \sum_{k=1:k\neq t}^{l}p_{ik}f_k) & \\ (\theta_i)^{-1}\sum_{t=1}^{l}\xi_{k,i-l+1}^t(f_{U,1}^t\cdot\delta(j=t) + f_j\cdot\delta(j\neq t) & \text{if } i>l, j\leq l,\\ \quad -p_{it}f_{U,1}^t - \sum_{k=l+1}^{n}p_{ik}f_{U,k-l+1}^t - \sum_{k=1:k\neq t}^{l}p_{ik}f_k) & \\ (\theta_i)^{-1}\xi_{k,1}^i(f_{U,j-l+1}^i - \sum_{k=l+1}^{n}p_{ik}f_{U,k-l+1}^i - \sum_{k=1}^{l}p_{ik}f_k) & \text{if } i\leq l, j>l,\\ (\theta_i)^{-1}\xi_{k,1}^i(f_j - \sum_{k=l+1}^{n}p_{ik}f_{U,k-l+1}^i - \sum_{k=1}^{l}p_{ik}f_k) & \text{if } i\leq l, j\leq l.\end{cases} \qquad (13)$$

To compute α_{ij}, substitute $\xi_{k,p}^t$ by β_p^t without subscript k. Since computing the gradient of stability term requires much computation, a possible method to reduce this computational cost is to use only a subset of unlabeled examples to compute the gradient. We call this algorithm **Robust Graph Hyperparameter Learning (RobustHL)**. The graph constructed by this technique is called *robust graph*. Algorithm 1 shows the efficient gradient computation of RobustHL.

4 Experimental Results

In the experiment, we compare the prediction accuracy of four model selection methods, namely 5-fold Cross Validation, MinEnt, LOOHL, and RobustHL.

Algorithm 1. An efficient gradient update algorithm for RobustHL

initialize gradient $g \leftarrow (0, ..., 0)^{\mathrm{T}} \in \mathbb{R}^m$
compute α_{ij}, ρ_{ij}^k, and $s'(\bar{f}_{U,k})$
for $i, j = 1, ..., n$ **do**
  for each dimension $d = 1, ..., m$ **do**
    $g_d \leftarrow g_d + (\alpha_{ij} + \sum_{k=2}^{u+1} s'(\bar{f}_{U,k}) \cdot \rho_{ij}^k) \cdot \partial w(i,j)/\partial \sigma_d$
  end for
end for

These techniques are used to learn the hyperparameters of the graph. Then the accuracy of label assignment is computed. Instead of comparing the performance of our technique with other semi-supervised techniques, we pay more attention to the comparison of different model selection techniques. Furthermore, characteristics of the robust graph learned using the proposed stability measure are investigated. The benchmark used in the experiment consists of six data sets as shown in Table 1. The benchmark is well-known and is widely used to assess the power of various semi-supervised learning techniques. They can be categorized into two groups. The first group includes `g241c`, `g241d`, and `Digit1` which are artificially created in order for certain assumptions to hold. The data sets in second group are derived from real data to indicate the performance of algorithms in real world applications. It includes `USPS`, $\texttt{COIL}_2$, and `BCI`. The detailed explanation of each data set can be found in [11].

In the first experiment, we assess the performance of algorithms when there are 10 labeled examples for each data set. To see the effect of soft-label transformation, both thresholding and CMN are applied in each algorithm. To make the results from the experiment robust and independent of properties of the chosen data points, the algorithms are tested on randomly selected 12 subsets of labeled examples and report the average accuracy as well as its confidence interval.

Throughout the experiment, the smoothing factor ε is fixed at 0.001 for all model selection methods. The parameter γ is selected from $\{0.2, 0.4, 0.6, 0.8\}$. For 5-fold CV, the parameter σ_d is selected from $\{\frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2, 4, 8\}$ which is the same for all dimensions. The final objective function of other three methods is:

$$C_1 \times Loss + C_2 \times \sum_d (1/\sigma_d - 1/\tilde{\sigma}_d)^2/m \ ,$$

where *Loss* is the original cost function of MinEnt, LOOHL, and RobustHL. The second term in the above equation is introduced to regularize the value of σ_d by preventing the value from approaching either 0 or ∞. For each ratio $C_1 : C_2$ in $\{10^{-2}, 10^{-1}, 1, 10, 100\}$, the $\tilde{\sigma_d}$ is taken from $\{0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$ to avoid the local minima. All input vectors are normalized to have length 1. The result of this experiment is shown in Table 2.

In the second experiment, the data set is split into traning set and test set. In traning set, the number of labeled examples varies across 10%, 30%, 50%, 70%, and 90% of this set. Then the labels are assigned to unlabeled examples in both traning set and test set. For each particular set of labeled examples, the

Table 1. Basic properties of benchmark data sets

Data Set	Dimension	Points	Comment
g241c	241	1500	artificial
g241d	241	1500	artificial
Digit1	241	1500	artificial
USPS	241	1500	imbalance
COIL_2	241	1500	
BCI	117	400	

Table 2. Classification accuracy(%) and confidence interval with 10 labeled examples

Method	g241c	g241d	Digit1	USPS	COIL_2	BCI
CV+Thresh	50.90±3.0	51.64±4.5	51.31±5.2	80.01±.05	55.78±5.4	50.28±1.1
CV+CMN	52.14±4.2	52.61±5.3	57.04±13.7	80.02±.04	58.11±6.9	49.94±.75
MinEnt+Thresh	55.32±6.1	56.18±6.8	59.91±4.2	75.24±7.7	56.87±9.0	54.74±4.4
MinEnt+CMN	57.28±7.7	58.15±7.0	63.81±6.5	**82.86±4.6**	58.94±10.3	58.08±5.9
LOOHL+Thresh	76.27±4.0	57.04±5.0	63.39±3.0	80.27±5.2	68.47±5.9	69.10±5.4
LOOHL+CMN	73.28±4.8	**59.18±6.9**	63.9±7.7	81.60±4.1	69.74±7.8	69.40±4.9
RobustHL+Thresh	**76.50±2.5**	56.44±3.1	62.55±2.1	78.22±4.5	67.32±5.0	71.21±3.5
RobustHL+CMN	72.51±2.8	58.18±4.8	**65.03±7.1**	81.25±3.4	**71.25±5.6**	**72.12±2.8**

classification error and stability on test set are calculated. In this experiment, the same model selection methods as previous experiment are used. Figure 2 shows the plot of classification error and stability of four model selection methods when the number of labeled examples in each data set increases.

From the result in Table 2, we can make the following conclusions.

1. MinEnt, LOOHL, and RobustHL graph learning methods perform better than typical CV in all data sets except MinEnt+Thresh and RobustHL+Thresh in USPS. In this data set, CV+Thresh achieves higer accuracy than these two methods.
2. LOOHL and RobustHL in general achieve higher accuracy than MinEnt in all data sets except USPS. In this data set, MinEnt+CMN achieves higher accuracy than both LOOHL and RobustHL.
3. For LOOHL and RobustHL, both methods have competitive results. The accuracy achieved by these two techniques differs no more than 2%. Indeed, this is reasonable because the objective function of LOOHL and RobustHL is similar. The only difference between these methods is the stability measure introduced in RobustHL.
4. As a result of stability measure introduced in RobustHL, the confidence interval of RobustHL is tighter than that of LOOHL. This results is supported by the work of Belkin et al. [5] that the stability also affects the confidence of the classification. This means the traning accuracy of RobustHL should be closer to the true accuracy than LOOHL.

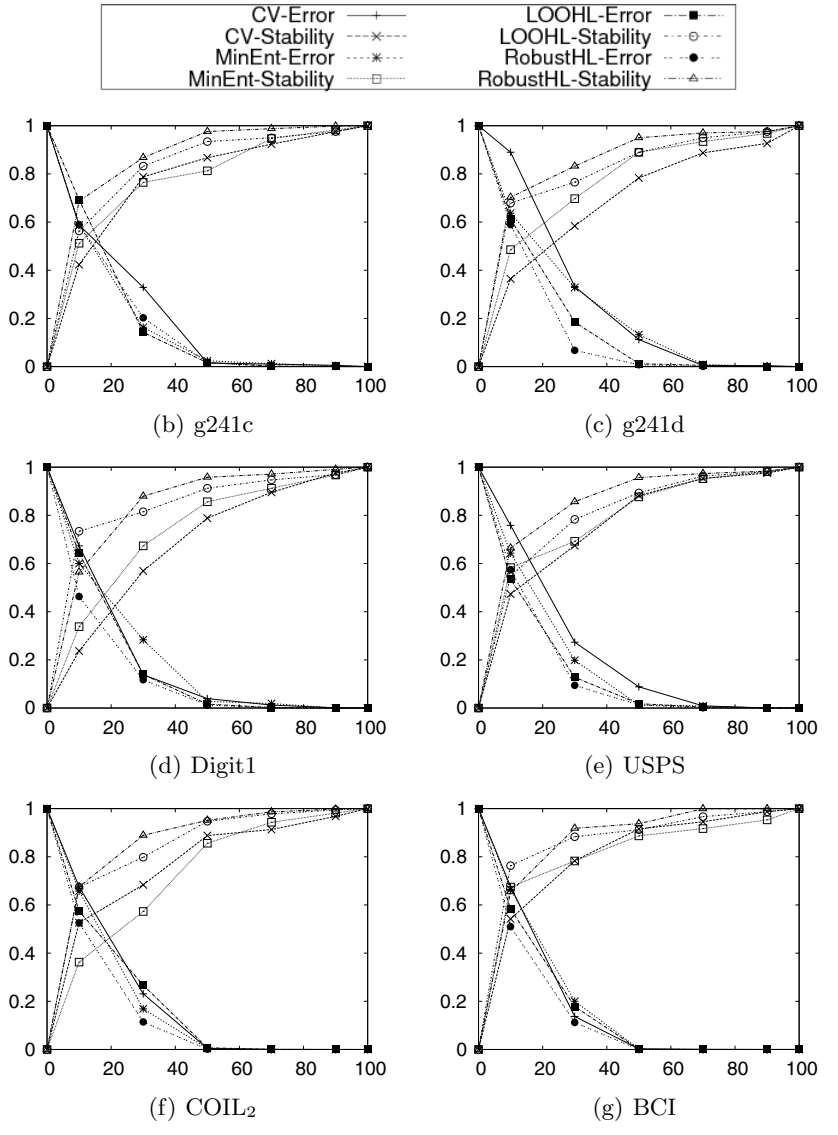


Fig. 2. The classification error and stability of the labeling function obtained from different model selection techniques against increasing number of labeled examples(%)

It is interesting to mention the following points about the stability of each model selection technique shown in Fig. 2.

1. In some data sets, the MinEnt method has lower stability than CV, which generally has lowest stability. As a result, it implies that the average label entropy may not be a good criterion for learning a robust graph. In [10], the

authors also state that MinEnt may fail to learn a good graph. This means the the average label entropy may be insufficient in graph learning.
2. In general, LOOHL method has higher stability than CV and MinEnt. Mentioned in [5] and [12], the stability of the learner is defined in term of leave-one-out scenario. As a result, the leave-one-out model selection scheme usually results in a stable function. Therefore, it is not surprising that the LOOHL can achieve such high stability.
3. For RobustHL, introducing explicit regularized term of stability in addition to the leave-one-out prodiction loss improves the stability of labeling function. As can be seen in Fig. 2, the stability on labeling function of RobustHL is the highest one. Therefore, we conclude that using the stability measure on unlabeled examples as a criterion in graph learning improves the stability of the labeling function, that consequently yields an improvement in classification accuracy.

In general, the labels of unlabeled examples can be predicted with higher confidence when more label information is avaliable. The predicted label of unlabeled examples are determined by their soft-label using simple thresholding or CMN. However, this soft-label may dramatically change and result in a different prediction when we add more labeled examples.

The stability of the soft-label on unlabeled examples can be used as a criterion to determine the sufficient number of labeled examples. As can be seen in Fig. 2, the stability of soft-label on unlabeled examples increases as more labeled examples are added. However, at particular point, it will increase very slowly, i.e., adding more labeled examples will no longer affect the soft-label of the unlabeled examples. In other words, the prediction result is unlikely to change. At this point, we can stop providing label information since doing so will no longer improve the accuracy of the classification.

According to the experimental results, RobustHL generally achieves higher stability and lower classification error than other three techniques. This means that the stability of the labeling function of RobustHL can be used as the criteria to determine the sufficient number of labeled examples very effectively.

5 Conclusion

In this work, the new hyperparameter learning in graph-based semi-supervised classification is proposed. The parameters is learned by minimizing leave-one-out prediction error on labeled examples while retaining high stability on unlabeled examples. The problem can be solved efficiently using gradient-based optimization. Encouraging results show that the proposed technique has high performance compared to existing model selection techniques in graph-based semi-supervised learning.

Acknowledgments. This work is supported in part by the Young Scientist and Technologist Programme or YSTP (SIIT-NSTDA:S1Y48/F-002) of the National

Science and Technology Development Agency, Thailand. Special thank to Xinhua Zhang for providing the useful codes for testing.

References

1. Zhu, X., Ghahramani, Z.: Learning from labeled and unlabeled data with label propagation. Technical report, Carnegie Mellon University (2002)
2. Blum, A., Chawla, S.: Learning from labeled and unlabeled data using graph mincuts. In: Brodley, C.E., Danyluk, A.P., Brodley, C.E., Danyluk, A.P. (eds.) ICML, pp. 19–26. Morgan Kaufmann, San Francisco (2001)
3. Blum, A., La, J., Rwebangira, M., Reddy, R.: Semi-supervised learning using randomized mincuts (2004)
4. Zhou, D., Bousquet, O., Lal, T., Weston, J., Schülkopf, B.: Learning with local and global consistency. In: Thrun, S., Saul, L.K., Schülkopf, B. (eds.) NIPS 2003, vol. 16, pp. 321–328. MIT Press, Cambridge (2004)
5. Belkin, M., Matveeva, I., Niyogi, P.: Regularization and semi-supervised learning on large graphs. In: Shawe-Taylor, J., Singer, Y. (eds.) COLT 2004. LNCS, vol. 3120, pp. 624–638. Springer, Heidelberg (2004)
6. Kapoor, A., Qi, Y.A., Ahn, H., Picard, R.: Hyperparameter and kernel learning for graph based semi-supervised classification. In: Weiss, Y., Schölkopf, B., Platt, J. (eds.) Advances in Neural Information Processing Systems 18, pp. 627–634. MIT Press, Cambridge (2006)
7. Carreira-Perpiñán, M.Á., Zemel, R.S.: Proximity graphs for clustering and manifold learning. In: NIPS (2004)
8. Wang, F., Zhang, C.: Label propagation through linear neighborhoods. In: ICML 2006 (2006)
9. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using gaussian fields and harmonic functions. In: Proceedings of the 20th International Conference on Machine Learning (ICML 2003) (2003)
10. Zhang, X., Lee, W.S.: Hyperparameter learning for graph based semi-supervised learning algorithms. In: Schölkopf, B., Platt, J., Hoffman, T. (eds.) Advances in Neural Information Processing Systems 19, pp. 1585–1592. MIT Press, Cambridge (2007)
11. Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-Supervised Learning. MIT Press, Cambridge (2006)
12. Bousquet, O., Elisseeff, A.: Algorithmic stability and generalization performance. In: Leen, T.K., Dietterich, T.G., Tresp, V. (eds.) NIPS, pp. 196–202. MIT Press, Cambridge (2000)

Regularized Local Reconstruction for Clustering

Jun Sun[1,2], Zhiyong Shen[1,2], Bai Su[1,2], and Yidong Shen[1]

[1]State Key Laboratory of Computer Science,
Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
[2]Graduate University, Chinese Academy of Sciences, Beijing 100049, China
{junsun,zyshen,subai,ydshen}@ios.ac.cn

Abstract. Motivated by the local reconstruction approach to discovering low dimensional structure in high dimensional data, we propose a novel clustering algorithm that effectively utilizes local reconstruction information. We obtain the local reconstruction weights by minimizing the reconstruction error between each data point and the reconstruction from its neighbors. An entropy regularization term is incorporated into the reconstruction objective function so that the smoothness of the reconstruction weights can be explicitly controlled. The reconstruction weights are then used to obtain the clustering result by employing spectral clustering techniques. Experimental results on a number of datasets demonstrate that our algorithm performs well relative to other approaches, which validate the effectiveness of our approach for clustering.

Keywords: Clustering, Local Reconstruction, Entropy Regularization.

1 Introduction

Data clustering, also known as cluster analysis, is one of the most fundamental problems and an active and important research area in data mining and machine learning. Generally speaking, the goal of clustering is to discover meaningful and natural groupings in data automatically. Usually it involves partitioning a finite set of data objects into several disjoint clusters so that intra-cluster similarity and inter-cluster separability are maximized. Since no labeled data (supervised information) are available to guide the partitioning process, data clustering falls under the category of unsupervised learning.

Various clustering techniques have been proposed over the years. Among these techniques, one of the most interesting approaches is spectral clustering [11], which has received considerable attention in recent years. Spectral clustering algorithms aim to recover the clustering structure in data by exploiting the top eigenvectors of a specially constructed matrix. Spectral clustering algorithms are often easy to implement using linear algebra software packages and can be very efficient if the specially constructed matrix is sparse. Motivated from a graph partitioning perspective, various spectral clustering algorithms have been proposed based on different graph cut objectives such as ratio cut [4] and normalized cut [11]. Spectral clustering techniques are also used in clustering methods which

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 110–121, 2009.
© Springer-Verlag Berlin Heidelberg 2009

are based on the local learning idea [13,16]. A lot of applications exist for spectral clustering techniques, such as image segmentation [11,17], circuit layout [4], speech separation [1] and so on.

In this paper, we derive a new clustering algorithm that also utilizes the top eigenvectors of a specially constructed matrix, hence inheriting the advantages of spectral clustering. However, our algorithm is motivated by the local reconstruction approach to dealing with high dimensional data that lie on or near a low dimensional manifold [9]. The local reconstruction approach tries to capture geometric properties of the underlying data manifold by locally reconstructing each data point from its neighbors. The local reconstruction perspective has already proved very useful in dimensionality reduction [8,9] and semi-supervised learning [15].

Based on the local reconstruction perspective, we propose a novel clustering algorithm that effectively utilizes local reconstruction information. Our proposed approach consists of two parts. In the first part, the local reconstruction weights are obtained by minimizing the reconstruction error between each data point and the reconstruction from its neighbors. Important geometric characteristics of local neighborhoods can be preserved by the local reconstruction weights [9]. In the second part of our approach, the reconstruction weights are then used to produce the final clustering result.

In our clustering approach, the reconstruction of each data point is performed in the reproducing kernel Hilbert space. Thus, nonlinear relations between each data point and its neighbors can be captured. Besides, an entropy regularization term is incorporated into the reconstruction objective function so that the smoothness of the reconstruction weights can be explicitly controlled. In the second part of our algorithm, the special structures of the scaled cluster label matrix and the reconstruction weight matrix make the optimization problem much easier to analyze, and spectral clustering techniques can be employed to solve the clustering problem efficiently. Experimental results on a number of real-world datasets demonstrate that our algorithm performs well relative to other spectral clustering approaches, which validate the effectiveness of our approach in obtaining good clusterings.

The remainder of the paper is organized as follows. We introduce the formulation of the proposed clustering model in Section 2. In Section 3, we derive the detailed clustering algorithm. Experimental results are presented in Section 4. Section 5 concludes the paper and discusses future works.

2 Clustering Model

2.1 Notations

First we introduce some notations. Boldface lowercase letters, such as $\mathbf{x}$ and $\mathbf{y}$, denote column vectors. Boldface uppercase letters, such as $\mathbf{M}$ and $\mathbf{W}$, denote matrices. $\mathbf{M}^T$ denotes the transpose of $\mathbf{M}$. $\mathbf{W} \geq 0$ means that every entry in $\mathbf{W}$ is nonnegative. $\mathbf{1}_m \in \mathbb{R}^m$ is a vector of 1's. $\Delta_m = \left\{\mathbf{w} \in \mathbb{R}_m \,|\, \mathbf{w}^T\mathbf{1}_m = 1,\, \mathbf{w} \geq 0\right\}$

is the probability simplex. For any $\mathbf{w} \in \Delta_m$, the elements of $\mathbf{w}$ are nonnegative and sum to one. $\mathbf{I}_m \in \mathbb{R}^{m\times m}$ is the identity matrix of order m. For $\mathbf{x} = [x_1, \ldots, x_m]^T \in \mathbb{R}^m$, $\|\mathbf{x}\|_1 = \sum_{i=1}^m |x_i|$ denotes the ℓ_1 norm. The set of nonnegative real numbers is denoted as $\mathbb{R}_+$. For a square matrix $\mathbf{A} \in \mathbb{R}^{m\times m}$, Trace($\mathbf{A}$) is the trace of $\mathbf{A}$, i.e., the sum of the diagonal elements of $\mathbf{A}$.

2.2 Data and Label Representation

Let $\mathcal{X} \subseteq \mathbb{R}^d$ denote the input space from which n data points, $\mathbf{x}_1, \cdots, \mathbf{x}_n$, where $\mathbf{x}_i = [x_{i1}, \ldots, x_{id}]^T$, are sampled. Given a set of data points $\{\mathbf{x}_i\}_{i=1}^n \subseteq \mathcal{X} \subseteq \mathbb{R}^d$, the goal of data clustering is to find a set of disjoint and exhaustive clusters $\{\pi_l\}_{l=1}^c$ from the data where π_l is the l-th cluster. c is the number of clusters. $|\pi_l|$ is the number of points in the l-th cluster.

Given a positive semi-definite kernel function $K(\cdot,\cdot)$, the data points from the original input space $\mathcal{X}$ are mapped to a possibly infinite dimensional reproducing kernel Hilbert space $\mathcal{F}$ [10]. The mapping is denoted as $\phi : \mathcal{X} \mapsto \mathcal{F}$. The reconstruction of each data point $\mathbf{x}_i$ is then performed in $\mathcal{F}$ so that nonlinear relations between $\mathbf{x}_i$ and its neighbors can be captured. We denote the inner product in $\mathcal{F}$ as $\langle\cdot,\cdot\rangle_{\mathcal{F}}$, so

$$\langle\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle_{\mathcal{F}} = K(\mathbf{x}_i, \mathbf{x}_j) \qquad \text{and} \qquad \|\phi(\mathbf{x}_i)\|_{\mathcal{F}}^2 = \langle\phi(\mathbf{x}_i), \phi(\mathbf{x}_i)\rangle_{\mathcal{F}} \tag{1}$$

A clustering solution is represented by a *Scaled Cluster Label Matrix* $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_n]^T = [y_{il}] \in \mathbb{R}^{n\times c}$ where $\mathbf{y}_i = [y_{i1}, y_{i2}, \ldots, y_{ic}]^T \in \mathbb{R}^c$. Specifically,

$$y_{il} = \begin{cases} \frac{1}{|\pi_l|} & \text{if } \mathbf{x}_i \in \pi_l, \\ 0 & \text{if } \mathbf{x}_i \notin \pi_l. \end{cases} \tag{2}$$

$\mathbf{Y}$ is the unknown variable in our clustering model. Once $\mathbf{Y}$ is known, the resultant clustering can be obtained easily. So each data point $\mathbf{x}_i$ is associated with a cluster label $\mathbf{y}_i \in \mathbb{R}^c$. The scaling in $\mathbf{Y}$ is used for producing balanced clusters.

In the following, neighborhood $\mathcal{N}_i$ denotes a set of nearest neighbors of point $\mathbf{x}_i$, not including $\mathbf{x}_i$. The number of neighbors of $\mathbf{x}_i$ is $n_i = |\mathcal{N}_i|$. Usually, $n_i \ll n$.

2.3 Clustering Model

The proposed clustering model consists of two parts, which produce the local reconstruction weights (represented by $\mathbf{W}$) and final clustering ($\mathbf{Y}$) respectively.

In the first part, we try to locally reconstruct each data point $\phi(\mathbf{x}_i)$ from its neighbors. The local reconstruction for data point $\phi(\mathbf{x}_i)$ is $\sum_{\mathbf{x}_j\in\mathcal{N}_i} w_{ij}\phi(\mathbf{x}_j)$ where $w_{ij} \geq 0$ and $\sum_{\mathbf{x}_j\in\mathcal{N}_i} w_{ij} = 1$. All the reconstruction weights form a weight matrix $\mathbf{W} = [w_{ij}] \in \mathbb{R}^{n\times n}$ where $w_{ij} = 0$ if $\mathbf{x}_j \notin \mathcal{N}_i$. Besides, we assume that $\mathbf{w}_i \in \mathbb{R}^{n_i}$ is composed of w_{ij} where $\mathbf{x}_j \in \mathcal{N}_i$, so $\mathbf{w}_i \in \Delta_{n_i}$. The reconstruction error for $\phi(\mathbf{x}_i)$ is denoted by $L_x\left(\phi(\mathbf{x}_i), \sum_{\mathbf{x}_j\in\mathcal{N}_i} w_{ij}\phi(\mathbf{x}_j)\right)$, the value of which is to be minimized. Important geometric characteristics of local neighborhoods

can be preserved by the local reconstruction weights w_{ij}, which in turn can be used to obtain the final clustering result. Here, a natural choice for the error measure L_x is $L_x\left(\phi(\mathbf{x}_i), \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\phi(\mathbf{x}_j)\right) = \left\|\phi(\mathbf{x}_i) - \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\phi(\mathbf{x}_j)\right\|_{\mathcal{F}}^2$.

In the second part, the reconstruction weights are used to obtain the clustering result. The objective is to minimize the cluster label reconstruction error $\mathcal{E}(\mathbf{Y}) = \sum_{i=1}^{n} L_y\left(\mathbf{y}_i, \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\mathbf{y}_j\right)$, which measures how well the cluster label of each data point can be reconstructed from its' neighbors' cluster labels, using the same reconstruction weights obtained in the first part of our clustering model. Here, we use sum of absolute error as the discrepancy measure, namely, $L_y\left(\mathbf{y}_i, \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\mathbf{y}_j\right) = \left\|\mathbf{y}_i - \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\mathbf{y}_j\right\|_1$.

The two parts of our clustering model are formalized as follows.

1. Compute the local reconstruction weights. For each $\mathbf{x}_i$ where $1 \leq i \leq n$, solve the following problem:

$$\begin{aligned} \min_{\mathbf{w}_i \in \mathbb{R}^{n_i}} \quad & \mathcal{E}_i(\mathbf{w}_i) = \left\|\phi(\mathbf{x}_i) - \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\phi(\mathbf{x}_j)\right\|_{\mathcal{F}}^2 - \gamma\, \widehat{H}(\mathbf{w}_i) \\ \text{subject to} \quad & \mathbf{w}_i \in \Delta_{n_i} \end{aligned} \tag{3}$$

2. Compute the label matrix $\mathbf{Y}$ by solving the following problem:

$$\begin{aligned} \min_{\mathbf{Y} \in \mathbb{R}^{n \times c}} \quad & \mathcal{E}(\mathbf{Y}) = \sum_{i=1}^{n} \left\|\mathbf{y}_i - \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\mathbf{y}_j\right\|_1 \\ \text{subject to} \quad & \mathbf{Y} \text{ is defined in (2)} \end{aligned} \tag{4}$$

In Eq.(3), $-\widehat{H}(\mathbf{w}_i)$ is the entropy regularization term which explicitly controls the smoothness of the weight components in $\mathbf{w}_i$. $\gamma \geq 0$ is a pre-specified parameter. A larger value of γ means that more uniform weights are preferable. $\widehat{H}(\cdot)$ is the generalized entropy which will be defined and discussed in Section 2.4. A key property of $\widehat{H}(\cdot)$ is that the more uniform the weights in $\mathbf{w}_i$ are, the larger the value of $\widehat{H}(\mathbf{w}_i)$ becomes.

2.4 Generalized Entropy

We introduce the concept of generalized entropy as a measure of the degree of uncertainty within a discrete probability distribution (over a set of m elements) that can be represented by a vector $\mathbf{w} = [w_1, \ldots, w_m]^T \in \Delta_m$. A definition of generalized entropy recently proposed in [7] is provided in the following:

Definition 1. Generalized entropy *is defined as a mapping*

$$\widehat{H} : \Delta_m \mapsto \mathbb{R}_+$$

that satisfies the following two criteria (symmetry and concavity):

1. *For any* $\mathbf{w}_a \in \Delta_m$*, and any* $\mathbf{w}_b \in \Delta_m$ *whose elements are a permutation of the elements of* $\mathbf{w}_a$*,* $\widehat{H}(\mathbf{w}_a) = \widehat{H}(\mathbf{w}_b)$*.*
2. $\widehat{H}(\cdot)$ *is a concave function.*

This definition reflects an important property of generalized entropy $\widehat{H}(\cdot)$: more uniform elements of $\mathbf{w}$ indicate larger values of $\widehat{H}(\mathbf{w})$. For example, if $\mathbf{v}_1 = [0.1\ 0.3\ 0.6]^T \in \Delta_3$ and $\mathbf{v}_2 = [0.3\ 0.3\ 0.4]^T \in \Delta_3$, then $\widehat{H}(\mathbf{v}_1) \leq \widehat{H}(\mathbf{v}_2)$. Many entropies proposed in the literature are special cases of Definition 1 [7]. Some examples are given as follows:

1. $\widehat{H}(\mathbf{w}) = \sum_{i=1}^{m} -w_i \log(w_i)$, which is the *Shannon entropy.*
2. $\widehat{H}(\mathbf{w}) = 1 - \mathbf{w}^T\mathbf{w}$, which will be referred to as ℓ_2*-entropy.*

3 Algorithm

In this section, we derive a clustering algorithm based on the clustering model presented in the previous section. Our algorithm consists of two parts. The first part computes the local reconstruction weight matrix $\mathbf{W}$ by solving the optimization problem (3). The second part computes the final clustering by solving problem (4).

3.1 The Computation of W

The kernel matrix over $\{\mathbf{x}_i\}_{i=1}^n$ is denoted as $\mathbf{K} \in \mathbb{R}^{n\times n}$ so that $K(\mathbf{x}_i, \mathbf{x}_j)$ is the element in the i-th row and j-th column of $\mathbf{K}$. The kernel matrix over $\mathcal{N}_i$ is denoted as $\mathbf{K}_i \in \mathbb{R}^{n_i\times n_i}$, which is a submatrix of $\mathbf{K}$ corresponding to the data points in $\mathcal{N}_i$. If $N_j(\mathbf{x}_i)$ denotes the j-th neighbor of $\mathbf{x}_i$, then the vector $\mathbf{k}_i = [K(\mathbf{x}_i, N_1(\mathbf{x}_i)), K(\mathbf{x}_i, N_2(\mathbf{x}_i)), \ldots, K(\mathbf{x}_i, N_{n_i}(\mathbf{x}_i))]^T \in \mathbb{R}^{n_i}$ is used to denote the kernel function values between data point $\mathbf{x}_i$ and its neighbors.

After some algebraic operations using Eq.(1), problem (3) can be simplified to an equivalent problem as follows.

$$\begin{aligned} \min_{\mathbf{w}_i\in\mathbb{R}^{n_i}} \quad & \mathbf{w}_i^T\mathbf{K}_i\mathbf{w}_i - 2\mathbf{k}_i^T\mathbf{w}_i - \gamma\,\widehat{H}(\mathbf{w}_i) \\ \text{subject to} \quad & \mathbf{w}_i \in \Delta_{n_i} \end{aligned} \tag{5}$$

Even with generalized entropy defined in Section 2.4, this sub-problem for computing $\mathbf{w}_i$ is still a small-scale (since $n_i \ll n$) convex programming problem, since $\widehat{H}(\cdot)$ is a concave function and all the constraints with respect to $\mathbf{w}_i$ are linear. There're very effective and efficient algorithms that can solve convex programs reliably [2]. For the algorithm derived in this section, we'll use $\widehat{H}(\mathbf{w}) = 1 - \mathbf{w}^T\mathbf{w}$ which is the ℓ_2*-entropy.* Problem (3) is then equivalent to the following quadratic programming problem.

$$\begin{aligned} \min_{\mathbf{w}_i\in\mathbb{R}^{n_i}} \quad & \mathbf{w}_i^T\left(\mathbf{K}_i + \gamma\mathbf{I}_{n_i}\right)\mathbf{w}_i - 2\mathbf{k}_i^T\mathbf{w}_i \\ \text{subject to} \quad & \mathbf{w}_i \in \Delta_{n_i} \end{aligned} \tag{6}$$

3.2 The Computation of Y

In this subsection, we want to obtain the final clustering by optimizing problem (4). This objective function appears difficult to optimize because of the constraint (2) and the ℓ_1 norm $\|\cdot\|_1$ in problem (4). However, two key properties of $\mathbf{A}$ in (7) ensure that the problem can be optimized using spectral clustering techniques. The two key properties of $\mathbf{W}$ are

$$\mathbf{W} \geq 0 \qquad \text{and} \qquad \mathbf{W}\mathbf{1}_n = \mathbf{1}_n \tag{7}$$

For simplifying problem (4), we define $\mathbf{H} = [\tilde{\mathbf{h}}_1, \ldots, \tilde{\mathbf{h}}_c] = [h_{il}] \in \mathbb{R}^{n \times c}$ as follows

$$h_{il} = \begin{cases} \sqrt{\frac{1}{|\pi_l|}} & \text{if } \mathbf{x}_i \in \pi_l, \\ 0 & \text{if } \mathbf{x}_i \notin \pi_l. \end{cases} \tag{8}$$

An important property of $\mathbf{H}$ is that $\mathbf{H}^T\mathbf{H} = \mathbf{I}_c$.

Also, we define $\mathbf{A} = \mathbf{W} + \mathbf{W}^T$ and $\mathbf{M} = \mathrm{Deg}(\mathbf{A}) - \mathbf{A}$ where $\mathrm{Deg}(\mathbf{A})$ is the degree matrix of $\mathbf{A}$, i.e., the diagonal matrix whose diagonal elements are the sums of rows of $\mathbf{A}$.

Then the optimization problem in (4) can be proved to be equivalent to the following problem [13]:

$$\min_{\mathbf{H} \in \mathbb{R}^{n \times c}} \mathrm{Trace}\left(\mathbf{H}^T\mathbf{M}\mathbf{H}\right) \qquad \text{s.t. } \mathbf{H} \text{ is defined in (8)} \tag{9}$$

A sketched proof can be found in the Appendix.

Relaxation. As is done in a typical spectral clustering algorithm, $\mathbf{H}$ defined in (8) is relaxed to be any matrix in $\left\{\mathbf{H} \in \mathbb{R}^{n \times c} \,|\, \mathbf{H}^T\mathbf{H} = \mathbf{I}_c\right\}$. The relaxed optimization problem is in the following:

$$\min_{\mathbf{H} \in \mathbb{R}^{n \times c}} \mathrm{Trace}\left(\mathbf{H}^T\mathbf{M}\mathbf{H}\right) \qquad \text{s.t. } \mathbf{H}^T\mathbf{H} = \mathbf{I}_c \tag{10}$$

According to the *Ky Fan* Theorem [18], the globally optimal solution set of problem (10) is as follows:

$$\left\{\mathbf{H}^\star\mathbf{Q} \,|\, \mathbf{Q} \in \mathbb{R}^{c \times c}, \mathbf{Q}^T\mathbf{Q} = \mathbf{I}_c\right\} \tag{11}$$

where $\mathbf{H}^\star \in \mathbb{R}^{n \times c}$ is formed by the eigenvectors corresponding to the c smallest eigenvalues of $\mathbf{M}$.

Discretization. In this step, the solution $\mathbf{H}^\star$ to the relaxed problem (10) needs to be discretized to produce a clustering result. We use the discretization method proposed in [17], which tries to rotate $\mathbf{H}^\star$ so that it's close to a cluster indicator matrix. The details of this method can be found in [17].

Algorithm: RLRC ($\mathbf{K}$, c, k, γ)
Input: Kernel matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$, number of output clusters c, number of nearest neighbors k and parameter γ.
Output: Clustering solution $\{\pi_l\}_{l=1}^{c}$.
Procedure:
1. For each $\mathbf{x}_i$, compute the k nearest neighbors ($\mathcal{N}_i$) based on kernel matrix $\mathbf{K}$.
2. **for** each $1 \leq i \leq n$
   Compute the reconstruction weight vector $\mathbf{w}_i$ by solving the quadratic programming problem (6).
   end for
3. Construct the sparse matrix $\mathbf{W}$ by combining the $\mathbf{w}_i$'s together.
4. Construct the sparse matrix $\mathbf{M} = \mathrm{Deg}(\mathbf{W} + \mathbf{W}^T) - \mathbf{W} - \mathbf{W}^T$.
5. Construct the matrix $\mathbf{H}^\star$ by computing the eigenvectors corresponding to the c smallest eigenvalues of $\mathbf{M}$.
6. Discretize $\mathbf{H}^\star$ to get the clustering solution $\{\pi_l\}_{l=1}^{c}$.
7. **return** $\{\pi_l\}_{l=1}^{c}$;

Fig. 1. RLRC **algorithm**

3.3 Main Algorithm

In practice, the number of neighbors n_i for each data point $\mathbf{x}_i$ is often fixed to a small value $k \ll n$, i.e., $n_i = k$ for all $1 \leq i \leq n$. Thus, each neighborhood $\mathcal{N}_i$ is defined as the set of k-nearest neighbors of $\mathbf{x}_i$, not including $\mathbf{x}_i$, using some distance metric. The distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ is measured by

$$\mathcal{D}\left(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\right) = \left\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\right\|_{\mathcal{F}} = \sqrt{K(\mathbf{x}_i, \mathbf{x}_i) + K(\mathbf{x}_j, \mathbf{x}_j) - 2K(\mathbf{x}_i, \mathbf{x}_j)}$$

Therefore, given a kernel matrix, the neighborhood of each data point can be computed easily. Here, k is provided by domain experts.

The main algorithm is summarized in Fig.1. We name the algorithm Regularized Local Reconstruction for Clustering (RLRC).

3.4 Computational Complexity

The main computational load comes from step 1, 2 and 5. Given a kernel matrix, the k-nearest neighbors of each data point can be computed with complexity $O(n)$. So the overall complexity for step 1 is $O(n^2)$. For step 2, since each quadratic programming problem requires time $O(k^3)$, the overall time complexity for step 2 is $O(nk^3)$. For step 5 which eigen-decomposes a $n \times n$ matrix with $O(nk)$ nonzero elements to obtain the top c eigenvectors, the time complexity is $O(n^2c)$ without special optimizations. Therefore, the time complexity of RLRC algorithm in Fig.1 is $O(nk^3 + n^2c)$. Note that the complexity of step 5 can be reduced to subquadratic in n by adopting more efficient methods for sparse eigen-decomposition problem [5].

Table 1. Summary of datasets

Name	Source	n	d	c
cranmed	CRANFIELD/MEDLINE	2431	41681	2
k1a	WebACE	2340	21839	20
k1b	WebACE	2340	21839	6
re0	Reuters-21578	1504	2886	13
re1	Reuters-21578	1657	3758	25
tr11	TREC	414	6429	9
tr12	TREC	313	5804	8
tr23	TREC	204	5832	6
tr31	TREC	927	10128	7
tr41	TREC	878	7454	10
tr45	TREC	690	8261	10
wap	WebACE	1560	8460	20

4 Experimental Results

In this section, we conduct experiments on a number of real-world datasets to evaluate the effectiveness of our clustering algorithm by comparing its performance with other related methods.

4.1 Summary of Datasets

In this subsection, we will introduce the datasets used in our experiments. We use 12 document datasets[1] from the CLUTO toolkit [19]. Table 1 summarizes the basic information of the datasets.

Dataset *cranmed* is composed of the CRANFIELD and MEDLINE abstracts[2]. The three datasets *k1a*, *k1b* and *wap* are from the WebACE project where each document corresponds to a web page listed in the subject hierarchy of Yahoo!. Datasets *re0* and *re1* are derived from Reuters-21578 text collection [6]. The six datasets *tr11*, *tr12*, *tr23*, *tr31*, *tr41*, and *tr45* are from the TREC collection [14]. The processed datasets are also available from `http://shi-zhong.com/software/docdata.zip`. These datasets provide a good representation of different data distributions and characteristics, and so are good candidates for evaluating different clustering algorithms.

4.2 Clustering Evaluation Criteria

In all the experiments, the "true" number of classes c is provided to all the considered clustering algorithms. Given class labels, we evaluate the clustering results using two external validity measures, Normalized Mutual Information (*NMI*) [12] and Clustering Accuracy (*Acc*) [13,16]. *NMI* is an information-theoretic

[1] `http://glaros.dtc.umn.edu/gkhome/fetch/sw/cluto/datasets.tar.gz`
[2] Available from `ftp://ftp.cs.cornell.edu/pub/smart`

measure that's previously defined in [12] to compare different clusterings. *Acc* [13,16] is calculated based on the one-to-one correspondence between the clusters and the classes. We denote a permutation function as $\sigma(\cdot) : \{i\}_{i=1}^{c} \mapsto \{j\}_{j=1}^{c}$ which maps each cluster index i to a class index $\sigma(i)$. *Acc* is calculated as follows:

$$Acc = \frac{\max\left(\sum_{i=1}^{c} n_{i,\sigma(i)}\right)}{n} \tag{12}$$

where $n_{i,\sigma(i)}$ is the number of points which are in both the i-th cluster and $\sigma(i)$-th class. Note that larger values of *NMI* and *Acc* suggest better clustering solutions.

4.3 Experimental Settings

We normalize each document vector (Bag-of-Words) to unit norm and then the cosine kernel [16] is adopted as the kernel function in all the experiments. We compare the performance of the following clustering algorithms:

- Spectral clustering with normalized cut (NCut) [11]. The symmetric weighted k-nearest neighbor graph is used as the affinity graph. The edge weight between two connected data points in the graph is calculated using the kernel function (cosine similarity).
- Local Learning based Clustering Algorithm[3] (LLCA) proposed in [16]. There's a regularization parameter λ in LLCA. As is done in [16], we choose this parameter from: $\lambda \in \{0.1, 1, 1.5\}$. We report the best performance among the results produced when different values of λ are used, which gives LLCA an unfair advantage over others.
- Clustering via local regression (CLOR) proposed in [13]. This algorithm is also based on the local learning idea as LLCA.
- Our proposed algorithm RLRC. We fix the regularization parameter $\gamma = 1$ to avoid the difficult parameter selection procedure.

All the above algorithms cluster the data by utilizing local neighborhood information (k-nearest neighbors) of the data points. Besides, all of them use spectral clustering techniques to optimize the objective functions. We use the same discretization method[4] for all of them. Note that in all the experiments, the number of classes c is provided to all the clustering algorithms.

4.4 Performance Results

In this subsection, we will provide an empirical study of the considered algorithms. Clustering performance results on the datasets in terms of both *NMI* and *Acc* values will be compared and discussed .

In particular, we aim to address the following two important questions:

[3] The code is available at `www.kyb.tuebingen.mpg.de/bs/people/mingrui/LLCA.zip`
[4] The code is available at `http://www.cis.upenn.edu/~jshi/software/`

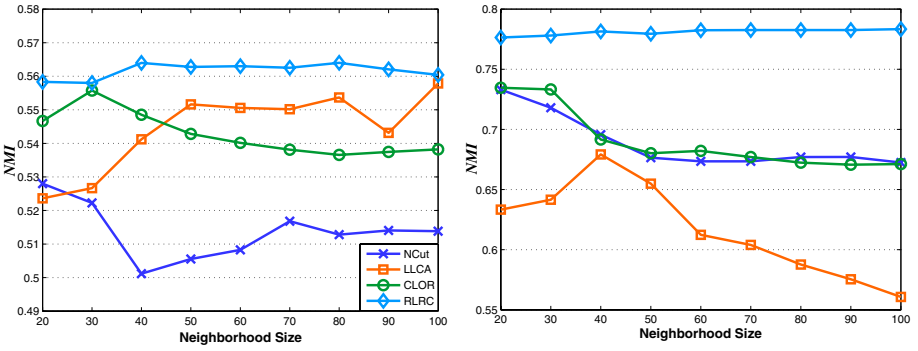
(1) How effective is the proposed algorithm RLRC compared with the other three considered algorithms?
(2) How will the performance of RLRC change with different neighborhood sizes?

Generally, the optimal value of k is not easy to determine. In practice, we choose k on the order of $\log(n)$ so that the k-nearest neighbor graph is asymptotically connected under some special conditions [3]. Therefore, for datasets with size on the order of 1000, we can choose k to be a small multiple of 10. The experimental results on the 12 datasets when neighborhood size $k = 40$ are presented in Table 2. It can be observed that CLOR and the proposed algorithm RLRC achieve the best *NMI* and *Acc* values. On 8 out of 12 datasets, our algorithm RLRC achieves the best *NMI* and *Acc* values. On dataset *k1a*, our algorithm achieves the best *NMI* value but CLOR achieves the best *Acc* value. On dataset *re0* and *re1*, our algorithm achieves the best *Acc* values but CLOR achieves the best *NMI* values. On dataset *tr41*, CLOR outperforms RLRC with respect to both *NMI* and *Acc* values. However, the *NMI* and *Acc* values achieved by our algorithm RLRC are very close to those achieved by CLOR on dataset *tr41*. The clustering results in Table 2 demonstrate the effectiveness and potential of our proposed clustering approach in discovering accurate clusterings.

We also conduct experiments on the considered datasets with different neighborhood sizes k. Figure 2 shows how clustering performance results (*NMI*) vary with different values of k on dataset *k1a* and *k1b*. The horizontal axis denotes the neighborhood size k and the vertical axis denotes the clustering performance measured by *NMI*. With different values of k, our algorithm RLRC achieves more stable results compared with other algorithms. Among the remaining three algorithms, CLOR is more stable than the other two algorithms with respect to different values of k.

Table 2. Experimental results of the considered algorithms on the datasets. Both *NMI* and *Acc* results are listed in the table. For each dataset, the results with highest *NMI* and *Acc* values are shown in boldface. Here, the neighborhood size $k = 40$ for all the considered algorithms.

	NMI				*Acc*			
	NCut	LLCA	CLOR	RLRC	NCut	LLCA	CLOR	RLRC
cranmed	0.8628	0.4163	0.8906	**0.9413**	0.9778	0.8198	0.9835	**0.9930**
k1a	0.5012	0.5412	0.5485	**0.5640**	0.3970	0.4132	**0.4432**	0.4282
k1b	0.6957	0.6792	0.6916	**0.7814**	0.7987	0.8162	0.7368	**0.8688**
re0	0.4005	0.4085	**0.4264**	0.3954	0.3265	0.3883	0.3291	**0.4402**
re1	0.4842	0.4854	**0.4982**	0.4957	0.3621	0.3784	0.3856	**0.4683**
tr11	0.6242	0.6201	0.6740	**0.7234**	0.5725	0.5652	0.6522	**0.7271**
tr12	0.5994	0.6221	0.6568	**0.7373**	0.6454	0.6613	0.7380	**0.7796**
tr23	0.3441	0.2982	0.3343	**0.3574**	0.4118	0.3775	0.4510	**0.4608**
tr31	0.4566	0.4990	0.4832	**0.5341**	0.5491	0.5620	0.5383	**0.6095**
tr41	0.6031	0.6220	**0.6416**	0.6203	0.5809	0.5763	**0.6355**	0.6253
tr45	0.5582	0.5848	0.6311	**0.6641**	0.5783	0.6681	0.6493	**0.6739**
wap	0.5249	0.5420	0.5406	**0.5768**	0.3904	0.4231	0.4199	**0.4795**



(a) *k1a* (b) *k1b*

Fig. 2. The clustering results (*NMI*) by the considered algorithms on datasets *k1a* and *k1b* with different neighborhood sizes (k). The legend is shown only in subfigure (a) for clarity.

5 Conclusion

In this paper, we propose a new clustering algorithm, namely, Regularized Local Reconstruction for Clustering (RLRC), which is motivated by the local reconstruction approach to dealing with high dimensional data that lie on or near a low dimensional manifold [9]. Our clustering method effectively utilizes local reconstruction information which captures important geometric characteristics of local neighborhoods by locally reconstructing each data point from its neighbors. Spectral clustering techniques can be employed to solve the clustering problem efficiently. Experimental results on a number of datasets demonstrate the effectiveness of our proposed algorithm. Future work includes how to select the neighborhood size and the regularization parameter automatically.

Acknowledgments. This work is supported in part by NSFC grants 60673103 and 60721061.

References

1. Bach, F.R., Jordan, M.I.: Learning Spectral Clustering, with Application To Speech Separation. Journal of Machine Learning Research 7, 1963–2001 (2006)
2. Boyd, S., Vandenberghe, V.: Convex Optimization. Cambridge University Press, Cambridge (2004)
3. Brito, M., Chavez, E., Quiroz, A., Yukich, J.: Connectivity of the Mutual K-nearest-neighbor Graph in Clustering and Outlier Detection. Statistics and Probability Letters 35(1), 33–42 (1997)
4. Chan, P.K., Schlag, M.D.F., Zien, J.Y.: Spectral K-way Ratio-cut Partitioning and Clustering. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 13(9), 1088–1096 (1994)

5. Fokkema, D.R., Sleijpen, G.L.G., van der Vorst, H.A.: Jacobi–Davidson Style QR and QZ Algorithms for the Reduction of Matrix Pencils. SIAM Journal on Scientific Computing 20(1), 94–125 (1998)
6. Lewis, D.D.: Reuters-21578 text categorization test collection
7. Luo, P., Zhan, G., He, Q., Shi, Z., Lü, K.: On defining partition entropy by inequalities. IEEE Transactions on Information Theory 53(9), 3233–3239 (2007)
8. Roweis, S., Saul, L.: Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science 290(5500), 2323–2326 (2000)
9. Saul, L., Roweis, S.: Think Globally, Fit Locally: Unsupervised Learning of Low Dimensional Manifold. Journal of Machine Learning Research 4, 119–155 (2003)
10. Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press, Cambridge (2002)
11. Shi, J., Malik, J.: Normalized Cuts and Image Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
12. Strehl, A., Ghosh, J.: Cluster Ensembles — A Knowledge Reuse Framework for Combining Multiple Partitions. Journal of Machine Learning Research 3, 583–617 (2002)
13. Sun, J., Shen, Z., Li, H., Shen, Y.: Clustering via Local Regression. In: Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (2008)
14. TREC: Text REtrieval Conference, `http://trec.nist.gov`
15. Wang, F., Zhang, C.: Label Propagation through Linear Neighborhoods. IEEE Transactions on Knowledge and Data Engineering 20(1), 55–67 (2008)
16. Wu, M., Schölkopf, B.: A Local Learning Approach for Clustering. Advances in Neural Information Processing Systems 19 (2006)
17. Yu, S.X., Shi, J.: Multiclass Spectral Clustering. In: Proceedings of the 9th International Conference on Computer Vision (2003)
18. Zha, H., He, X., Ding, C., Gu, M., Simon, H.D.: Spectral Relaxation for K-means Clustering. Advances in Neural Information Processing Systems 14 (2001)
19. Zhao, Y., Karypis, G.: Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering. Machine Learning 55, 311–331 (2004)

Appendix

Proof:

$$\mathcal{E}(\mathbf{Y}) = \sum_{i=1}^{n} \left\| \mathbf{y}_i - \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\mathbf{y}_j \right\|_1 = \sum_{l=1}^{c}\sum_{i=1}^{n} \left| y_{il} - \sum_{j=1}^{n} w_{ij} y_{jl} \right| \tag{13}$$

$$= \sum_{l=1}^{c} \left(\sum_{\mathbf{x}_i \in \pi_l} \frac{|1 - \sum_{\mathbf{x}_j \in \pi_l} w_{ij}|}{|\pi_l|} + \sum_{\mathbf{x}_i \notin \pi_l} \frac{|0 - \sum_{\mathbf{x}_j \in \pi_l} w_{ij}|}{|\pi_l|} \right) \tag{14}$$

$$= \sum_{l=1}^{c} \frac{\sum_{\mathbf{x}_i \in \pi_l} \sum_{\mathbf{x}_j \notin \pi_l} (w_{ij} + w_{ji})}{|\pi_l|} = \sum_{l=1}^{c} \tilde{\mathbf{h}}_l^T \left(\mathrm{Deg}(\mathbf{A}) - \mathbf{A} \right) \tilde{\mathbf{h}}_l \tag{15}$$

$$= \sum_{l=1}^{c} \tilde{\mathbf{h}}_l^T \mathbf{M} \tilde{\mathbf{h}}_l = \mathrm{Trace}\left(\mathbf{H}^T \mathbf{M} \mathbf{H} \right) \tag{16}$$

Clustering with Lower Bound on Similarity*

Mohammad Al Hasan[1], Saeed Salem[1],
Benjarath Pupacdi[2], and Mohammed J. Zaki[1]

[1] Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY
[2] Chulabhorn Research Institute, Laksi, Bangkok, Thailand

Abstract. We propose a new method, called **SimClus**, for clustering with lower bound on similarity. Instead of accepting k the number of clusters to find, the alternative similarity-based approach imposes a lower bound on the similarity between an object and its corresponding cluster representative (with one representative per cluster). SimClus achieves a $O(\log n)$ approximation bound on the number of clusters, whereas for the best previous algorithm the bound can be as poor as $O(n)$. Experiments on real and synthetic datasets show that our algorithm produces more than 40% fewer representative objects, yet offers the same or better clustering quality. We also propose a dynamic variant of the algorithm, which can be effectively used in an on-line setting.

1 Introduction

In many application domains that involve clustering, it is difficult to guess the number of clusters. It is even more challenging in dynamic domains like newsgroups and blogosphere clustering, where the number of topics is typically unknown and may even change.

An alternative to the parameter k that defines the number of clusters is to provide a lower bound, β that defines the desired (minimum) similarity between a cluster member and a representative object in that cluster. This similarity-based formulation has several benefits. Firstly, the lower bound β automatically imposes a similarity bound among the members of a cluster. Secondly, it supports overlapping clusters in a very natural way. Thirdly, this paradigm can easily be adapted to work in a dynamic setting, where new objects are added in or existing objects are removed.

The optimization problem of clustering with a lower bound on similarity is in fact NP-Hard (as we show below). We thus propose a greedy solution, named **SimClus**, that achieves a $O(\log n)$ approximation bound (where n is the size of the object-set). We also show that the approximation bound for the existing best algorithm, *star clustering* [2], is as worse as $O(n)$. We then propose a variant of **SimClus** that is suitable for clustering in a dynamic setting. We experiment with different synthetic and real-world datasets like random graphs and newsgroup

* This work was supported in part by NSF Grants EMT-0829835, and CNS-0103708, and NIH Grant 1R01EB0080161-01A1.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 122–133, 2009.
© Springer-Verlag Berlin Heidelberg 2009

documents. In all experiments, **SimClus** achieves similar or better clustering performance in comparison to Star clustering algorithm with more than 40% fewer representatives. It also outperforms traditional clustering algorithms (k-Medoids, hierarchical clustering) in terms of cluster quality and execution time.

2 Background

Consider a set of objects, $\mathcal{O}$, and a similarity function, $sim : \mathcal{O} \times \mathcal{O} \rightarrow [0, 1]$ such that for any $x \in \mathcal{O} : sim(x, x) = 1$. Assume that the objective is to cluster the objects in $\mathcal{O}$ such that the objects in a cluster are at least β-similar for a user defined $\beta \in [0, 1]$ (with minimum number of clusters). This formulation leads to an interesting graph problem, given as follows: Let $G(V, E)$ be a graph whose vertices are the objects, and an edge $e(u, v) \in E$ implies that the similarity between vertices u and v is at least β. Below we shall refer to it as the (β) *similarity graph*. Now, any clique in this graph can be taken as one cluster in some clustering, since the distances between the elements in a clique would satisfy the required pair-wise similarity constraints. The clustering objective then becomes to cover the entire graph G by a minimum number of cliques. However, this formulation is difficult to solve; in fact, it leads to an NP-Complete problem, named covering a graph by cliques, which can not be approximated in polynomial time [11].

A relaxation of the above problem can be obtained which requires the similarity bound (β) to hold only between the cluster elements and a fixed center object belonging to that cluster. The center object is the representative for the corresponding cluster. Thus if a cluster has m elements, out of all $\binom{m}{2}$ similarities, $m-1$ are guaranteed to be greater than or equal to β. Nevertheless, all the m objects in the cluster would be sufficiently similar due to triangular inequality or other forms of transitive bounds. We call this relaxed formulation the **lower bound similarity clustering (LBSC)** problem, and it is the main focus of this paper. LBSC seeks exactly one cluster center (representative object) for every cluster. Thus a center object c together with all the objects s such that $sim(c, s) \geq \beta$, form a cluster. All objects s that satisfy the above inequality are called β-similar objects with respect to the object c. If s is β-similar to multiple centers, it belongs to multiple clusters. Thus this model naturally supports overlapping clusters. Since every object s belongs to at least one cluster, s is β-similar to at least one representative object, say c. In that case, we say that c *covers* s. Thus, the clustering objective is to cover all the objects with the smallest number of center objects. A center always covers itself. Unfortunately, the optimization task of LBSC is also NP-Hard, since the corresponding decision problem is NP-Complete.

Lemma 1. *For a given collection of objects $\mathcal{O}$, and a user-defined similarity threshold β, to determine whether there exists a set of representative objects, $\mathcal{C} \subseteq \mathcal{O}$, of size k is NP-Complete.*

Proof: Given graph $G = (V, E)$, the vertex dominating set problem, a known NP-Complete problem [7], asks whether there exists a subset $C \subseteq V$ (called

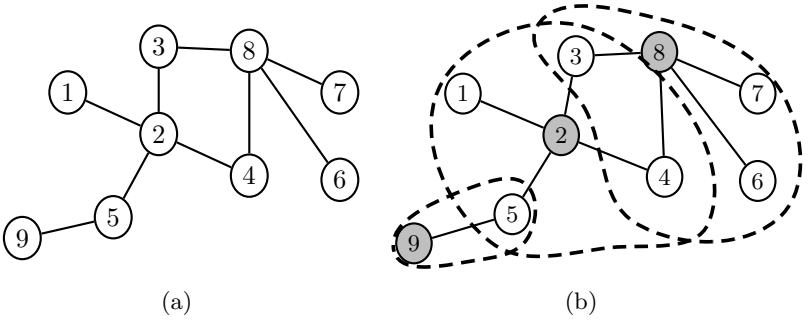


Fig. 1. Lower bound similarity clustering. (a) β-similar graph. (b) overlapping clusters

dominating set), of size at most k, such that every vertex in $V \setminus C$ is adjacent to at least one element in C. If G is the β-similarity graph then each center set $\mathcal{C}$ corresponds to a dominating set, since every object in $\mathcal{O}$ is either in the center set or is covered by at least one cluster center. ■

The above relation to the dominating set problem suggests a graph-based formulation. From $\binom{n}{2}$ similarity values for a set of n objects, we can first construct a β-similarity graph. Then, in this graph we need to find a vertex dominating set, which would constitute the representative center set. Figure 1(a) shows an example β-similarity graph. Figure 1(b) shows a LBSC of these objects with 3 clusters (dotted closed curves, with gray circles as center objects).

2.1 Related Work

LBSC did not get much attention in the past. Star clustering [2] is the leading algorithm in this paradigm (we discuss it in detail below). [6] highlighted some of the problems of the Star algorithm and proposed an alternative solution to improve it, however they did not provide any approximation bounds. Furthermore, they considered only the static scenario. In terms of dynamic clustering, [3] propose an online version of the popular EM-based approach. [1] solves it by using an online component and an off-line component, whereas [5] proposed an online variant of different mixture model clustering for text data. One may consider the streaming model as dynamic clustering, however, this model imposes very strict requirements [4] on available memory and the number of passes over the data. Hence, the approximation quality is generally poor.

Star Clustering: Aslam et. al. [2] proposed a greedy solution to LBSC called *Star clustering*. It sorts the vertices in the descending order of degree; it then selects the first vertex in the sorted order as one of the cluster centers. Any other vertex covered by this one is deleted from the sorted list and the procedure is repeated until all the vertices are covered. Provided the similarity matrix is given, the computational complexity of this method is $O(|V| \lg |V|)$. Figure 1(b)

shows an example of Star clustering. The greedy solution of the Star algorithm is obviously not optimal. Moreover, the approximation bound can be very bad.

Lemma 2. *The greedy Star clustering algorithm generates an independent set.* PROOF: Once it selects an object as a center-node, it removes all adjacent (covered) nodes. In later iterations they never appear in the dominating set, thus it is an independent set. ■

The independent set restriction is the main reason why Star clustering yields unnecessarily many star-centers. In Figure 2, we show an example that illustrates the poor bound of the Star clustering algorithm. Instead of choosing $\{2, 8\}$ as center-nodes (in gray), Star algorithm chooses $\{1, 3, 4, 5, 8\}$, to respect the independence set condition. This example can be generalized to obtain a bound in an asymptotic sense. Assume, we have n vertices arranged in two star-shapes (as in Figure 2), each with $\frac{n}{2} - 1$ satellites and one star-center. Assume further that the two star-centers are adjacent to each other. In this case, Star clustering would return $\frac{n}{2}$ centers, whereas the optimal LBSC would have only 2 centers; thus the approximation ratio can be as bad as $O(n)$.

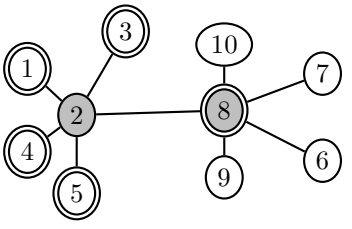


Fig. 2. Star clustering: poor bound

3 SimClus: Lower Bound Similarity Clustering

We now describe our set cover formulation for LBSC, named **SimClus**. It is also greedy in nature, but with a better $O(\log n)$ approximation bound. It thus produces fewer cluster centers, and better representative objects for each cluster.

Lemma 3. *LBSC can be reduced to a set cover problem.*

PROOF: Let $G = (V, E)$ be the β-similarity graph. For each vertex $u \in V$, we construct a new set $s_u = \{u\} \cup \{v : (u, v) \in E\}$. Let $\mathcal{S} = \{s_u : u \in V\}$. Obviously, $|\mathcal{S}| = |V|$ and there exists a one-to-one correspondence between the elements in $\mathcal{S}$ and the vertices in V. The set cover problem is to find the smallest set of elements $\mathcal{T} \subseteq \mathcal{S}$, that covers all the vertices, i.e., $\bigcup_u \{S_u : u \in \mathcal{T}\} = V$). If for each $s_x \in \mathcal{T}$, we insert the corresponding vertex $x \in V$ in the center-set $\mathcal{C}$, then $\mathcal{C}$ is also an optimal solution for the LBSC problem, since the minimality of $|\mathcal{T}|$ ensures the minimality of $|\mathcal{C}|$. ■

The set-cover problem is NP-Complete, but it can be solved using an efficient greedy algorithm that achieves $O(\log n)$ approximation bound [10]. The greedy heuristic iteratively chooses the set that covers the largest number of uncovered elements until all the elements are covered. Using the same heuristic for LBSC then yields the $O(\log n)$ approximation bound with much fewer representatives. For the dominating set problem, the hardness of approximation result is also

available, that proves that unless $P = NP$, no polynomial time algorithm can have better than $O(\log n)$ approximation [9]. So, it is highly unlikely that a better bound then $\log n$ can be achieved for LBSC.

The main difference between Star clustering and SimClus is that instead of choosing the uncovered object with the highest degree (as done in Star), SimClus chooses the object that can cover the most uncovered elements. This drastically reduces the number of clusters; the clusters are more dense and hence, more informative.

SimClus follows the reduction shown in Lemma 3. At the beginning, all the objects are not covered and the center-set is empty. We then map each vertex, u of LBSC similarity graph to a set s_u that contains itself and all objects that it can cover. To facilitate the greedy heuristic, the set $s_u, \forall u \in V$ contains only those objects that are uncovered. So, we call the set s_u as the *uncovered cover-set* of u, which intuitively means that it holds those objects that are uncovered and can be covered by choosing u as a center. Hence, once an object is chosen as a center, the *uncovered cover-set* of all the non-center objects are updated by removing any object that is covered by the newly chosen center. In every iteration a new center is selected using the above greedy criterion and the process is repeated until all the objects are covered. If there is a tie in the above criterion, we break the tie by selecting the object that has the largest degree in the similarity graph. If there is still a tie, we break it arbitrarily.

Example: Consider the similarity graph in Figure 2. The uncovered cover-set of vertex 1 is $\{1, 2\}$, and for vertex 2 it is $\{1, 2, 3, 4, 5, 8\}$. At the beginning, the uncovered cover-set of both the vertices 2 and 8 have the same size. They also have equal degrees. So, we arbitrarily choose, say, vertex 2 as a center. After updating the uncovered cover-set for all the objects, we have empty uncovered cover-set for $1, 3, 4$ and 5 and only one object for $6, 7, 9$ and 10. But, the uncovered cover-set for 8 contains $\{6, 7, 9, 10\}$. So, 8 is selected as the next center. At this point all the objects are covered, and the algorithm terminates by returning $\{2, 8\}$ as representatives.

Complexity Analysis: SimClus computes the entire similarity matrix to find the β-similarity graph. So, the overall complexity of the algorithm is at least $O(|V|^2)$. If we consider that the similarity matrix is given, then the overall worst-case complexity can be computed in terms of the number of the centers obtained. Assume, that the final clustering has k centers. In each iteration, the best center can be obtained in $O(\lg |V|)$ time by using a max_heap data structure. The cover-set update affects only those vertices that are adjacent to the newly covered objects. For the sake of worse case analysis, if we assume that to be $O(|V|)$, the worst case complexity of the algorithm is $O(k|V|)$.

4 Dynamic SimClus: Dynamic Similarity Clustering

The static algorithm that is provided in the previous section requires that the entire β-similarity graph is available before the algorithm is applied. However,

in many practical application scenarios, this requirement does not hold. For information retrieval, new documents can be added or old documents may be deleted from the repositories and the clustering may need to be updated. One option is to re-cluster the objects by running the static clustering algorithm for every change in the collection. But, for most of the cases, the changes are only local; so re-computing the entire clustering is wasteful. Also note that re-computation dramatically changes the cluster-centers; so if the objective of LBSC is to find representative objects in a dynamic setting, one may prefer the dynamic algorithm over the static algorithm, since the former retains a large portion of the representative set. It is also useful in adopting a lazy update scheme for a static version. Since, the dynamic version is generally worse in terms of number of clusters compared to the static version (the $O(\log n)$ approximation guarantee may not hold), a static update may follow after a batch of dynamic updates to re-optimize the clustering.

Dynamic SimClus: In a dynamic environment, we assume that we have an initial solution for the lower bound similarity clustering. New requests for insertion or deletion of an object come in an arbitrary manner. We need to satisfy the requests efficiently while maintaining the minimality of the representative set as much as possible. We are allowed to change the status of an object in either direction (a center can be made non-center and vice-versa). It is worth pointing that this model is not the same as the *online model* that is studied in theoretical computer science. In the online model, the decision that is made based on present information cannot be altered later when further information is available at future. So in a pure online setting, if a node is made representative, its status cannot be changed later. Nevertheless, the online version of LBSC is very difficult; in fact its *competitive ratio* is as worse as $O(n-1)$ [8].

Since the dynamic model assumes that a valid solution to the lower bound similarity clustering exists, addition of new objects cannot be made based on the greedy criteria of the static algorithm, as only one object (the new object) is uncovered and all vertices adjacent to it have exactly the same size (one) for the *uncovered cover-set*. So, we propose to obtain cluster-centers in such way that the following three conditions are satisfied.

1. Every object is either a cluster-center or is adjacent to at least one cluster center.
2. A cluster center with $degree > 0$ must be adjacent to at least one non-center object of smaller or equal degree.
3. No cluster-center is redundant, i.e. every cluster-center covers at least one object exclusively.

Note that the first of the above conditions is from the definition of LBSC. However, the second and the third conditions are chosen for LBSC to have a reasonably good solution in a dynamic setting. In the experimental section, we shall show that the above dynamic algorithm generates less centers in comparison to both static and dynamic Star clustering algorithm.

Inserting a new object: A center is called *illegal* if it does not satisfy condition (2) above, i.e. all its adjacent vertices have degree strictly higher than it. A center is called *redundant*, if its removal does not change (increase) the number of nodes that need to be covered. The dynamic SimClus accepts as input the similarity graph and the *covered* vector, which stores the current coverage of each existing vertex. Once a new object v is added, the adjacency list of similarity graph G is updated. If v is adjacent to any existing center, it is properly reflected in the *covered* vector. Now, we have two different cases to consider.

In the first case, when v is covered, condition 1 is already satisfied. Then, we check condition 2 (illegal center) for all the centers that are adjacent to v. Since addition of new vertices can change the adjacency list of some of the existing vertices, this may change a legal center into an illegal one. If this check succeeds (some illegal neighboring center is found), we make the illegal center as non-center and v becomes a center. This step is called recursively on success. In case the above step fails (v is not a center yet), we test whether v should be a center as it has strictly higher degree than any of its neighboring centers. Note that this step is not essential for correctness according to our conditions, but this heuristic has potential to generate better centers.

For the second case, when v is not covered, then none of its adjacent vertices is a center and to fulfill the coverage requirements, at least one of these vertices should be a center. We first test whether it should be some $x \in adj(v)$ by checking whether it has any illegal neighbors. If not, then we choose the vertex with the highest degree as a center.

When a node is made a center, some auxiliary updates are required. First, redundancy check is required for all other centers that are adjacent to it to satisfy the condition 3. Moreover, some adjacent non-centers can also become a center as one of its neighboring center becomes illegal. Figure 3 shows an example of insertion, where we explain the steps of the insertion algorithm.

Deleting an existing object: Deletion is comparably easier. Like insertion, we first update the adjacency and covered vector as necessary. Then, we consider the following two cases. The first is when the deleted node v was a center. In that case, we need to check if any of its adjacent vertex becomes isolated. All of those become centers. For the remaining, if they are still covered, we return immediately. Assume x is not covered, then we cover x by making the highest degree vertex (among itself and its neighbors) a center. For the second case, when v was not a center, its removal does not violate condition (1), so we check for condition (2) and (3) by calling the same methods as in insert routine.

Complexity Analysis: Complexity of dynamic SimClus is difficult to analysis, since it depends on the degree of the vertices that are adjacent to the inserted (or deleted) vertex. For the sake of average-case analysis, we assume Erdos-Renyi random graph model, where the (expected) degree of a node is equal to $d = p.|V|$; p is the probability of an edge to exist. The redundancy checking of a node takes $O(d)$ time as it just reads the *covered* array of its adjacent nodes. Similarly, the illegal center condition checking of one node also takes $O(d)$ time, as it reads the

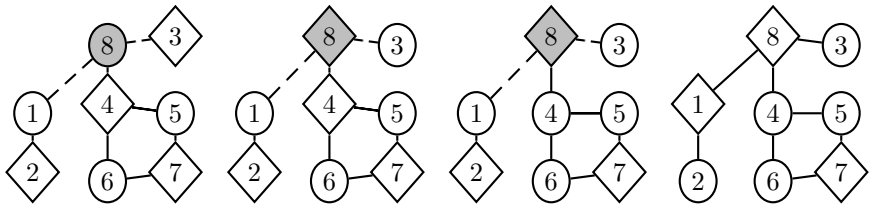


Fig. 3. Insertion example; a diamond shape indicates a center. (a) after inserting node 8, node 3 is illegal center. (b) node 3 becomes non-center and node 8 becomes a center; now, node 4, a center-neighbor of 8 is redundant. (c) redundant center removed; a non-center neighbor of 8, namely 1, has an illegal center 2 as neighbor. (d) Final result; an off-line algorithm will produce identical result.

degree of the adjacent nodes. So, considering the above check for all the nodes adjacent to the inserted (deleted) node, the total cost is $O(d^2)$ But, this check is done recursively on success. If the recursion is successful for I times, the total cost is $O(Id^2)$. But, generally the value of I is very close to $O(1)$ as we show next.

From the definition, an illegal center has the smallest degree among all its neighbors. Assume u is a node, $deg(u)$ is its degree and $adj(u)$ is its adjacency list. The probability that u is an illegal node is given as:

$$
\begin{aligned}
&= \sum_{i=1}^{n-2} P\left[\forall v \in adj(u) : deg(v) \geq i+1 \mid deg(u) = i\right] \\
&= \sum_{i=1}^{n-2} \left(P\left[deg(v) \geq i+1 | v \in adj(u)\right]\right)^i \cdot P\left[deg(u) = i\right] \\
&= \sum_{i=1}^{n-2} \left[\left(\sum_{k=i+1}^{n-2} \binom{n-2}{k} p^k (1-p)^{n-k-2} \right)^i \cdot \binom{n-1}{i} p^i (1-p)^{n-i-1} \right]
\end{aligned}
$$

The probability that the node u has degree i is equal to $\binom{n-1}{i} p^i (1-p)^{n-i-1}$ and the probability that a node $v \in adj(u)$ has degree greater than i is equal to $\sum_{k=i+1}^{n-2} \binom{n-2}{k} p^k (1-p)^{n-k-2}$. For a node to be illegal, all its neighbors should have greater degree. Since an illegal node can have degree from 1 to $n-2$, the outer sum aggregates these probabilities. If the above probability is p_{repeat}, using geometric distribution, the average value of I is $\frac{1}{1-p_{repeat}}$. For all possible values of p, the value for I is less than 2 (found with direct calculation in Matlab) when $n >= 10$. For example, the highest value of I from n is 10 is around 1.75 and for $n = 1000$, it is around 1.015 and with larger n, the value diminishes further. Intuitively, this means that for random graphs, every node is generally connected to some vertices with higher and some vertices with lower degree; hence, very few nodes are illegal. For real-life graphs, the constant may be a bit higher. So, considering I a constant, the complexity of dynamic algorithm for one insertion or deletion is $O(d^2)$.

5 Experiments

The objective of the experiments is to show the following. Firstly, that SimClus returns a smaller number of representatives for LBSC, yet it yields similar quality as Star clustering. Secondly, that the representative objects are of good quality and can generate clustering that is better than the traditional clustering algorithms. Finally, that the LBSC approach is particularly suitable for clustering multi-label dataset.

Besides SimClus and Star, the experiments use two other traditional clustering algorithms: k-Medoids, and hierarchical (UPGMA). For all the algorithms, the document similarity is modeled by using cosine similarity and a similarity matrix is provided as input. For k-Medoids, we use our own implementation which works very similar to k-Means, but in each iteration, when k-Means chooses a new center by averaging the vectors in that cluster, it chooses it by finding the object with the best average similarity to all the objects in that cluster (thus it has a quadratic time complexity). For hierarchical clustering, we use the CLUTO implementation (`glaros.dtc.umn.edu/gkhome/views/cluto/`). The Star algorithm was implemented as described in [2].

5.1 Synthetic Data: Random Graphs

The first experiment considers synthetic data, in the form of random similarity graph of various size and types for both static and dynamic scenario. For the dynamic experiments, we shuffle the vertices of the similarity graphs randomly and insert them in the existing graph (staring from an empty graph) in that order. With a small probability (0.05), we also delete a random vertex after every insertion. For random graph type, we considers the following two models: (1) Erdos-Renyi and (2) Power-law graphs. Table 1 shows the center-set sizes. For both static and dynamic versions, SimClus achieves a smaller number of clusters in comparison to Star. Specifically, the static version significantly outperforms the static (and dynamic) version of Star. As expected, dynamic SimClus does not perform as good as the static version, yet it performs better than Star.

Table 1. Performance on random graphs

vertex	edge	Number of clusters			
		SimClus		Star	
		static	*dynamic*	*static*	*dynamic*
Erdos-Renyi random graphs					
1000	5000	144	194	209	209
1000	10000	84	123	126	130
1000	100000	16	22	21	24
10000	50000	1424	1951	2066	2068
10000	100000	840	1227	1323	1311
10000	1000000	147	225	247	245
Power Law graphs					
1000	5000	547	564	579	579
1000	10000	525	539	550	549
1000	100000	496	502	502	503
10000	50000	3285	3498	4015	4024
10000	100000	2734	2928	3340	3340
10000	1000000	1991	2094	2156	2156

5.2 Newsgroup Dataset

For a real-life dataset, we chose the Twenty-newsgroup data from the UCI Repository (`www.ics.uci.edu/~mlearn`). This dataset is interesting for our experiments, as it has a set of documents that have multiple labels. We used the rainbow package (`www.cs.cmu.edu/~mccallum/bow`) to convert the documents into word vectors and then used the best 100 words (in terms of mutual information gain statistics) as feature vectors. We discarded any document that did not have any of the top 100 words. The final dataset had 16701 documents. Out of those, 16199 had a unique label, 497 documents had 2 labels, and only 5 documents had more than 2 labels. There were 20 different labels in total.

In this experiment, we cluster the documents using different clustering algorithms and compare the clustering performance using supervised performance metrics, like precision, recall and F-measure. For k-Medoids and Hierarchical (UPGMA) we set the k value to be 20 to obtain 20 different clusters; then for every cluster, we use majority voting to label the cluster. For any object with multiple labels, its vote is counted for all its labels. In case of SimClus and Star, we cannot use k, so we cluster the documents by two different similarity thresholds (we used, $\beta = 0.5$ and $\beta = 0.75$). Number of clusters obtained are generally higher than 20 (exact values are shown in the k column). As the similarity threshold increases, the number of clusters also increases. Then we classify each of these clusters as the label of the cluster representative. Note that many representatives (thus many clusters) can have the same class-label; all of those clusters are collectively classified with that label. If the representative object has multiple labels, all the objects adjacent to that representative objects gets multiple labels. An object can also get multiple labels by being adjacent to multiple representatives with different class-labels. While computing precision/recall for multi-label objects, if any of the predicted labels match with any of the actual labels, the object is considered as a true positive.

Table 2 shows the results. Compared to k-Medoids and Hierarchical, both Star and SimClus achieve much better performance using F-score measure, because of the very high recall that these two achieve. The possible reason is that, instead of choosing exactly 20 centers (which may not be enough for the dataset, since the cluster boundary is not regular), they sample enough representatives to cover the entire cluster. So, a document has much higher chance to be a neighbor of one of the many centers that matches with its class label. For example, if a document has a label "comp.graphics", out of, say 500 representatives, roughly $500/20 = 25$ representatives may have a label "comp.graphics". So, the document has a much

Table 2. Comparison on newsgroup dataset

algorithm	parameters		average	average	F-Measure
	β	k	precision	recall	
k-Medoids	-	20	59.80	40.21	48.09
UPGMA	-	20	48.37	37.33	42.14
Star	0.50	125	53.50	85.96	65.95
	0.75	1060	66.83	83.79	74.36
SimClus	0.50	77	51.83	82.38	62.06
	0.75	687	70.00	82.79	75.86

higher probability to be a neighbor of any of these. Comparing Star and SimClus, their F-values are similar; for smaller β, the former and for larger β, the latter has marginally better F-values. However, SimClus chooses 40% less centers compared to the Star algorithm.

Our third set of experiments compares the ability of SimClus and Star to predict multiple labels. For this, we find the predicted labels of the objects that has more than one actual labels (502 documents qualify). An actual multi-label is considered as recalled if at least two predicted labels matches with two of the actual labels. Table 3 shows the result for three different values of similarity threshold. The multi-recall columns of the corresponding algorithms show their respective recalls in percentage. The recall values for both the algorithms drop with increasing threshold values. The reason is with large threshold β-similarity graphs become more and more sparse, so an object is not connected to many representatives. Thus the ability to predict multiple labels diminishes. In comparison to Star, **SimClus** performs substantially better than Star. For example, for a β value of 0.60, even with 47% less centers. As β increases, it suffers somewhat because it optimizes the number of centers. In fact, as we investigated we found that Star achieved better multi-recall, by actually selecting many of the multi-labeled nodes as the center objects.

Table 3. Overlapping clustering performance

similarity threshold (β)	**SimClus**		**Star**	
	center count	multi-recall	center count	multi-recall
0.60	144	76.49	273	48.41
0.70	382	41.63	550	45.80
0.80	1274	28.88	1811	31.47

We also compare the timing performance of SimClus in comparison to other algorithms. The result is shown in Table 4. The table shows only the execution time (in seconds) of the clustering task; the I/O cost of loading the similarity matrix is excluded for the sake of fair comparison. The timings were obtained using a 2.1 GHz machine with 2GB RAM with Linux OS. For this experiment, we randomly select documents from the newsgroup dataset to make 3 different smaller-size datasets (4000, 8000, 12000). Besides k-Medoids, and UPGMA we also use another algorithm, RB (from the Cluto software) which is a fast partitional algorithm that uses repeated bisection. For k-Medoids, UPGMA and RB, which require a k value to cluster, $k = 20$ is used as it is the natural number of clusters for the newsgroup dataset. For Star and SimClus, timing for two different similarity thresholds (0.5 and 0.75) are reported. Different thresholds account for different number of edges in the β-similarity graphs. From Table 4, we see that Star is the fastest among all the algorithms as it just needs to sort the vertex-set only once based on degree and its timing varies

Table 4. Execution time comparison

algorithm		Size			
	k or β	4000	8000	12000	16701
k-Medoids	20	33.36	301.21	1200.12	3100.38
UPGMA	20	6.71	33.48	75.76	216.96
RB	20	25.64	116.68	297.88	677.28
Star	0.50	0.02	0.04	0.06	0.09
(static)	0.75	0.02	0.04	0.05	0.08
SimClus	0.50	5.09	27.62	68.68	197.49
(static)	0.75	0.50	2.53	6.78	15.22

negligibly based on the number of edges. SimClus is the second fastest and its timing varies based on the number of edges in the similarity graph since the time complexity has a term related to $|E|$. The execution time of **SimClus** for 0.5 threshold (which is very relaxed for LBSC) with the entire newsgroup dataset is 1.1, 3.4, and 15.7 times better than UPGMA, RB, and k-Medoids respectively. We also compared the execution time of our dynamic algorithm to evaluate its utility over the static algorithm. For the newsgroup dataset, average insertion time of one document is 0.13 second, which is much faster in comparison to the re-clustering time (15.22 seconds) using the static algorithm.

6 Conclusions

In this paper we proposed a clustering algorithm that uses lower bound on similarity to cluster a set of objects from the similarity matrix. Experiments on real-life and synthetic datasets show that the algorithm is faster and produces higher quality clustering in comparison to existing popular algorithms. Furthermore, it provides representative centers for every cluster; hence, it is effective in summarization or semi-supervised classification. It is also suitable for multi-label or dynamic clustering.

References

1. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: VLDB Proceedings (August 2003)
2. Aslam, J., Pelekhov, J.E., Rus, D.: The star clustering algorithm for static and dynamic information organization. Graph Algorithms and Application 8(1), 95–129 (2004)
3. Azoury, K.S., Warmuth, M.K.: Relative loss bounds for on-line density estimation with the exponential family of distributions. Machine Learning 43(3), 211–246 (2001)
4. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and issues in data streams. In: ACM SIGMOD-SIGACT-SIGART symposium on Principles of Database Systems. ACM, New York
5. Banerjee, A., Basu, S.: Topic models over text streams: A study of batch and online unsepervised learning. In: SIAM Data Mining (2007)
6. Gil-García, R.J., Badía-Contelles, J.M., Pons-Porrata, A.: Extended Star Clustering. In: Sanfeliu, A., Ruiz-Shulcloper, J. (eds.) CIARP 2003. LNCS, vol. 2905, pp. 480–487. Springer, Heidelberg (2003)
7. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, New York (1979)
8. King, G., Tzeng, W.: On-line algorithm for the dominating set problem. Information Processing Letters 61, 11–14 (1997)
9. Lund, C., Yannakakis, M.: On the hardness of approximating minimization problems. Journal of the ACM 41(5), 960–981 (1994)
10. Vazirani, V.V.: Approximation Algorithms. Springer, Heidelberg
11. Zuckerman, D.: Np-complete problems have a version that's hard to approximate. In: Proc. of Eighth Annual Structure in Complexity Theorey, pp. 305–312. IEEE Computer Society, Los Alamitos (1993)

Approximate Spectral Clustering*

Liang Wang, Christopher Leckie, Kotagiri Ramamohanarao, and James Bezdek

Department of Computer Science and Software Engineering
The University of Melbourne, Parkville, Victoria 3010, Australia
{lwwang, caleckie, rao}@csse.unimelb.edu.au, jbezdek@cs.uwf.edu

Abstract. While spectral clustering has recently shown great promise, computational cost makes it infeasible for use with large data sets. To address this computational challenge, this paper considers the problem of approximate spectral clustering, which enables both the feasibility (of approximately clustering in very large and unloadable data sets) and acceleration (of clustering in loadable data sets), while maintaining acceptable accuracy. We examine and propose several schemes for approximate spectral grouping, and make an empirical comparison of those schemes in combination with several sampling strategies. Experimental results on several synthetic and real-world data sets show that approximate spectral clustering can achieve both the goals of feasibility and acceleration.

Keywords: Spectral clustering, scalability, matrix approximation, sampling.

1 Introduction

As an exploratory data analysis tool, clustering aims to group objects of a similar kind into their respective categories (see [1] for a comprehensive survey). Given a data set $\mathcal{O}$ comprising n objects $\{o_1, o_2, \cdots, o_n\}$, (crisp) clustering partitions the data into c groups $G_1, G_2, \cdots, G_c$, so that $G_i \cap G_j = \emptyset$ if $i \neq j$ and $G_1 \cup G_2 \cup \cdots \cup G_c = \mathcal{O}$. In particular, pairwise grouping methods, such as spectral clustering [2], present an appealing alternative to traditional central grouping techniques (such as K-means), because 1) they are applicable to situations in which the objects are not naturally representable in terms of feature vectors; and 2) they avoid the assumption that all examples in a cluster must be close to a prototype. This means they are amenable to irregular-shaped clusters.

Spectral clustering algorithms usually rely on the eigendecomposition of a $n \times n$ similarity matrix (where n is the number of examples), which generally takes $\mathrm{O}(n^3)$ time and $\mathrm{O}(n^2)$ space complexity. In addition, to obtain such a similarity matrix it is necessary to compare all possible pairs of examples, which is computationally expensive for a large data set. These limitations make spectral clustering methods impractical (or computationally infeasible) when handling large data sets. Additional strategies are thus required to adapt to growing

* This work was supported by ARC Discovery Project DP0663196.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 134–146, 2009.
© Springer-Verlag Berlin Heidelberg 2009

data sizes while maintaining both cluster quality and speed. A spectral grouping approach based on the Nyström approximation was proposed in [3], which first solves a small-scale eigendecomposition problem on randomly chosen sample data, then computes approximated eigenvectors via extrapolation. Another incremental spectral clustering algorithm was proposed to handle "dynamic" evolving data in [4] by introducing the incidence vector/matrix. In [5], the spectral clustering algorithm is parallelized across distributed machines. However, these two methods either sacrifice accuracy or require distributed computing infrastructure. In contrast, this paper considers approximate spectral clustering (ASC) for "static" data without the use of distributed computation.

The motivations of approximate spectral clustering can be described as follows. When the data set is large and unloadable on the available computing platform, ASC provides an approximate solution to clustering (i.e., making clustering feasible), whereas it is impossible to use a literal clustering approach in batch mode on such data. If the data set is small, medium, or merely large but still loadable, then ASC may offer an approximation comparable to the literal solution but at a significantly reduced computational cost. In summary, the benefits of an approximate clustering scheme are "feasibility" for very large data sets and "acceleration" for manageably-sized data sets. To this end, this paper proposes two new methods for approximate spectral clustering, as well as the examination of an existing method described in [3]. One of them is based on matrix approximation, and the other uses a "sampling plus extension" approach. Our major contributions are as follows: 1) we present two different schemes for approximate spectral clustering; 2) we provide a comprehensive quantitative comparison of several approximate spectral clustering algorithms, together with a comparison of several sampling schemes; and 3) we provide extensive experimental results on synthetic and real data sets, and several meaningful conclusions are highlighted.

The rest of this paper is organized as follows. Section 2 gives a brief review of spectral clustering. Section 3 details several approximate algorithms. Section 4 introduces four kinds of sampling schemes. The results are presented in Section 5, prior to discussion and conclusion in Section 6.

2 Spectral Clustering

Spectral methods for clustering, e.g., normalized cut [6] and max-min cut [7], are based on the eigenvectors and eigenvalues of a symmetric positive semidefinite (SPSD) matrix of size $n \times n$ derived from a given data set. Let the symmetric $W \in \mathcal{R}^{n \times n}$ denote the weighted adjacency matrix for a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V}$ representing the n objects in $\mathcal{O}$ to be analyzed and edges $\mathcal{E}$ whose weights capture pairwise affinities between objects. Let D be a diagonal matrix with entries $D_{ii} = d_i$, where $d_i = \sum_j W_{ij}$ denotes the degree of the ith node, then the graph Laplacian matrix is defined as $L = D - W$ [8].

Let C_1 and C_2 be a bipartition of $\mathcal{V}$, i.e., $C_1 \cap C_2 = \emptyset$ and $C_1 \cup C_2 = \mathcal{V}$, and the volume of a set as the sum of the degrees within the set, i.e., $\mathrm{vol}(C_j) = \sum_{i \in C_j} d_i$.

The normalized cut between sets C_1 and C_2 is defined as [6]

$$\text{ncut}(C_1, C_2) = \frac{2 \cdot \text{cut}(C_1, C_2)}{\text{vol}(C_1) \parallel \text{vol}(C_2)} \tag{1}$$

where $\parallel$ denotes the harmonic mean and $\text{cut}(C_1, C_2) = \sum_{i \in C_1, j \in C_2} W_{ij}$. To minimize (1), Shi and Malik [6] showed that an approximate solution may be obtained by thresholding the eigenvector corresponding to the second smallest eigenvalue of the normalized Laplacian matrix $\mathcal{L}$, i.e.,

$$\mathcal{L} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2} \tag{2}$$

The matrix $\mathcal{L}$ is positive semidefinite, even when W is indefinite.

Extensions to multiple groups are possible, e.g., using multiple eigenvectors [9]. In this work we adopt this approach by computing the leading eigenvectors V from $(D^{-1/2} W D^{-1/2})V = V\Lambda_\lambda$. These eigenvectors induce an embedding of the objects in a low-dimensional subspace, in which the K-means clustering algorithm is then used to discover final partitions by grouping columns of V.

3 Approximate Spectral Clustering Algorithms

To address the complexity of spectral decomposition for large n, we now outline several approximate approaches to the problem. Spectral clustering generally deals with an $n \times n$ SPSD matrix, say M, which can be decomposed as $M = U\Sigma U^T$ with Σ the eigenvalues of M and U the associated eigenvectors. Suppose $m \ll n$ columns of M are sampled without replacement. Let A be the $n \times m$ matrix of these sampled columns, and S be the $m \times m$ matrix consisting of the intersection of these m columns with the corresponding m rows. Without loss of generality, we can rearrange the columns and rows of M such that

$$M = \begin{pmatrix} S & B \\ B^T & C \end{pmatrix} \text{ with } A = \begin{pmatrix} S \\ B^T \end{pmatrix} \tag{3}$$

where $B \in \mathcal{R}^{m \times (n-m)}$ contains the elements from the samples to the rest of the objects, and $C \in \mathcal{R}^{(n-m) \times (n-m)}$ contains the elements between all of the remaining objects. In the case of $m \ll n$, C is usually large.

3.1 nSPEC

The Nyström approximation has recently been studied in the machine learning community, e.g., for fast approximate Gaussian process classification [10] and low-rank approximation to the kernel matrix [11]. A spectral grouping method based on the Nyström approximation was proposed for image segmentation in [3]. We refer to this as nSPEC (Nyström-based Spectral Clustering).

The Nyström approximation uses S and A to approximate M as

$$M \approx \tilde{M} = AS^+ A^T \tag{4}$$

where '+' is the pseudoinverse. The Nyström approximation models C by BS^+B^T, and the resulting approximate eigenvalues and eigenvectors of M are

$$\tilde{\Sigma} = (\frac{n}{m})\Sigma_S \text{ and } \tilde{U} = \sqrt{\frac{m}{n}} A U_S \Sigma_S^+ \tag{5}$$

where $S = U_S \Sigma_S U_S^+$ [10].

The eigenvectors generated from the Nyström approximation are not exactly orthogonal because they are extrapolated from the eigenvectors of S, losing orthogonality in this process. If S is positive definite, Fowlkes et al. [3] used a one-shot method to solve for the orthogonalized approximate eigenvectors. Define $Q = S + S^{-1/2}BB^TS^{-1/2}$ and diagonalize it as $Q = U_Q \Lambda_Q U_Q^T$, then $\tilde{M}$ can be diagonalized as $\tilde{M} = \hat{U}\Lambda_Q\hat{U}^T$ with

$$\hat{U} = AS^{-1/2}U_Q\Lambda_Q^{-1/2} \tag{6}$$

3.2 cSPEC

An alternative column-sampling technique has been analyzed in the theoretical computer science community [12], which can also be used to approximate spectral decomposition of a large matrix using a subset of columns. The column-sampling method was initially introduced to approximate SVD (Singular Value Decomposition) for any rectangular matrix [12,13]. However, it has not yet explicitly been used in spectral clustering. Here we use it to approximate the spectral decomposition of M, and call the resulting method cSPEC (Column-sampling Spectral Clustering).

The column-sampling technique approximates the eigendecomposition of M by using the SVD of A directly. Suppose $A = U_A \Sigma_A V_A^T$, then the approximate eigenvectors and eigenvalues of M are given by the left singular vectors of A and the corresponding scaled singular values, i.e.,

$$\tilde{\Sigma} = \sqrt{\frac{n}{m}}\Sigma_A \text{ and } \tilde{U} = U_A = AV_A\Sigma_A^+ \tag{7}$$

Accordingly the approximation of M can be written as [14]

$$M \approx \tilde{M} = A\left(\sqrt{\frac{m}{n}}(A^TA)^{1/2}\right)^+ A^T \tag{8}$$

which has a very similar form to (4). When n is very large, the SVD on A directly is still quite demanding. Fortunately, we have

$$A^TA = V_A\Sigma_A^TU_A^TU_A\Sigma_AV_A^T = V_A\Sigma_A^2V_A^T \tag{9}$$

It is thus easy to obtain U_A and Σ_A by computing the SVD on $A^TA \in \mathcal{R}^{m\times m}$. Relatively, A^TA can be easily computed even for large n.

3.3 eSPEC

Another way to attack the scalability problem is "extensibility" (e.g., [15,16]). An extended scheme applies a clustering algorithm to a representative sample set, and then extends the sample result to obtain (approximate) clusters for the remaining data. Also, the extended clustering schemes can be effectively applied in cases in which data are collected sequentially. Here we propose a "sampling, clustering plus extension" solution, called eSPEC (Extensible Spectral Clustering), for approximate spectral clustering.

After the sample data S is obtained, we first use a literal spectral clustering algorithm to group them. Next we address the problem of out-of-sample extension, i.e., to assign each of the remaining $n-m$ objects to one of the c previously determined groups. We regard the whole $m \times n$ matrix (i.e., A^T) as a semantically-meaningful vectorial representation (*i.e.*, each object has m attributes, each of which corresponds to a similarity relation between the object and one of the m sample objects, leading to a virtual data set $\{x_i\}_{i=1}^n$). For learning the cluster-preserving embedding space from S, we adopt computationally-efficient locality preserving projection (LPP) [17], which is very similar to the mapping procedure used in spectral clustering algorithms.

Let $\mathcal{G}_\mathcal{S}$ denote a graph with m nodes corresponding to the m labeled samples $\{x_j^S\}_{j=1}^m$. An edge occurs between nodes i and j if x_i^S and x_j^S are "close" according to k-nearest neighbors ($k \in \mathcal{N}$). Let W_S be a symmetric $m \times m$ matrix, whose element W_S^{ij} is the weight of the edge joining nodes i and j, and is 0 if there is no such edge (W_S is thus sparse). To obtain the embedding space, we solve the generalized eigenvector problem

$$SL_S S^T f = \lambda S D_S S^T f \tag{10}$$

where L_S and D_S are the corresponding Laplacian matrix and diagonal degree matrix of $\mathcal{G}_\mathcal{S}$. Let the column vectors $f_1, \cdots, f_l$ be the solutions of the eigenvectors in (10), ordered according to their eigenvalues, $\lambda_1 < \cdots < \lambda_l$ ($l \leq m$). Thus, the embedding of x_i in the l-dimensional spectral space is represented as

$$y_i = F^T x_i \text{ with } F = [f_1, f_2, \cdots, f_l] \tag{11}$$

Out-of-sample extension can then be treated as a prediction problem in this embedding space. For each $x_j^e (j = m+1, m+2, \cdots, n)$ in B to be extended, we use F to project x_j^e to y_j^e in the learned embedding space. Together with the embedding $\{y_j^S\}_{j=1}^m$ of the m labeled samples o_j^S, we use the k-nearest neighbor classifier to assign the object o_j^e to the class label with the maximum votes from its k nearest neighbors measured in the spectral domain.

4 Sampling Schemes

How to effectively sample a small set of representative samples is critical to encoding the structure of the whole data set. In this work, we focus on the following four sampling schemes in our empirical comparison:

- *Random sampling* (RS) uses a uniform distribution $\{p_i = 1/n\}_{i=1}^{n}$ to choose columns of M. This is a simple and commonly used method, e.g., [3,15,14].
- *Selective sampling* (SS) [18] was shown to be superior to progressive sampling in [16]. This method first selects h distinguished objects using a max-min farthest point strategy. Then each object in $\mathcal{O}$ is grouped to its nearest distinguished object. Finally a small number of samples are randomly selected from each of the h groups to form the sample set.
- *K-means sampling* (KS) suggested in [11] simply chooses the sample points as the K-means cluster centers. This method is inapplicable where the objects cannot be represented by feature vectors. In addition, the computation cost depends greatly on the data size n and the feature dimension.
- *Probabilistic sampling* (PS) [13] uses the probability distribution $\{p_i\}_{i=1}^{n}$ to choose columns of M with $p_i = |M^{(i)}|^2/\|M\|_F^2$, where $|M^{(i)}|$ is the length of the ith column of M, and $\|M\|_F$ is the Frobenius norm of M. This method needs the whole matrix to compute p_i, making it impractical for large n.

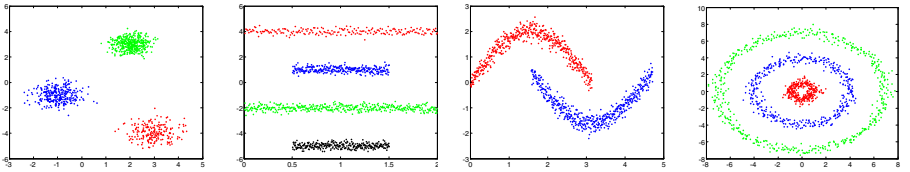
5 Experiments

In order to test these approximate algorithms, we carried out a number of experiments on several artificially generated and real-world data sets. Unless otherwise mentioned, in the following experiments the (Euclidean) distance matrix was computed in the original attribute space, which was then transformed to the affinity matrix by the Gaussian function, i.e., $W_{i,j} = \exp(-\|o_i - o_j\|^2/2\sigma^2)$. The number of clusters c was chosen manually, since choosing c is a difficult model-selection problem which lies outside of the scope of this work. All experiments were implemented in a Matlab 7.2 environment on a PC with an Intel 2.4GHz CPU and 2GB memory running Windows XP.

An accuracy metric AC has been widely used for clustering performance evaluation [4,19]. Suppose that l_i^c is the clustering label of object o_i and l_i^g is the ground truth label, AC is defined as $\max_{map} \sum_{i=1}^{n} \delta(l_i^g, map(l_i^c))/n$, where $\delta(l_1, l_2)$ is the delta function that equals 1 if and only if $l_1 = l_2$ and 0 otherwise, and map is the mapping function that permutes clustering labels to match equivalent ground-truth labels. The Kuhn-Munkres algorithm is usually used to obtain the best mapping [20]. In addition to accuracy, we also measure computational efficiency. For each experiment, we performed these approximate algorithms multiple times, and reported results in terms of the average accuracy (AAC) and the average computation time (ACT). Note that our programs have not been optimized for run-time efficiency.

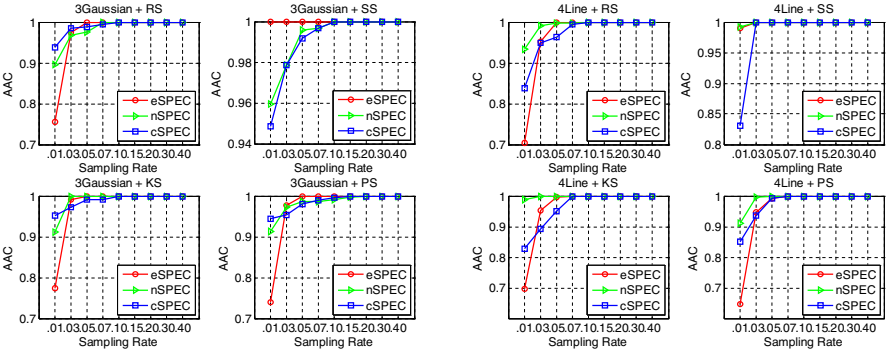
5.1 Results on Synthetic Data Sets

We begin with four synthetic data sets of different types and sizes (i.e., 3Gaussian, 4Line, 2HalfMoon, and 3Circle), whose scatter plots are shown in Figure 1, in which each color represents a cluster. The '3Gaussian' is a simple case in which even central grouping techniques can perform well. The later three cases

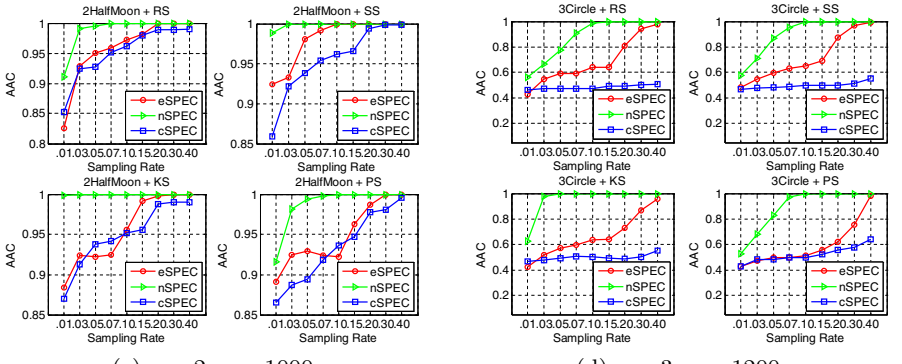


(a) 3Gaussian (b) 4Line (c) 2HalfMoon (d) 3Circle

Fig. 1. Synthetic data with different sizes and structures



(a) $c = 3, n = 900$ (b) $c = 4, n = 1200$

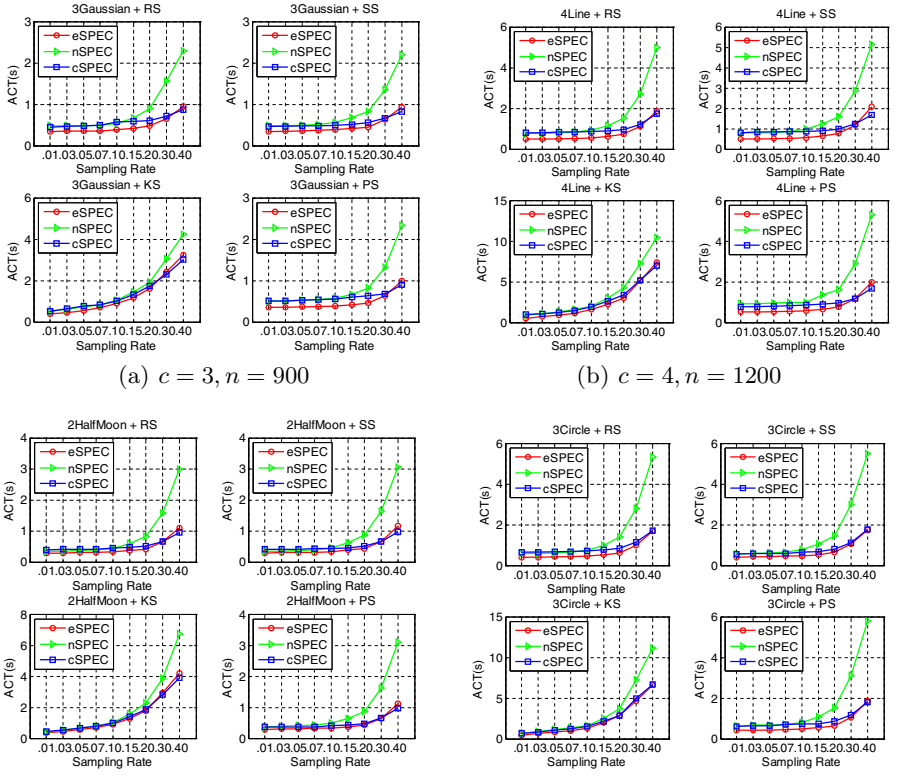


(c) $c = 2, n = 1000$ (d) $c = 3, n = 1200$

Fig. 2. Average accuracy of approximate clustering algorithms

are generally hard for central grouping, but easy for spectral clustering. These synthetic 2D data sets are relatively small so that we can perform both literal clustering and approximate clustering to measure the approximation error. We include these progressively harder data sets to test these approximate algorithms, though these synthetic cases are not necessarily realistic in practice.

First we performed literal spectral clustering on these data sets, and obtained fully correct clustering results at the computational cost of about 11s for



(a) $c = 3, n = 900$ (b) $c = 4, n = 1200$

(c) $c = 2, n = 1000$ (d) $c = 3, n = 1200$

Fig. 3. Average computation time of approximate clustering algorithms

'3Gaussian', 15s for '4Line', 27s for '2HalfMoon' and 27s for '3Circle'. Then, we applied eSPEC, nSPEC and cSPEC. We tried multiple sampling rates (i.e., m/n)[1]. For each sampling rate, 50 trials were made, then we computed the average clustering accuracy and the average computation time consumed by the clustering procedure. The results for the AACs and ACTs are respectively shown in Figures 2 and 3, from which it can be seen that:

– The approximate algorithms can achieve a good approximation, sometimes with the same accuracy as the literal solution when the sample size is sufficient. Moreover, these estimates are obtained using only a small fraction of the data and in much less time than the literal problem.

[1] What we are really concerned with is the sample size m, not m/n, since the ideal m generally depends just on the data structure (i.e., the number of clusters c and their distributions) than the data size n. That is, the necessary sample size m is basically fixed if the data structure is unchanged regardless of the real data size n.

- For complex-shaped clusters (e.g., 2HalfMoon and 3Circle), more samples are generally required to obtain stable results. When the number of samples is sufficient, the accuracy curve remains flat as the sample size increases further, but the computation time increases quickly.
- For simple data sets such as 3Gaussian and 4Line, the three algorithms perform similarly. But for more complex data sets such as 2HalfMoon and 3Circle, nSPEC performs better than the other two.
- In terms of accuracy, nSPEC performs the best overall, then eSPEC, and finally cSPEC. But in terms of computation time, nSPEC is the most expensive, and eSPEC is slightly cheaper than cSPEC.
- In terms of overall accuracy, SS performs best, then KS, and finally RS and PS. The computation time of RS, PS and SS is similar, though PS and SS are a little higher than RS. However, KS spends the most time, even in these small 2D data sets.

These numerical experiments on synthetic data sets suggest that the strategy of approximate spectral clustering can obtain comparable results to the literal approach but in much less time. In addition, overall nSPEC performs best but is most expensive in time. SS achieves the best tradeoff between accuracy and computation efficiency. Considering that synthetic data sets with controllable structures are designed only for simulation and are not realistic, real-world data sets with unknown data distributions would speak louder. Therefore, we will further evaluate these algorithms on several real data sets.

5.2 Results on Real Data Sets

We first considered two medium-sized data sets. 1) The multiple features (MF) data set from the UCI consists of binary image features of handwritten numerals ('0' $\sim$ '9') extracted from a collection of Dutch utility maps. Each class has 200 patterns, thus there are $n = 2000$ patterns in total. These digits are represented as a 649-dimensional vector in terms of 6 feature sets. We set the number of clusters $c = 10$ corresponding to 10 different numerals. 2) The Yale-B face data set[2] contains single light source images of 10 individuals, each seen under 585 viewing conditions (9 poses $\times$ 65 illumination conditions) [21]. Hence, the total number of images is $n = 5850$. Each original image was down-sampled to 30×40 pixels, leading to a 1200-dimensional vector representation. We set the number of clusters $c = 10$ corresponding to 10 different subjects.

For each data set, we applied the algorithms 25 times for each of several sampling rates with the selective sampling scheme. The AACs and ACTs are summarized in Table 1, where *ane* means the average number of examples for each cluster in the sample set. Table 1 shows that 1) the computation time of nSPEC is most expensive, then cSPEC, and finally eSPEC, which is consistent with the results on the synthetic data sets. 2) On the MF data set, nSPEC obtained the best accuracy, then eSPEC, and finally cSPEC, which is basically

[2] http://markus-breitenbach.com/machine_learning_data.php

Table 1. Summary of the results on the MF and Yale-B data sets

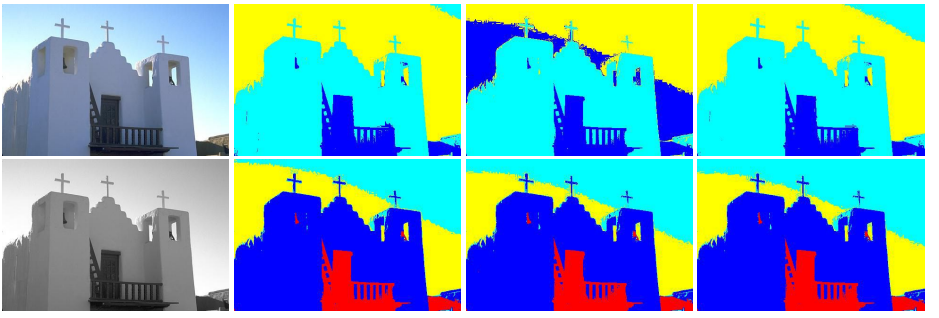
Algorithms			nSPEC		cSPEC		eSPEC	
Data sets	m/n	*ane*	AAC(%)	ACT(s)	AAC(%)	ACT(s)	AAC(%)	ACT(s)
MF	0.03	6	80.10	5.39	73.34	5.09	72.75	1.83
$c = 10$	0.05	10	80.51	5.65	73.76	5.27	75.29	2.08
$n = 2000$	0.07	14	80.57	5.74	76.99	5.33	78.85	2.39
Yale-B	0.01	≈ 6	80.21	13.89	82.03	12.89	72.25	4.05
$c = 10$	0.03	≈ 18	81.97	15.20	84.39	13.09	78.07	6.13
$n = 5850$	0.05	≈ 29	82.11	18.97	85.56	14.46	77.16	8.79

consistent with the results on the synthetic data sets (with similar data sizes). However, it is interesting to see that on the Yale-B data, cSPEC performed better than the other two algorithms. Note that cSPEC performs SVD on a larger submatrix of M than does the Nyström method ($A_{n\times m}$ versus $S_{m\times m}$). This could be a reason why cSPEC performs better than nSPEC in such a relatively large real-world problem.

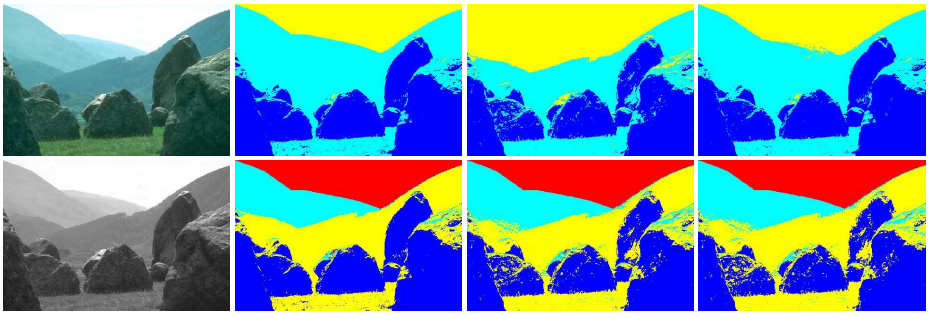
We also applied these clustering algorithms to the problem of high-resolution image segmentation (where it is generally infeasible to use literal spectral clustering). Different features (such as intensity, color, texture, and proximity) can be used to compute the similarities between image pixels, e.g., locally-windowed color and texture histograms were used in [3]. We just used the intensity feature since our main concern is to demonstrate the feasibility of these approximate algorithms in the context of image segmentation, but not purely for image segmentation. Figure 4 shows segmentation results on three 481×321 images[3], in which pixels with the same color represent one group. We set $c = 3$ (or 4) for these images according to the number of visually meaningful components.

Running a literal spectral clustering algorithm on the whole image (which contains $n = 481 \times 321 = 154,401$ pixels) would be simply impossible in the Matlab environment. For these images, the number of sampled pixels was empirically chosen to be 150 (less than 0.1% of the number of total pixels), considering that there are far fewer coherent groups (*i.e.,* $c \ll n$) in a scene than pixels. We cannot measure the clustering error in this case because literal spectral clustering cannot be performed and we lack any form of ground truth. So, the best we can do for evaluation here is to resort to visual inspection of the segmentation results. In these three cases, all algorithms partitioned the images into meaningful components when $c = 4$ regardless of slight differences. More interesting, when $c = 3$, nSPEC gave results that were inconsistent with human perception of the intensity values in the images (i.e., a tendency to over-segmentation), whereas cSPEC and eSPEC performed similarly well. This seems to demonstrate again that cSPEC could be superior to nSPEC on these larger image data sets.

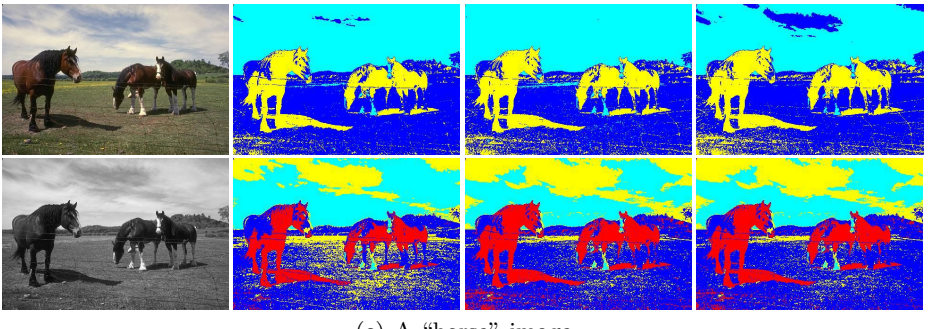
[3] http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/



(a) A “house” image



(b) A “stone” image



(c) A “horse” image

Fig. 4. Intensity-based image segmentation results. For each of (a), (b) and (c), the left-most column is original color image (top) and corresponding intensity image (bottom); and the right three columns are respectively the segmentation results using eSPEC, nSPEC and cSPEC with $c = 3$ (top row) and $c = 4$ (bottom row).

6 Discussion and Conclusion

This paper has examined several approximate spectral clustering approaches. Extensive experiments on synthetic and real-world data sets show that these algorithms are not only feasible for very large data sets, but also provide acceleration on large data sets with comparable accuracy compared to the literal

solution. In particular, in terms of memory, the matrix (i.e., A, or S and B) needed in these three approximate algorithms can be simply computed on demand. This greatly reduces the memory requirements for very large-scale problems.

Accuracy and efficiency are two important factors in data clustering. Comparative results on the synthetic data sets have shown that nSPEC performs best in terms of accuracy, but this is not always the case in real-world data sets. We cannot thus say that a specific algorithm is always superior to the others. The computation time of nSPEC is consistently highest among all of the three algorithms, which may be due to its additional strategy for computing approximated orthogonalized eigenvectors. Relatively, eSPEC is cheapest, which makes out-of-sample extension more appealing when a large number of samples have been accumulated. To summarize, just as discussed in [1], there is no clustering algorithm that can be universally used to solve all problems. It is more likely that the performance of each clustering method depends strongly on the real characteristics of the data sets used. In this sense, it is not rational to claim a "best" in the context of clustering algorithms, though comparison on a wider variety of data is possible. However, among the four compared sampling schemes, selective sampling provides the best choice in terms of accuracy, efficiency and applicability for various types of data.

References

1. Xu, R., Wunsch II, D.: Survey of clustering algorithms. IEEE Trans. Neural Networks 16(3), 645–678 (2005)
2. Luxburg, U.: A tutorial on spectral clustering. Technical report, Max Planck Institute for Biological Cybernetics, Germany (2006)
3. Fowlkes, C., Belongie, S., Chung, F., Malik, J.: Spectral grouping using the Nyström method. IEEE Trans. Pattern Analysis and Machine Intelligence 26(2), 214–225 (2004)
4. Ning, H., Xu, W., Chi, Y., Gong, Y., Huang, T.: Incremental spectral clustering with application to monitoring of evolving blog communities. In: SIAM Conference on Data Mining (2007)
5. Miao, G., Song, Y., Zhang, D., Bai, H.: Parallel spectral clustering algorithm for large-scale community data mining. In: International Conference on WWW (2008)
6. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
7. Ding, C., He, X., Zha, H., Gu, M., Simon, H.: A min-max cut algorithm for graph partitioning and data clustering. In: International Conference on Data Mining, pp. 107–114 (2001)
8. Chung, F.: Spectral Graph Theory. American Mathematical Society (1997)
9. Ng, A., Jordan, M., Weiss, Y.: On spectral clustering: analysis and an algorithm. In: Advances in Neural Information Processing Systems (2001)
10. Williams, C., Seeger, M.: Using the Nyström method to speed up kernel machines. In: Advances in Neural Information Processing Systems, pp. 682–688 (2000)
11. Zhang, K., Tsang, I.W., Kwok, J.T.: Improved Nyström low-rank approximation and error analysis. In: International Conference on Machine Learning (2008)

12. Deshpande, A., Rademacher, L., Vempala, S., Wang, G.: Matrix approximation and projective clustering via volume sampling. In: Symposium on Discrete Algorithms (2006)
13. Drineas, P., Kannan, R., Mahoney, M.: Fast Monte Carlo algorithms for matrices II: computing a low-rank approximation to a matrix. SIAM Journal on Computing 36(1), 158–183 (2006)
14. Talwalkar, A., Kumar, S., Rowley, H.: Large-scale manifold learning. In: International Conference on Computer Vision and Pattern Recognition (2008)
15. Pavan, M., Pelillo, M.: Efficient out-of-sample extension of dominant-set clusters. In: Advances in Neural Information Processing Systems (2004)
16. Bezdek, J., Hathaway, R., Huband, J., Leckie, C., Kotagiri, R.: Approximate clustering in very large relational data. International Journal of Intelligent Systems 21(8), 817–841 (2006)
17. He, X., Niyogi, P.: Locality preserving projections. In: Advances in Neural Information Processing Systems (2003)
18. Wang, L., Bezdek, J.C., Leckie, C., Kotagiri, R.: Selective sampling for approximate clustering of very large data sets. International Journal of Intelligence Systems 23(3), 313–331 (2008)
19. Cai, D., He, X., Han, J.: Document clustering using locality preserving indexing. IEEE Trans. Knowledge and Data Engineering 17(2), 1637–1642 (2005)
20. Lovasz, L., Plummer, M.: Matching Theory. Akademiai Kiado. North Holland, Budapest (1986)
21. Georghiades, A., Belhumeur, P., Kriegman, D.: From few to many: illumination cone models for face recognition under variable lighting and pose. IEEE Trans. Pattern Analysis and Machine Intelligence 23(6), 643–660 (2001)

An Integration of Fuzzy Association Rules and WordNet for Document Clustering

Chun-Ling Chen[1], Frank S.C. Tseng[2], and Tyne Liang[1]

[1] Dept. of Computer Science, National Chiao Tung University, Taiwan, ROC
{chunling,tliang}@cs.nctu.edu.tw
[2] Dept. of Information Management National Kaohsiung 1st Univ. of Sci. & Tech., Taiwan, ROC
imfrank@ccms.nkfust.edu.tw

Abstract. With the rapid growth of text documents, document clustering has become one of the main techniques for organizing large amount of documents into a small number of meaningful clusters. However, there still exist several challenges for document clustering, such as high dimensionality, scalability, accuracy, meaningful cluster labels, and extracting semantics from texts. In order to improve the quality of document clustering results, we propose an effective Fuzzy Frequent Itemset-based Document Clustering (F^2IDC) approach that combines fuzzy association rule mining with the background knowledge embedded in WordNet. A term hierarchy generated from WordNet is applied to discovery fuzzy frequent itemsets as candidate cluster labels for grouping documents. We have conducted experiments to evaluate our approach on Reuters-21578 dataset. The experimental result shows that our proposed method outperforms the accuracy quality of FIHC, HFTC, and UPGMA.

Keywords: Fuzzy association rule mining, Text mining, Document clustering, Frequent itemsets, WordNet.

1 Introduction

With the rapid growth of text documents, document clustering has become one of the main techniques for managing large document collections [2]. Several effective document clustering algorithms have been proposed including the *k*-means, Bisecting *k*-means Algorithm, Hierarchical Agglomerative Clustering (HAC), Unweighted Pair Group Method with Arithmatic Mean (UPGMA), etc. However, there still exist some challenges for the clustering quality [1][4][6], such as (1) have high-dimensional term features, (2) are not scalable for large document sets (like UPGMA), (3) require the user to specify the number of clusters as an input parameter, which is usually unknown in advance, (4) do not provide a meaningful label (or description) for cluster, and (5) do not embody any external knowledge to extract semantics from texts.

To resolve the problems of high dimensionality, large size, and understandable cluster description, Beil *et al.* [1] developed the first frequent itemsets-based algorithm, namely Hierarchical Frequent Term-based Clustering (HFTC), where the

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 147–159, 2009.
© Springer-Verlag Berlin Heidelberg 2009

frequent itemsets are generated based on the association rule mining, e.g., Apriori [3]. They consider the only low-dimensional frequent itemsets as clusters. However, the experiments of Fung *et al.* [4] showed that HFTC is not scalable. For a scalable algorithm, Fung *et al.* proposed a novel approach, namely Frequent Itemset-based Hierarchical Clustering (FIHC), by using derived frequent itemsets to construct a hierarchical topic tree for clusters. They also proved that using frequent itemsets for document clustering can reduce the dimension of a vector space effectively. Yu *et al.* [14] presented another frequent itemset-based algorithm, called TDC, to improve the clustering quality and scalability. This algorithm dynamically generates a topic directory from a document set using only closed frequent itemsets and further reduces dimensionality.

Recently, WordNet [10], which is one of the most widely used thesauruses for English, have been used to group documents with its semantic relations of terms [6] [8]. Many existing document clustering algorithms mainly transform text documents into simplistic flat bags of document representation, e.g., term vectors or bag-of-words. Once terms are treated as individual items in such simplistic representation, their semantic relations will be lost. Thus, Dave *et al.* [8] employed synsets as features for document representation and subsequent clustering. However, synsets would decrease the clustering performance in all experiments without considering word sense disambiguation. Accordingly, Hotho *et al.* [6] used WordNet in document clustering for word sense disambiguation to improve the clustering performance.

In order to flexibly conduct the association rule mining for more applications, some research works [5][9] have been proposed to integrate fuzzy set theory and association rule mining to find interesting fuzzy association rules from given transactions. Moreover, Kaya *et al.* [7] think that fuzzy association rule mining is understandable to humans because it integrates linguistic terms with fuzzy sets. Thus, we propose an effective Fuzzy Frequent Itemset-based Document Clustering (F^2IDC) approach based on fuzzy association rule mining in conjunction with WordNet for clustering textual documents. We focus on the use of fuzzy association rule mining devised by Hong *et al.* [5] to find fuzzy frequent itemsets and further obtain appropriate topic labels for the derived clusters. Moreover, we show how to take into account hypernyms of WordNet for capturing conceptual similarity of terms, and add these hypernyms as term features for the document representation.

In summary, our approach has the following advantages:

1. It presents a means of dynamically deriving a hierarchical organization of concepts from the WordNet thesaurus based on the content of each document without use of training data or standard clustering techniques;
2. Following Hong *et al.* [5], it extends a fuzzy data representation to text mining for discovering the generalized fuzzy frequent itemsets as the candidate clusters;
3. By conducting experimental evaluation on Reuters-21578 dataset, it has been proven that our approach outperforms the accuracy quality of FIHC, HFTC, and UPGMA.

The subsequent sections of this paper are organized as follows. In section 2, the basic notations used in this paper are defined. In section 3, a detailed description of our approach with an example is presented. The experimental evaluation is described and the results are shown in section 4. Finally, we conclude in section 5.

2 Basic Notations

In this section, we define the basic notations used in the rest of the paper. Given a *document set D*, each document $d_i \in D$, denoted $d_i = \{(t_1, f_{i1}), (t_2, f_{i2}),\ldots, (t_j, f_{ij}),\ldots, (t_m, f_{im})\}$, is represented by a set of pair (term, frequency) where the frequency f_{ij} represents the occurrence of the key term t_j in d_i.

From a set of documents $D = \{d_1, d_2,\ldots, d_i,\ldots, d_n\}$, we can obtain a *term set* $T_D = \{t_1, t_2,\ldots, t_j,\ldots, t_s\}$. The *key term set* of D, denoted $K_D = \{t_1, t_2,\ldots, t_j,\ldots, t_m\}$, is a subset of T_D, including only meaningful key terms, which are not appeared in a well-defined stop word list, and all satisfy the pre-defined minimum *tf-idf* threshold α. The symbols n and m are denoted as the total number of documents and the total number of key terms, respectively.

After the mining process, a *candidate cluster set* of D, denoted $\tilde{C}_D = \{\tilde{c}_1^1,\ldots, \tilde{c}_{l-1}^2, \tilde{c}_l^q,\ldots, \tilde{c}_k^q\}$, is generated, where k is the total number of candidate clusters. A *candidate cluster* $\tilde{c} = (\tilde{D}_c, \tau)$ is a two-tuple, where $\tilde{D}_c$ is a subset of the document set D, such that it includes those documents which contain all the key terms in $\tau = \{t_1, t_2,\ldots, t_q\} \subseteq K_D$, $q \geq 1$, where K_D is the key term set of D and q is the number of key terms contained in τ. In fact, τ is a fuzzy frequent itemset for describing $\tilde{c}$. To illustrate, $\tilde{c}$ can also be denoted as $\tilde{c}^q_{(t_1, t_2,\ldots, t_q)}$ or $\tilde{c}^q_{(\tau)}$, and will be used interchangeably hereafter. For instance, the *candidate cluster* $\tilde{c}^1_{(\text{trade})} = (\{d_2, d_3\}, \{\text{trade}\})$, as the term "trade" appeared in these documents.

For assigning each document to a fitting cluster, each candidate cluster $\tilde{c}^q_{(\tau)}$ with fuzzy frequent itemset τ are considered in the clustering process. τ will be regarded as a reference point for generating a target cluster. In order to represent the degree of importance of a document d_i in a candidate cluster $\tilde{c}_l^q$, an $n \times k$ Document-Cluster Matrix (DCM) will be constructed to calculate the similarity of terms in d_i and $\tilde{c}_l^q$. A formal illustration of DCM can be found in Fig. 1.

Based on DCM, c_i^q may or may not be assigned a subset of documents. For the documents in each c_i^q, the intra-cluster similarity is minimized and the inter-clusters similarity is maximized. We call each c_i^q a *target cluster* in the following. Finally, the target cluster set, denoted $C_D = \{c_1^1, c_2^1,\ldots, c_i^q,\ldots, c_f^q\}$, can be obtained for D.

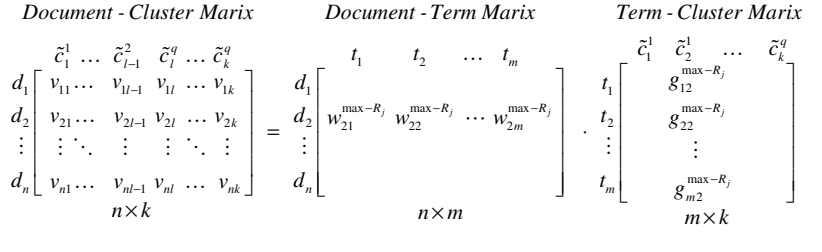


Fig. 1. Document-Cluster Matrix

For merging similar target clusters, we measure the similarity between two target clusters c_x^q and c_y^q, $c_x^q \neq c_y^q$. The inter-cluster similarity measurement between two target clusters is defined as follows.

$$Inter_Sim(c_x^q, c_y^q) = \frac{\sum_{i=1,\, d_i \in c_x^q, c_y^q}^{n} v_{ix} \times v_{iy}}{\sqrt{\sum_{i=1,\, d_i \in c_x^q}^{n} (v_{ix})^2 \times \sum_{i=1,\, d_i \in c_y^q}^{n} (v_{iy})^2}} \quad (1)$$

where v_{ix} and v_{iy} stand for two entries, such that $d_i \in c_x^q$ and $d_i \in c_y^q$, in DCM, respectively. If the *Inter-Sim* value is higher, then both clusters are regarded nearly the same. In the following, the minimum Inter-Sim will be used as a threshold δ to decide whether two target clusters should be merged.

3 Fuzzy Frequent Itemset-Based Document Clustering (F^{2}IDC) Algorithm

There are three stages in our framework as shown in Fig. 2. We explain them as follows:

(1) *Document Analyzing.* After the steps of document preprocessing and enrichment, the designated document representation is prepared for the later mining algorithm.
(2) *Fuzzy Frequent Itemsets mining.* Starting from the document representation of all documents, we run fuzzy association rule mining algorithm to discovery fuzzy frequent itemsets and then generate the candidate clusters.
(3) *Document Clustering.* To represent the degree of importance of a document d_i in a candidate cluster $\tilde{c}_l^q$, an $n \times k$ DCM will be constructed to calculate the similarity of terms in d_i and $\tilde{c}_l^q$. Based on the obtained DCM, each document will be assigned into a target cluster.

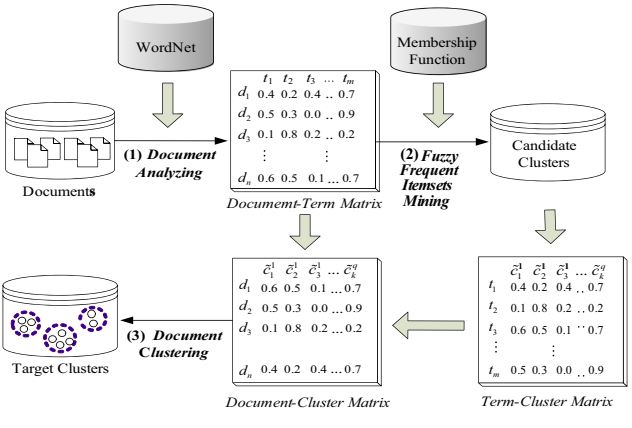


Fig. 2. The F^{2}IDC framework

3.1 Stage 1: Document Analyzing

The feature description of a document is constituted by terms of the document set to form a term vector. The immediate drawback of a term vector for document clustering is the size of the vector. A term vector with high dimensions is easy to make clustering inefficient and difficult in principle. Hence, in this paper, we employ tf-idf threshold method to produce a low dimensional term vector.

Document Pre-processing. We describe the details of the pre-processing in the following:

(1) *Extract terms.* Divide the sentences into terms and extract nouns as term features.
(2) *Remove the stop words.* We use a stop word list that contains words to be excluded. The list is applied to remove commonly used words that do not discriminate for topics.
(3) *Conduct word stemming*. Use the developed stemming algorithms, such as Porter [11], to convert a word to its stem or root form.
(4) *Select key terms*. We apply tf-idf threshold method to choose key terms of the document set.

Definition 1. tf-idf (term frequent × inverse document frequency): It is denoted as $tfidf_{ij}$ and used for the measure of the importance of a term t_j within a document d_i. For preventing a bias for longer documents, the weighted frequency of each term is usually normalized by the maximum frequency of all terms in d_i, and is defined as follows:

$$tfidf_{ij} = 0.5 + 0.5 * \frac{f_{ij}}{\max_{t_j \in d_i}(f_{ij})} \times log\left(1 + \frac{|D|}{|\{d_i | t_j \in d_i, d_i \in D\}|}\right) \quad (2)$$

where f_{ij} is the frequency of t_j in d_i, and the denominator is the maximum frequency of all terms in d_i. $|D|$ is the total number of documents in the document set D, and $|\{d_i \mid t_j \in d_i, d_i \in D\}|$ is the number of documents containing t_j.

Document Enrichment. After the step of document preprocessing, we additionally enrich the document representation by using WordNet, a source repository of semantic meanings. WordNet, developed by Miller *et al.* [10], consists of so-called synsets, together with a hypernym/hyponym hierarchy.

The basic idea of document enrichment is to add the generality of terms by corresponding hypernyms of WordNet based on the key terms appeared in each document. Each key term are linked up to five levels of hypernyms. For a simple and effective combination, these added hypernyms form a new key term set, denoted $K_D = \{t_1, t_2, \ldots, t_m, h_1, \ldots, h_r\}$, where h_j is a hypernym. The enriched document d_i is represented by $d_i = \{(t_1, f_{i1}), (t_2, f_{i2}), \ldots, (t_m, f_{im}), (h_1, hf_{i1}), \ldots, (h_r, hf_{ir})\}$, where a weight of 0 will be assigned to several terms appearing in some of the documents but not in d_i. The frequency f_{ij} of a key term t_j in d_i is mapped to its hypernyms $\{h_1, \ldots, h_j \ldots, h_r\}$ to accumulate as the frequency hf_{ij} of h_j.

The reason of using hypernyms of WordNet is that hypernyms can reveal hidden similarities to identify related topic, potential leading to the better clustering quality [12]. For example, a document about 'sale' may not be associated to a document

about 'trade' by the clustering algorithm if there are only 'sale' and 'trade' in the key term set. But, if the more general term 'commerce' is added to both documents, their semantic relation is revealed. The suitable representation of each document for the later mining can be derived by Algorithm 1.

Algorithm 1. Basic algorithm to obtain the designated representation of all documents

Input: A document set D; A well-defined stop word list; WordNet; The minimum tf-idf threshold $\alpha = 0.6$.

Output: The formal representation of all documents in D.

1. Extract the term set $T_D = \{t_1, t_2, \ldots, t_j, \ldots, t_s\}$
2. Remove all stop words from T_D
3. Apply Stemming for T_D
4. For each $d_i \in D$ do //key term selection
   For each $t_j \in T_D$ do
   (1) Evaluate its $tfidf_{ij}$ weight // defined by Formula (2) in Section 2
   (2) Retain the term if $tfidf_{ij} \geq \alpha$
5. Form the key term set $K_D = \{t_1, t_2, \ldots, t_j, \ldots, t_p\}$
6. For each $d_i \in D$ do //document enrichment step
   For each $t_j \in K_D$ do
   (1) If (h_j is hypernyms of t_j) then //refer to WordNet
   (a) $hf_{ij} \rightarrow hf_{ij} + f_{ij}$
   (b) $K_D \rightarrow K_D \cup \{h_j\}$
7. For each $d_i \in D$ do //in order to decrease noise from hypernyms, tf-idf method is executed again
   For each $t_j \in K_D$ do
   (1) Evaluate its $tfidf_{ij}$ weight
   (2) Retain the term if $tfidf_{ij} \geq \alpha$
8. Form the new key term set $K_D = \{t_1, t_2, \ldots, t_p, h_1, \ldots, h_r\}$
9. For each $d_i \in D$, record the frequency f_{ij} of t_j and the frequency hf_{ij} of h_j in d_i to obtain the final representation of $d_i = \{(t_1, f_{i1}), (t_2, f_{i2}), \ldots, (t_p, f_{ip}), (h_1, hf_{i1}), \ldots, (h_r, hf_{ir})\}$

3.2 Stage 2: Fuzzy Frequent Itemsets Mining

In the mining process, it is considered that documents and key terms are transactions and purchased items, respectively. In the following, we define the membership functions and present our fuzzy association rule mining algorithm for texts.

The membership functions. Each pair (t_j, f_{ij}) of a document d_i can be transformed into a fuzzy set $F_{ij} = w_{ij}^{Low}/t_j.Low + w_{ij}^{Mid}/t_j.Mid + w_{ij}^{High}/t_j.High$ with its frequency being represented by three fuzzy regions, namely *Low*, *Mid*, and *High*, to depict its grade of membership within d_i. Each fuzzy value w_{ij}^r has a corresponding membership function, denoted $w_{ij}^r(f_{ij})$, to convert the key tem frequency f_{ij} into a value of the range [0, 2], where r can be *Low*, *Mid*, and *High*, and the corresponding membership functions $w_{ij}^r(f_{ij})$ are defined by Formulas (3), (4), and (5), respectively. The derived membership functions are shown in Fig. 3.

In Formulas (3), (4), and (5), $min(f_{ij})$ is the minimum frequency of terms in D, $max(f_{ij})$ is the maximum frequency of terms in D, and $avg(f_{ij}) = \lceil \frac{\sum_{i=1}^{n} f_{ij}}{|K|} \rceil$, where $f_{ij} \neq min(f_{ij})$ or $max(f_{ij})$, and $|K|$ is the number of summed key terms.

$$w_{ij}^{Low}(f_{ij}) = \begin{cases} 0, & f_{ij} = 0 \\ 1 + \frac{f_{ij} - a}{b - a}, & a \le f_{ij} \le b \\ 2, & b < f_{ij} < c \\ 1 + \frac{f_{ij} - d}{c - d}, & c \le f_{ij} \le d \\ 1, & f_{ij} > d \end{cases} \quad \begin{array}{l} a = 0, \\ b = min(f_{ij}), \\ c = \frac{\lceil avg(f_{ij}) - min(f_{ij}) \rceil}{2} - min(f_{ij}), \\ d = avg(f_{ij}) \end{array} \quad (3)$$

$$w_{ij}^{mid}(f_{ij}) = \begin{cases} 0, & f_{ij} = 0 \\ 1, & f_{ij} < a \\ 1 + \frac{f_{ij} - a}{b - a}, & a \le f_{ij} \le b \\ 2, & b < f_{ij} < c \\ 1 + \frac{f_{ij} - d}{c - d}, & c \le f_{ij} \le d \\ 1, & f_{ij} > d \end{cases} \quad \begin{array}{l} a = min(f_{ij}), \\ b = \frac{\lceil avg(f_{ij}) - min(f_{ij}) \rceil}{2}, \\ c = avg(f_{ij}), \\ d = avg(f_{ij}) + \frac{\lceil max(f_{ij}) - avg(f_{ij}) \rceil}{4} \end{array} \quad (4)$$

$$w_{ij}^{high}(f_{ij}) = \begin{cases} 0, & f_{ij} = 0 \\ 1, & f_{ij} < a \\ 1 + \frac{f_{ij} - a}{b - a}, & a \le f_{ij} \le b \\ 2, & b < f_{ij} < c \\ 1 + \frac{f_{ij} - d}{c - d}, & c \le f_{ij} \le d \end{cases} \quad \begin{array}{l} a = avg(f_{ij}), \\ b = avg(f_{ij}) + \frac{\lceil max(f_{ij}) - avg(f_{ij}) \rceil}{4}, \\ c = avg(f_{ij}) + \frac{\lceil max(f_{ij}) - avg(f_{ij}) \rceil}{2}, \\ d = max(f_{ij}) \end{array} \quad (5)$$

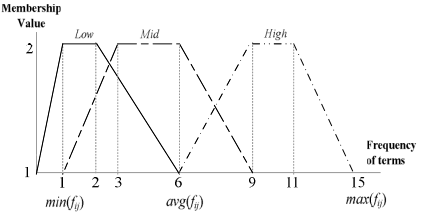


Fig. 3. The predefined membership functions

The fuzzy association rule mining algorithm for texts. Algorithm 2 generates fuzzy frequent itemsets based on pre-defined membership functions and the minimum support value θ, from a large textual document set, and obtains a candidate cluster set according to the minimum confidence value λ. Since each discovered fuzzy frequent itemset has an associated fuzzy count value, it can be regarded as the degree of importance that the itemset contributes to the document set.

Algorithm 2. Basic algorithm to obtain the fuzzy frequent itemsets from the document set.

Input: A set of documents $D = \{d_1, d_2, \ldots, d_n\}$, where $d_i = \{(t_1, f_{i1}), (t_2, f_{i2}), \ldots, (t_j, f_{ij}), \ldots, (t_m, f_{im})\}$; A set of membership functions (as defined in Section 3.2.1); The minimum support value θ; The minimum confidence value λ.

Output: A set of candidate cluster $\tilde{C}_D$.

1. For each $d_i \in D$ do
   For each $t_j \in d_i$ do
   (1) $f_{ij} \rightarrow F_{ij} = w_{ij}^{Low} / t_j.Low + w_{ij}^{Mid} / t_j.Mid + w_{ij}^{High} / t_j.High$ //using membership functions
2. For each $t_j \in K_D$ do
   For each $d_i \in D$ do
   (1) $count_j^{Low} = \sum_1^n w_{ij}^{Low}$, $count_j^{Mid} = \sum_1^n w_{ij}^{Mid}$, $count_j^{High} = \sum_1^n w_{ij}^{High}$
3. For each $t_j \in K_D$ do
   (1) $max\text{-}count_j = max(count_j^{Low}, count_j^{Mid}, count_j^{High})$
4. $L_1 = \{max\text{-}R_j | support(t_j) = \frac{max\text{-}count_j}{|D|} \ge \theta, 1 \le j \le m\}$ // $|D|$ is the number of documents.

5. For (q = 2; $L_{q-1} \neq \varnothing$; q++) do // Find fuzzy frequent q -itemsets L_q
   (1) C_q = **apriori_gen**(L_{q-1}, θ) // similar to the *a priori* algorithm [3]
   (2) For each candidate q -itemsets τ with key terms (t_1, t_2,…, t_q) $\in C_q$ do
      (a) For each $d_i \in D$ do
         $w_{i\tau} = min\left\{ w_{ij}^{max\text{-}R_j} \mid j = 1, 2, \ldots, q \right\}$ // $w_{ij}^{max\text{-}R_j}$ is the fuzzy membership value of the maximum region of t_j in d_i
      (b) $count_\tau = \sum_{i=1}^{n} w_{i\tau}$
   (3) $L_q = \{ \tau \in C_{q+1} \mid support(\tau) = \frac{count_\tau}{|D|} \geq \theta, 1 \leq j \leq q \}$

6. For all the fuzzy frequent q-itemsets τ containing key terms (t_1, t_2,…,t_q), where $q \geq 2$ do //construct the strong fuzzy frequent itemsets
   (1) form all possible association rules $\tau_1 \wedge \ldots \wedge \tau_{k-1} \wedge \tau_{k+1} \wedge \ldots \wedge \tau_q \rightarrow \tau_k$, k = 1 to q
   (2) Calculate the confidence values of all possible association rules

$$confidence(\tau) = \frac{\sum_{i=1}^{n} w_{i\tau}}{\sum_{i=1}^{n} (w_{i1} \wedge \ldots \wedge w_{ik\text{-}1}, w_{ik+1} \wedge \ldots \wedge w_{iq})}$$

   (3) $\tilde{C}_D = \{ \tau \in L_q \mid confidence(\tau) \geq \lambda \}$

7. $\tilde{C}_D \rightarrow \{L_1\} \cup \tilde{C}_D$

Procedure **apriori_gen**(L_{q-1}, θ)

1. For each itemset $l_1 \in L_{q-1}$ do
   For each itemset $l_2 \in L_{q-1}$ do
      (1) if ($l_1[1] = l_2[1] \wedge l_1[2] = l_2[2] \wedge \ldots \wedge l_1[k\text{-}2] = l_2[k\text{-}2] \wedge l_1[k\text{-}1] = l_2[k\text{-}1]$) then $C_q = \{ c \mid c = l_1 \times l_2 \}$
2. Return C_q

3.3 Stage 3: Document Clustering

The objective of Algorithm 3 is to assign each document to the best fitting cluster c_i^q, and finally obtain the target cluster set for output. The assignment process is based on the Document-Cluster matrix (DCM) derived from the Document-Term Matrix (DTM) and the Term-Cluster (TCM). For avoiding low the clustering accuracy, the inter-cluster similarity (defined by Formula (1) in Section 2) between two target clusters is calculated to merge the small target clusters with the similar topic.

Algorithm 3. Basic algorithm to obtain the target clusters

Input: A document set $D = \{d_1, d_2, \ldots, d_i, \ldots, d_n\}$; The key term set $K_D = \{t_1, t_2, \ldots, t_j, \ldots, t_m\}$; The candidate cluster set $\tilde{C}_D = \{ \tilde{c}_1^1, \ldots, \tilde{c}_{l-1}^1, \tilde{c}_l^q, \ldots, \tilde{c}_k^q \}$; A minimum *Inter-Sim* threshold δ;

Output: The target cluster set $C_D = \{ c_1^1, c_2^1, \ldots, c_i^q, \ldots, c_f^q \}$

1. Build $n \times m$ document-term matrix $W = \left[w_{ij}^{\max - R_j} \right]$ // $w_{ij}^{\max - R_j}$ is the weight (fuzzy value) of t_j in d_i and $t_j \in L_1$

2. Build $m \times k$ term-cluster matrix $G = \left[g_{jl}^{\max-R_j} \right]$ // $g_{jl}^{\max-R_j} = \dfrac{score(\tilde{c}_l^q)}{\sum_{i=1}^{n} w_{ij}^{\max-R_j}}$, $1 \le j \le m$, $1 \le l \le k$, and

$score(\tilde{c}_l^q) = \sum_{d_i \in \tilde{c}_l^q,\, t_j \in \tau} w_{ij}^{\max-R_j}$, where $w_{ij}^{\max-R_j}$ is the weight (fuzzy value) of t_j in $d_i \in \tilde{c}_l^q$ and $t_j \in L_1$

3. Build $n \times k$ document-cluster matrix $V = W \cdot G = [v_{il}] = \sum_{m=1}^{m} w_{im} g_{ml}$

4. Based on V, assign d_i to a target cluster c_l^q

(1) $c_l^q = \{ d_i \mid v_{il} = max\{v_{i1}, v_{i2}, ..., v_{il}\} \in \tilde{c}_l^q$, where the number of v_{il} is $1 \}$, otherwise (2)

(2) $c_l^q = \{d_i \mid v_{il} = max\{v_{i1}, v_{i2}, \ldots, v_{il}\} \in \tilde{c}_l^q$, where the number of $v_{il} > 1$ and $\tilde{c}_l^q$ with the highest fuzzy count value corresponding to its fuzzy frequent itemset}

5. Clusters merging

(1) For each $c_l^q \in C_D$ do

(a) If (c_l^q = null) then { remove this target clusters c_l^q from C_D }

(2) For each pair of target clusters (c_x^q, c_y^q) $\in C_D$ do

(a) Calculate the *Inter_sim*

(b) Store the results in the Inter-Cluster Similarity matrix *I*

(3) If (one of the *Inter_sim* value in $I \geq \delta$) then

(a) Select (c_x^q, c_y^q) with the highest *Inter_sim*

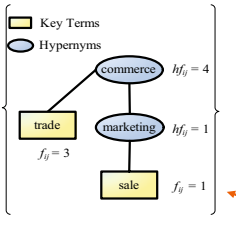
(b) Merge the smaller target cluster into the larger target cluster

(c) Repeat Step (2) to update *I*

6. Output C_D

3.4 An Example

Suppose we have a document set $D = \{d_1, d_2, \ldots, d_5\}$ and its key term set $K_D = \{$sale, trade, medical, health$\}$. Fig. 4 illustrates the process of Algorithm 1 to obtain the representation of all documents. Notice that we use a tabular representation, where each entry denotes the frequency of a key term (the column heading) in a document d_i (the row heading), to make our presentation more concise. This representation scheme will be employed in the following to illustrate our approach.



Docs	Key Term Set			
	Sale	trade	medical	health
d_1	1	0	0	0
d_2	0	3	0	0
d_3	1	3	0	0
d_4	0	0	8	1
d_5	0	0	11	2

Document Enrichment

Docs	Key Term Set				Hypernyms	
	sale	trade	medical	health	marketing	commerce
d_1	1	0	0	0	1	1
d_2	0	3	0	0	0	3
d_3	1	3	0	0	1	4
d_4	0	0	8	1	0	0
d_5	0	0	11	2	0	0

Fig. 4. The process of Algorithm 1 of this example

Consider the representation of all documents generated by Algorithm 1 in Fig. 4, the membership functions defined in Fig. 3, the minimum support value 70%, and the minimum confidence value 70% as inputs. The fuzzy frequent itemsets discovery procedure is depicted in Fig. 5.

Moreover, consider the candidate cluster set $\tilde{c}_D$ was already generated in Fig 5. Now, suppose the minimum *Inter-Sim* value is 0.5. Fig. 6 illustrates the process of Algorithm 3 and shows the final results.

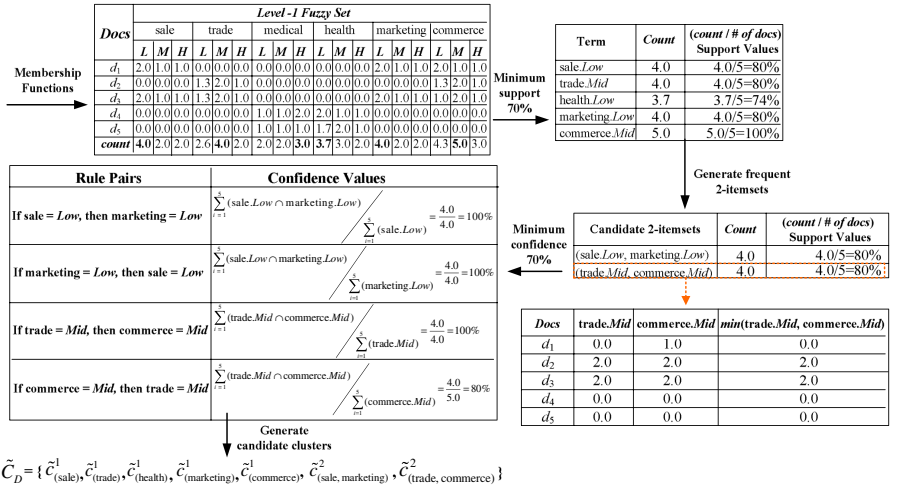


Fig. 5. The process of Algorithm 2 of this example

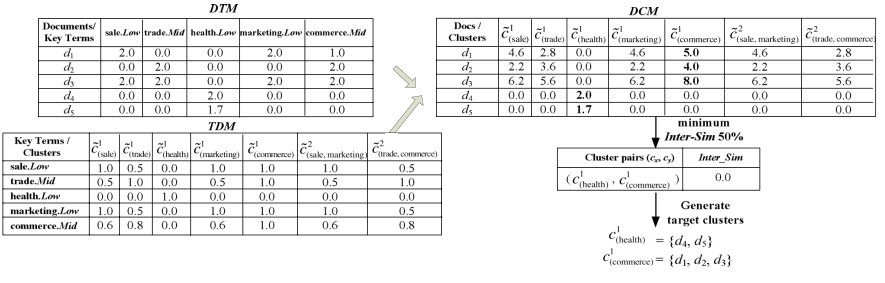


Fig. 6. The process of Algorithm 3 of this example

4 Experimental Evaluation

We have experimentally evaluated the performance of the proposed algorithm by comparing with that of the FIHC method. We make use of the FIHC 1.0 tool[1] to

[1] http://ddm.cs.sfu.ca/dmsoft/Clustering/products/

generate the results of FIHC. The produced results are then fetched into the same evaluation program to ensure a fair comparison. Moreover, Steinbach *et al.* [13] have compared the performance of some influential clustering algorithms, and the results indicated that UPGMA and Bisecting *k*-means are the most accurate clustering algorithms. Therefore, we compare the performance of our algorithm with that of FIHC, HFTC, and UPGMA algorithms in term of the clustering accuracy.

4.1 Data Sets

We used two standard datasets in our experiments, and the detailed information of these datasets is described as follow:

1. *Re0*: *Re0* is a text document dataset, derived from Reuters-21578 [2] text categorization test collection Distribution 1.0. *Re0* includes 1,504 documents belonging to 13 different classes.
2. *Reuters*: The documents in Reuters-21578 are divided into 135 topics mostly concerning business and economy. In our test, we discarded documents with multiple category labels, and the result is consisting of documents associated with a single topic of 8,293 documents and 65 classes.

There are several reasons presented in [6][12] to choose Reuters-21578 dataset in our experiments.

1. Its domain is not specific.
2. It is freely available for download.
3. It is not specially designed to combine with WordNet for facilitating the clustering result.
4. It had been compared in the experiments of the FIHC method.

4.2 Clustering Evaluation

The *F-measure* defined by Formula (6) is employed to evaluate the accuracy of the generated clustering results. Hence, we define a set of target clusters generated from our approach, denote *C*, and another set is natural clusters, denoted *C'*, which each documents is pre-defined into a single cluster. $doc(c_j)$ is the number of documents in target cluster c_j. Moreover, $doc(c'_i, c_j)$ is used to denote the number documents both in a natural cluster c'_i and a target cluster c_j. Then, the F-measure $F(c'_i, c_j)$ is as:

$$F(c_i', c_j) = \frac{2 \times r(c_i', c_j) \times p(c_i', c_j)}{r(c_i', c_j) + p(c_i', c_j)}, \quad \text{where } r(c_i', c_j) = \frac{doc(c_i', c_j)}{doc(c_i')} \text{ and } p(c_i', c_j) = \frac{doc(c_i', c_j)}{doc(c_j)} \tag{6}$$

Fung *et al.* [4] measured the quality of a clustering result *C* using the weighted sum of such maximum F-measures for all natural clusters according to the cluster size. Let $|D|$ denote the number of all documents in the documents set *D*. This measure is called the overall F-measure of *C,* denote $F(C)$ by Formula (7):

[2] http://www.daviddlewis.com/resources/testcollections/

$$F(C) = \sum_{c_i \in C'} \frac{doc(c_i')}{|D|} \max_{c_j \in C} \{F(c_i', c_j)\} \tag{7}$$

In general, the higher the $F(C)$ values, the better the clustering solution is.

4.3 Results

Table 1 presents the obtained F-Measure values for F^{2}IDC, FIHC, HFTC, and UPGMA by comparing four different numbers of clusters, namely 3, 15, 30, and 60. The results show that our algorithm, F^{2}IHC, outperforms other algorithms in terms of accuracy, specifically on the *Re0* dataset.

Table 1. Comparison of the F-Measure

Data Set (# of Natural Clusters)	# of Clusters	F^{2}IDC	FIHC	HFTC	UPGMA
Re0 (13)	3	0.52 *	0.40	N.A.	0.36
	15	0.54 *	0.41	N.A.	0.47
	30	0.54 *	0.38	N.A.	0.42
	60	0.54 *	0.40	N.A.	0.34
	Average	0.54 *	0.40	0.43	0.40
Reuters (65)	3	0.58	0.60 *	N.A.	×
	15	0.60 *	0.55	N.A.	×
	30	0.60 *	0.57	N.A.	×
	60	0.60 *	0.57	N.A.	×
	Average	0.60 *	0.57	0.49.	×

× means not available for large datasets * means the best competitor

The experimental results of HFTC and UPGMA are the same as that of FIHC.

5 Conclusion

Although numerous document clustering methods have been extensively studied for many years, there still exist several challenges for increasing the clustering quality. In this paper, we derived an effective Fuzzy Frequent Itemset-based Document clustering (F^{2}IDC) approach that combines fuzzy association rule mining with the external knowledge, WordNet, to alleviate these problems. The advantage of considering hypernyms of WordNet in the process of document clustering is to extract more conceptual cluster labels represented by conceptual terms instead of isolated terms. Moreover, the extracted cluster labels help for identifying the content of individual clusters. Moreover, our experiments show that the proposed algorithm has better accuracy quality than FIHC, HFTC, and UPGMA methods compared on Reuters-21578 dataset.

Acknowledgements. This research was partially supported by National Science Council, Taiwan, ROC, under Contract No. NSC 96-2416-H-327-008-MY2 and No. NSC 96-2221-E-009-168-MY2.

References

1. Beil, F., Ester, M., Xu, X.: Frequent Term-based Text Clustering. In: Int'l. Conf. on knowledge Discovery and Data Mining (KDD 2002), pp. 436–442 (2002)
2. Cutting, D.R., Karger, D.R., Pederson, J.O., Tukey, J.W.: Scatter/gather: a Cluster-based approach to Browsing Large Document Collections. In: 15th Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 318–329 (1992)
3. de Campos, L.M., Moral, S.: Learning Rules for a Fuzzy Inference Model. J. Fuzzy Sets and Systems. 59, 247–257 (1993)
4. Fung, B., Wang, K., Ester, M.: Hierarchical Document Clustering Using Frequent Itemsets. In: SIAM Int'l Conf. on Data Mining (SDM 2003), pp. 59–70 (2003)
5. Hong, T.P., Lin, K.Y., Wang, S.L.: Fuzzy Data Mining for Interesting Generalized Association Rules. J. Fuzzy Sets and Systems 138(2), 255–269 (2003)
6. Hotho, A., Staab, S., Stumme, G.: Wordnet Improves Text Document Clustering. In: SIGIR Int'l Conf. on Semantic Web Workshop (2003)
7. Kaya, M., Alhajj, R.: Utilizing genetic algorithms to optimize membership functions for fuzzy weighted association rule mining. Applied Intelligence 24(1), 7–15 (2006)
8. Kushal Dave, D.M.P., Lawrence, S.: Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In: 12th Int'l Conf. on World Wide Web (WWW 2003), pp. 519–528 (2003)
9. Martín-Bautista, M.J., Sánchez, D., Chamorro-Martínez, J., Serrano, J.M., Vila, M.A.: Mining web documents to find additional query terms using fuzzy association rules. Fuzzy Sets and Systems 148(1), 85–104 (2004)
10. Miller, G.A.: WordNet: a Lexical Database for English. J. Communications of the ACM 38(11), 39–41 (1995)
11. Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
12. Sedding, J., Kazakov, D.: WordNet-based Text Document Clustering. In: COLING 2004 Workshop on Robust Methods in Analysis of Natural Language Data (2004)
13. Steinbach, M., Karypis, G., Kumar, V.: A Comparison of Document Clustering Techniques. In: KDD-2000 Workshop on Text Mining, ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD) (2000)
14. Yu, H., Searsmith, D., Li, X., Han, J.: Scalable Construction of Topic Directory with Nonparametric Closed Termset Mining. In: ICDM 2004, pp. 563–566 (2004)

Nonlinear Data Analysis Using a New Hybrid Data Clustering Algorithm

Ureerat Wattanachon[1], Jakkarin Suksawatchon[1], and Chidchanok Lursinsap[2]

[1]Department of Computer Science, Faculty of Science, Burapha University, Chonburi, 20131 Thailand
{ureerat,jakkarin}@buu.ac.th
[2]Advanced Virtual and Intelligent Computing (AVIC) Center Department of Mathematics, Chulalongkorn University, Bangkok 10330, Thailand
lchidcha@chula.ac.th

Abstract. Existing clustering algorithms, such as single-link clustering, k-means, CURE, and CSM are designed to find clusters based on pre-defined parameters specified by users. These algorithms may be unsuccessful if the choice of parameters is inappropriate with respect to the data set being clustered. Most of these algorithms work very well for compact and hyperspherical clusters. In this paper, a new hybrid clustering algorithm called *Self-Partition and Self-Merging* (SPSM) is proposed. The SPSM algorithm partitions the input data set into several subclusters in the first phase and, then, removes the noisy data in the second phase. In the third phase, the normal subclusters are continuously merged to form the larger clusters based on the inter-cluster distance and intra-cluster distance criteria. From the experimental results, the SPSM algorithm is very efficient to handle the noisy data set, and to cluster the data sets of arbitrary shapes of different density.

Keywords: Data Clustering, Unsupervised Learning Algorithm, Data Analysis.

1 Introduction

Data clustering process groups a set of data into similar and dissimilar subsets. Important applications of data clustering can be found in areas such as grouping, decision-making, and machine learning, including data mining, document retrieval, image segmentation, and pattern classification [1,5,6]. Traditionally, clustering method can be broadly classified into the *partition and hierarchical clustering* algorithms. The *k-means* algorithm is one of the most famous partitional clustering algorithms because of its simplicity and easy implementation. Although it is widely used, the k-means algorithm suffers from some drawbacks [9]. Firstly, there is no efficient method for identifying the initial partitions and the number of clusters k. Second, it often terminates at a local optima and, it

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 160–171, 2009.
© Springer-Verlag Berlin Heidelberg 2009

is also sensitive to the noisy data. On the other hand, the well-known hierarchical clustering algorithms are single-link, complete-link, average-link, and so on [4]. These hierarchical clustering algorithms give incorrectness results when the clusters are close to one another even by noises, or when their shapes and sizes are not hyper-spherical.

To alleviate these drawbacks, the *hybrid* idea or two-phase clustering is introduced. Firstly, this strategy partitions the data set into the pre-specified number of sub-clusters and, then, continuously merges the sub-clusters based on similarity criteria in a hierarchical manner. This process continues until the exactly desired number of clusters is determined. Some of the hybrid clustering methods are: Balanced Iterative Reducing and Clustering using Hierarchies – BIRCH [10], Clustering Using representatives – CURE [3] and Cohesion-based Self-Merging – CSM [7].

At present, the development of the existing clustering algorithms concentrates on the computational resources rather than the *quality* of clustering result. In general, the development of clustering technique for a good quality result should consider the following requirements [2]: (1) minimal requirements of domain knowledge to determine input parameters, (2) discovery of clusters with arbitrary shapes, (3) ability to deal with the noisy clusters. As aforementioned, these well known clustering algorithms offer no solution to the combination of clustering requirements. Most of these methods do not produce a suitable estimation of the number of output clusters by themselves. Since they require pre-specified parameters, these algorithms may be unsuccessful if the choice of static parameters is incorrect with respect to the data set being clustered. Although the concept of noise removal allows these clustering algorithms to find clusters of arbitrary shapes and sizes in some data sets, these algorithms still have some troubles when clusters have different densities in the data set.

This work is focused on developing a new method for clustering data which is based on hybrid approach named *Self-Partition and Self-Merging* (SPSM) algorithm. Our idea is, first, to partition data points into several small sub-clusters using *self-partition* method (Phase 1). Each of sub-cluster may be all noisy data or all non-noisy data. In order to achieve a better clustering result and less noisy effects, the sub-clusters are separated into *normal* sub-clusters and *noisy* sub-clusters based on our density threshold in *noise removal* algorithm (Phase 2). Thereafter, those normal sub-clusters are continuously combined into larger clusters based on the homogeneity within each cluster and the closeness between two clusters in *self-merging* algorithm (Phase 3). With our hybrid approach, SPSM algorithm is capable to find clusters with different shapes and sizes as well as different densities without knowing the number of clusters in advance. Besides, it can handle any large data sets and successfully handles the noisy data sets. SPSM algorithm can conform with all clustering requirements to obtain a good clustering result. The rest of the paper is organized as follows. Section 2 presents the background material related to the proposed method. Section 3 introduces the self-partition algorithm. Section 4 addresses the noise removal process. The

self-merging procedure is described in Section 5. The experimental results and discussion are given in Section 6. The conclusion is in Section 7.

2 Background

The major focus of this section is to show all the measurements used in SPSM algorithm. We begin by defining the centroid and homogeneity for one cluster. Let $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, be a set of points in cluster c_p.

Definition 1. *The average intra-cluster distance of the cluster c_p, denoted by $D_{avgintra}(c_p)$, is the average distance of all nearest neighbor pairwise distances within cluster c_p defined as*

$$D_{avgintra}(c_p) = \frac{1}{N} \sum_{i=1}^{N} \left(MIN_{\forall \boldsymbol{x}_j \in c_p \,;\, i \neq j} \, \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \right)$$

where $\|\cdot\|$ is the Euclidean norm.

Definition 2. *The centroid distance $D_{centroid}(c_p, c_q)$ between two centers $\boldsymbol{m}_p$ and $\boldsymbol{m}_q$ of two clusters c_p and c_q, respectively, is defined as*

$$D_{centroid}(c_p, c_q) \; = \; \|\boldsymbol{m}_p - \boldsymbol{m}_q\|.$$

Definition 3. *The maximum intra-cluster distance of the cluster c_p, denoted by $D_{maxintra}(c_p)$, is the maximum distance selected from all nearest neighbor pairwise distances within cluster c_p defined as:*

$$D_{maxintra}(c_p) = MAX_{\forall \boldsymbol{x}_i \in c_p} \left(MIN_{\forall \boldsymbol{x}_j \in c_p \,;\, i \neq j} \, \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \right).$$

Definition 4. *The minimum inter-cluster distance $D_{mininter}(c_p, c_q)$ is the maximum of minimum distance chosen from the pairwise distances between two clusters c_p and c_q defined as*

$$D_{mininter}(c_p, c_q) = MAX(MIN_{\forall \boldsymbol{bx}_i \in c_p, \forall \boldsymbol{bx}_j \in c_q} \left(\|\boldsymbol{bx}_i - \boldsymbol{bx}_j\| \right),$$
$$MIN_{\forall \boldsymbol{bx}_i \in c_p, \forall \boldsymbol{bx}_j \in c_q} \left(\|\boldsymbol{bx}_j - \boldsymbol{bx}_i\| \right)),$$

where $\boldsymbol{bx}_i$ and $\boldsymbol{bx}_j$ are the boundary data points of the clusters c_p and c_q, respectively.

Definition 5. *The maximum inter-cluster distance $D_{maxinter}(c_p, c_q)$ is the maximum distance selected from the pairwise distances between two clusters c_p and c_q defined as*

$$D_{maxinter}(c_p, c_q) = MAX(MAX_{\forall \boldsymbol{bx}_i \in c_p, \forall \boldsymbol{bx}_j \in c_q} \left(\|\boldsymbol{bx}_i - \boldsymbol{bx}_j\| \right),$$
$$MAX_{\forall \boldsymbol{bx}_i \in c_p, \forall \boldsymbol{bx}_j \in c_q} \left(\|\boldsymbol{bx}_j - \boldsymbol{bx}_i\| \right)).$$

3 Phase 1: Self-partition

The main idea of this process is to partition the data set to form a number of small sub-clusters by recursively subdividing the data set into four sub-clusters according to data distribution. The partitioning process can be carried out by the proposed self-partitioned clustering model called *Dynamic Tree-Structured Self-Organizing Map* (DTS-SOM) algorithm.

3.1 Architecture of the DTS-SOM Algorithm

The initial topology of DTS-SOM is a two-dimension grid-tree structure. The tree used in DTS-SOM is a *4-ary* tree. Each node, accepting the leaves, has four descendants. Each leaf node represents each prototype vector of each sub-cluster. These leaves are hierarchically clustered into groups of four nodes. Each prototype vector i is denoted by $\mathbf{w}_i = (w_{i1}, w_{i2}, \ldots, w_{id})$. In addition, each node $\mathbf{w}_i$ in the network has a counter τ_i denoting the number of times this node has been the best matched unit during the DTS-SOM training. Each node has the lateral connections among neighboring nodes defined as the direct left, right, top, bottom, top-left, bottom-left, top-right, and bottom-right neighboring nodes as shown in Fig. 1.

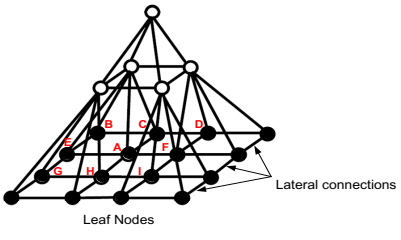


Fig. 1. A tree-structured topology of DTS-SOM algorithm at certain learning stage. (a) A is the best matching unit, and B, C, D, E, F, G, H, I are its direct neighbors.

3.2 Training of the DTS-SOM

The DTS-SOM algorithm starts by taking a single node from the available set of data vectors in a random manner. Then, the node is split to generate four new nodes and their counters are initialized. During the DTS-SOM training, the neural nodes are arranged in a tree topology and allowed to grow when any given branch receives a lot of times being the best matching unit from the training data vectors. The search for the best matching unit and its neighbors is conducted along the tree. The DTS-SOM algorithm is repeated for every vector on the training set until the stopping criterion is reached. The size of the tree is examined at the end of each epoch and compared with the size of the tree from the previous epoch. If their difference is less than a small threshold value, then the training is completed.

Algorithm. *DTS-SOM_Training*
1. Initialize the DTS-SOM network as described at the beginning of Sec. 3.2.
2. **repeat**
3. Select an input datum $\mathbf{x}_j(t)$ randomly from input space.
4. Find the best matching unit b using the tree search.
5. Update weight vectors of the best matching unit b and its direct neighboring nodes using Kohonen's rule.
6. Update their parents layer by layer by using $\mathbf{w}_i(t+1) = \frac{1}{nc_i} \sum_{\forall k} \mathbf{w}_k(t)$. where nc_i is the number of chidren of parent node i.
7. Decrease the counter τ_b of the best matching unit b by 1.
8. **if** the counter τ_b decreases to zero **then**
9. Generate four new neurons of original neuron b.
10. Initialize their new weights and their counter τ_i.
11. **until** the tree has grown less than a small threshold value.

4 Phase 2: Noise Removal

After Phase 1, the sub-clusters with low density can be considered as noisy sub-clusters while the others are considered as normal sub-clusters. The main process of Phase 2 is to filter out a majority of the noisy sub-clusters based on density distribution. The whole process of noise removal is composed of the following three steps.

4.1 Density Computation

Before finding the sub-cluster density, it is observed that the expected normal sub-clusters have some deviated points. These points will affect the correctness of density computation. Thus, all the data points in each sub-cluster are examined. The detail of of finding sub-cluster density is described in the following algorithm.

Algorithm. *DensityFinding*
1. **for** each sub-cluster c_p
2. **do if** the number of data points in cluster c_p $(|c_p|) > 1$ **then**
3. Compute all eigenvalues λ_k and eigenvectors $\mathbf{v}_k$ of sub-cluster c_p.
4. Compute $Volume(c_p) = \sqrt{\lambda_1} \times \ldots \times \sqrt{\lambda_i} \times \ldots \times \sqrt{\lambda_d}$
5. **for** each point $\mathbf{x}_i \in c_p$
6. **do** Compute $l = \|\mathbf{m}_p - \mathbf{x}_i\|$, where $\mathbf{m}_p$ is the cluster center.
7. **for** each eigenvector $\mathbf{v}_k$
8. **do** Compute angle θ_k between $\mathbf{v}_k$ and $\mathbf{x}_i$.
9. Compute projected line segment $l = l \times cos(\theta_k)$.
10. **if** $l > \sqrt{\lambda_k}$ **then**

Remove the point $\mathbf{x}_i$ as the noisy point $\mathbf{nx}_i$ and break.
Compute $Density(c_p) = \frac{|c_p|}{Volume(c_p)}$.
if $|c_p| \leq 1$
  then Remove $\mathbf{x}_i$ as $\mathbf{nx}_i$ and set $Density(c_p) = 0$
else Remove $\mathbf{x}_i$ as $\mathbf{nx}_i$ and set $Density(c_p) = 0$

4.2 Cluster Separation

The purpose of this stage is to separate the sub-clusters without the deviated data points into two classes– the normal sub-clusters and the noisy sub-clusters – based on the density threshold value. The question is how to set this density threshold value. The problem can be solved by sorting all the sub-cluster densities in ascending order and arranging them in terms of a logarithm scale with each scale ranging in 10 times. Presentation of densities on a logarithmic scale can be used to analyze the characteristic of the densities. Fig. 2 shows the example of density arrangement in terms of a logarithm scale. From the density arranging experiments of several data sets, it is not necessary to consider all density ranges to set the threshold value. Therefore, some density ranges can immediately identify to the densities of normal sub-clusters or noisy sub-clusters because the number of densities in such ranges is less with respect to the number of densities in the other ranges. As the result in Fig. 2, density range 1-10 can be set as the densities of noisy sub-clusters, and density range 1000-10000 can be considered as the densities of normal sub-clusters. From this observation, we consider the density range having the most number of densities to determine the threshold value and, then, we set this range as *RangeI*. Then, we select the another range, named *RangeII*, whose density range is less than the density range of RangeI. In this work, the threshold value $thres_{density}$ will separate all densities in RangeI and RangeII into two sets: the first set (densities of the noisy sub-clusters) and the second set (densities of the normal sub-clusters), based on the coefficient of variation (CV) [8]. Thus, the density threshold value should be set at the position such that the CV of the first set is greater than the CV of the second set. This implies that the variation of the first set is more than the second set.

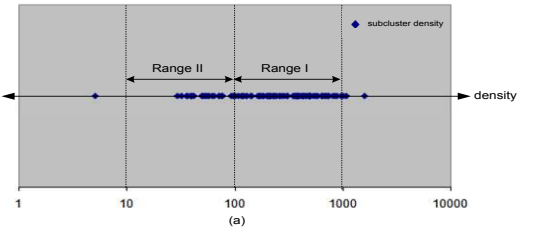


Fig. 2. (a) The example of the density arrangement of one data set. Density ranges, RangeI and RangeII, are used to find the density threshold value.

Algorithm. *ClusterSeparating*
1. **for** each sub-cluster c_p
2. **do if** $Density(c_p) < thres_{density}$
3. **then** Set sub-cluster c_p as noisy sub-cluster nc_x.
4. **else** Set sub-cluster c_p as normal sub-cluster c_p.
5. Let $density_{mean}$ be the average density of normal sub-clusters c_p.
6. Let $density_{std}$ be the standard deviation of normal sub-clusters c_p.
7. Compute $density_{mean}$.
8. Compute $density_{std}$.
9. **for** each normal sub-clusters c_p
10. **do if** $Density(c_p) < (density_{mean} - density_{std})$
11. **then** Set normal sub-cluster c_p as noisy sub-cluster nc_x.

5 Phase 3: Self-merging

The self-merging approach is the process of iterative joining neighboring sub-clusters into larger clusters according to similarity constraints. To obtain the complete results, the algorithm consists of the multiple sequential merging steps.

5.1 Neighboring Merging

Merging any two neighboring sub-clusters depends upon the noisy points lying in between them. Therefore, in this merging step, all removed noisy points must be placed back to their original locations. Then, an imaginary region called *partial region* overlapping these two sub-clusters is created to determine the merging possibility. We define the partial region as the hyper-box whose side lengths depend on how side lengths of two normal sub-clusters overlap. In our approach, each side length of any normal sub-clusters is the data range of each dimension. Fig. 3(a) shows the partial region occurred when the side lengths of two normal sub-clusters do not overlap but either side length of two normal sub-clusters in Fig. 3(b) overlaps. After that, the neighboring merging algorithm is performed in the following algorithm.

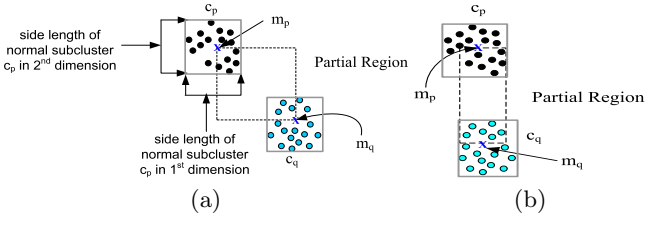


Fig. 3. An example of the 2-dimensional partial regions (hyper-boxes) illustrated by the dashed boxes. (a) The partial region of non-overlapping normal sub-clusters. (b) The partial region of overlapping normal sub-clusters. Each gray box represents the normal sub-cluster boundary. The normal sub-cluster centers are indicated by the symbol "x" which its dimension is the same as the dimension of data set, i.e. $\mathbf{m}_p = (m_{p1}, m_{p2}, \ldots, m_{pd})$ and $\mathbf{m}_q = (m_{q1}, m_{q2}, \ldots, m_{qd})$, where d is the dimension of data set.

Algorithm. *NeighboringMerging*

1. $g = 1$.
2. **while** all neighboring normal sub-clusters c_p are not merged
3. **do** Choose a neighboring normal sub-cluster c_p having the highest density as a new cluster c_g.
4. **while** the number of sub-clusters in cluster c_g does not change
5. **do** Compute $D_{maxintra}(c_g)$.
6. **repeat**
7. Compute $D_{centroid}(c_p, c_q)$.
8. **if** $D_{centroid}(c_p, c_q) < (D_{avgcentroid} - D_{stdcentroid})$
9. **then** Compute $D_{maxintra}(c_p)$.
10. Compute $D_{mininter}(c_g, c_p)$.
11. **if** $D_{mininter}(c_g, c_p) \leq Max(D_{maxintra}(c_g), D_{maxintra}(c_p))$
12. **then** Set a neighboring normal sub-cluster c_p to the same cluster c_g.
13. **until** no neighboring normal sub-clusters c_p of normal sub-clusters c_q in c_g merged.
14. Increase g by 1.

5.2 Local Merging I

The objective of this stage is to further aggregate the closest clusters. This step is shown as follws:

Algorithm. *LocalMergingI*

1. **while** there are clusters c_g merged.
2. Choose cluster c_g having a highest number of data points as the starting seed.
3. **while** there are clusters c_p merged to cluster c_g.
4. Compute $D_{avgintra}(c_g)$ of cluster c_g.
5. **for** each cluster c_p such that $g \neq p$
6. **do** Compute $D_{avgintra}(c_p)$ of cluster c_p.
7. Compute $D_{mininter}(c_g, c_p)$ clusters c_g and c_p.
8. **if** $D_{mininter}(c_g, c_p) \leq Max(D_{avgintra}(c_g), D_{avgintra}(c_p))$
9. **then** Set cluster c_p to the same cluster c_g.

5.3 Local Merging II

Before performing further merging process, the large clusters based on the threshold value are temporarily removed. Once the number of data points for each cluster is sorted in descending order, the threshold value can be set as the number of data points such that the difference between the number of data points in clusters c_g and c_{g+1} is maximum. After removing the large clusters, the further merging steps are shown in the *LocalMergingII* algorithm.

Algorithm. *LocalMergingII*

1. **while** there are clusters c_g merged.
2. Choose small cluster c_g having a highest number of data points as the starting seed.
3. **while** there are clusters c_p merged to cluster c_g.
4. Compute $D_{avgintra}(c_g)$ of cluster c_g.
5. **for** each small cluster c_p such that $g \neq p$
6. **do** Compute $D_{avgintra}(c_p)$ of cluster c_p.

```
7.                    Compute D_mininter(c_g, c_p) clusters c_g and c_p.
8.                    if D_mininter(c_g, c_p) ≤ Max(D_avgintra(c_g), D_avgintra(c_p))
9.                       then Set cluster c_p to the same cluster c_g.
10.                      else  if D_centroid(c_g, c_p) < D_avgcentroid
11.                                then Set cluster c_p to the same cluster c_g.
```

5.4 Refinement Merging

This last step is used for refining the clusters to obtain the final clusters. After the final clusters are obtained, each the remaining noisy point is assigned to be the member of the cluster. However, there are still some noisy points which cannot belong to any clusters since these points are very deviated from the others. Thus, these points are set as noise. The algorithm of this stage is described in the *RefinementMerging* algorithm.

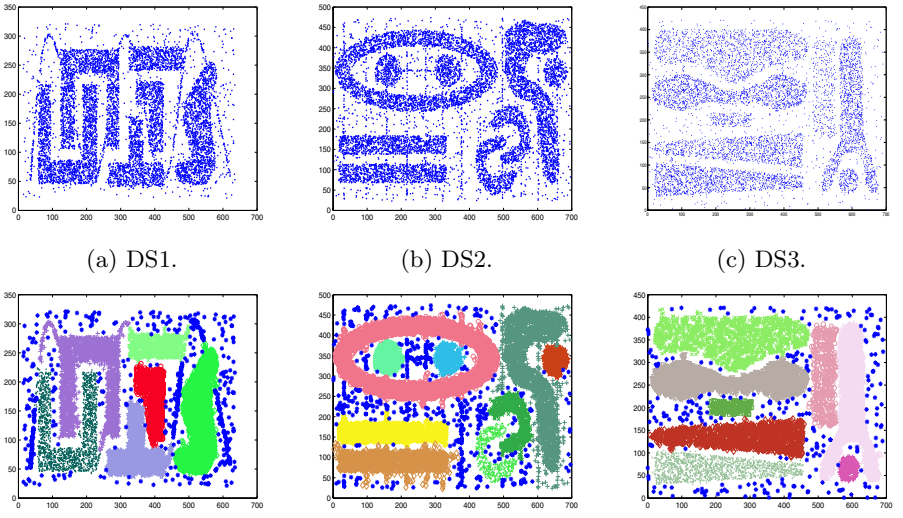
Algorithm. *RefinementMerging*

```
1.  while there are clusters c_g merged.
2.        Choose cluster c_g having a highest number of data points as the starting
          seed.
3.        while there are clusters c_p merged to cluster c_g.
4.              Compute D_maxintra(c_g) of cluster c_g.
5.              for each cluster c_p such that g ≠ p
6.                  do Compute D_maxintra(c_p) of cluster c_p.
7.                     Compute D_maxinter(c_g, c_p) clusters c_g and c_p.
8.                     if D_maxinter(c_g, c_p) ≤ 2×Max(D_maxintra(c_g), D_maxintra(c_p))
9.                        then Set cluster c_p to the same cluster c_g.
```

6 Experimental Results and Discussion

The purposes of this section is to measure the performance of the proposed SPSM algorithm in the complicated data sets and compare them to the clustering algorithms such as the single-link, complete-link, CURE and CSM algorithm. These data sets are consisted of DS1, DS2 and DS3 obtained from http://glaros.dtc.umn.edu/gkhome/cluto/cluto/download. The clustering results of SPSM algorithm are shown in Fig. 4. Our results show that SPSM algorithm is able to correctly identify the genuine clusters in all three data sets. Besides, these results illustrate that SPSM algorithm is very effective in finding clusters of arbitrary shapes and orientation, and is tolerant to noisy data. These data sets have been tested with CURE algorithm obtained from public domain at http://www.cs.cas.cz/∼petra/Liter-Odkazy-shluk.html.

Some of unsuccessful clustering results acquired from CURE algorithm are shown in Fig. 5. CURE fails to find the right clusters on all these data sets, although we have carefully chosen CURE's parameters. As these results, CURE fails because CURE is also hierarchical clustering algorithm that may produce the wrong results at the desired level. In the cases of DS1, DS2 and DS3, CURE selects the wrong pair of clusters to merge it together before forming a final



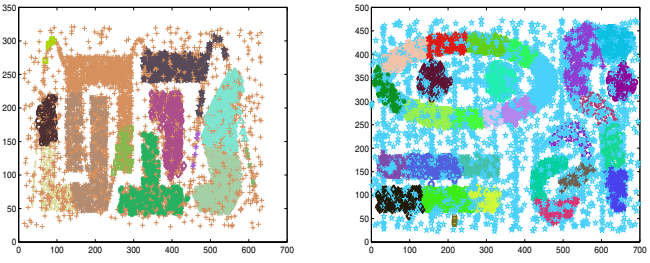
(a) DS1. (b) DS2. (c) DS3.

(d) DS1 (6 clusters). (e) DS2 (9 clusters) (f) DS3 (8 clusters).

Fig. 4. The clustering results produced in SPSM algorithm. Each cluster is denoted by a symbol. The symbol "*" indicates noisy points.

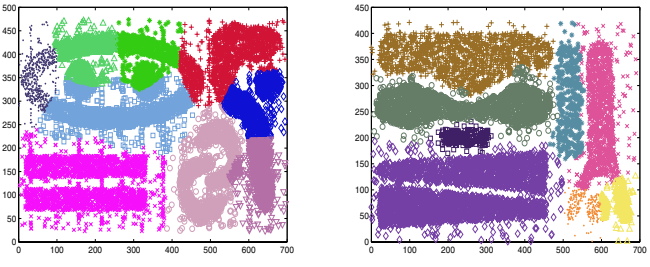
clustering results. Besides, CURE algorithm is not a suitable method for the long piece shapes in data sets. Since the shrinking method causes such clusters to be split if shrinking factor is set too large. On the other hand, if the shrinking factor is chosen too small, some clusters will be merged by noise links.

We also implemented the CSM algorithm with the best attempt because the suggestion of choosing the parameters of CSM algorithm is not provided. Then, SPSM algorithm has been evaluated the performance against CSM algorithm. Fig. 6(a) shows the results of algorithm CSM with the number of sub-clusters, the parameter $m = 16$, respectively. Fig. 6(b) shows the results of algorithm CSM with the parameter $m = 50$, respectively. The number in the parenthesis is the desired number of clusters specified by user. From the results, the CSM algorithm fails to identify the clusters because of the parameter selection. One of the parameters affects the correctness of the clustering results is the parameter m specified by users. If the sub-clusters produced in Phase 1 are be too many then the actual noisy sub-clusters will become dense sub-clusters. This may lead to CSM algorithm makes a mistake by merging the wrong pair of clusters. On the other hand, when the value of parameter m is too small, the number of sub-clusters obtained from Phase 1 will be too low. So many noisy sub-clusters may exist. These noisy sub-clusters may merge with other clusters and affect the clustering results. Besides, the clustering results of CSM algorithm also depend on the user-specified parameters used to compute the threshold value in the noise removal mechanism. In practice, it is hard to set with the proper parameters. Thus, algorithm CSM cannot precisely remove those noisy sub-clusters.



(a) DS1 (> 6 clusters). (b) DS2 (> 9 clusters).

Fig. 5. The clustering results obtained through the algorithm CURE. (a),(b) The final decomposition of DS1 and DS2, respectively. Each cluster is denoted by a symbol.



(a) DS2 (9 clusters). (b) DS3 (8 clusters).

Fig. 6. The clustering results produced by the algorithm CSM. (a),(b) The final decomposition of DS2 and DS3 with the parameter $m = 16$ and $m = 50$, respectively. Each cluster is denoted by a symbol.

7 Conclusion

In this paper, the 3-phase clustering called SPSM algorithm is proposed. SPSM algorithm is not only able to find the clusters of arbitrary shapes and size, but also able to tolerate noise. Furthermore, SPSM algorithm automatically determines the number of clusters. This is more appropriate when the user has no a priori information on the number of clusters presented in the data sets. In addition, SPSM algorithm concentrates on producing the good quality of clustering results rather than to speed up the computational resources. Thus, the experimental results on several varying data sets show that SPSM algorithm is able to find the clusters of different shapes very efficiently, tolerate noise, and provide better clustering results than those clustering algorithms. Besides, SPSM algorithm can discover the clusters of varied densities which most of existing algorithms fail to find.

Acknowledgments

This work was supported in part by the Development and Promotion for Science and Technology talents project of Thailand (DPST).

References

1. Duda, R.O., Hart, P.E., Stock, D.G.: Pattern Classification. John Wiley and Sons, Inc., Chichester (2001)
2. Ester, M., Kriegel, H., Sander, J., Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD 1996), pp. 226–231 (1996)
3. Guha, S., Rastogi, R., Shim, K.: CURE: an efficient clustering algorithm for large databases. In: Proc. ACM SIGMOD Int. Conf. on Management of Data, pp. 73–84 (1998)
4. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall, Englewood Cliffs (1988)
5. Jain, A.K., Duin, R.P.W., Mao, J.: Statistical pattern recognition: a review. IEEE Trans. on Pattern Anal. and Machine Intelligence, 4–37 (2000)
6. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Computing Surveys 31(3), 264–323 (1999)
7. Lin, C., Chen, M.: Combining partitional and hierarchical algorithms for robust and efficient data clustering with cohesion self-merging. IEEE Trans. Knowledge and Data Engineering 17(2), 145–159 (2005)
8. Weisstein, E.W.: Variation Coefficient, `http://mathworld.wolfram.com/Variation-Coefficient.html`
9. Xu, R., Wunsch, D.: Survey of clustering algorithms. IEEE Trans. Neural Networks 16(3), 645–678 (2005)
10. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: An efficient data clustering method for very large databases. In: Proc. Conf. Management of Data (ACM SIGMOD 1996), pp. 103–114 (1996)

A Polynomial-Delay Polynomial-Space Algorithm for Extracting Frequent Diamond Episodes from Event Sequences

Takashi Katoh[1], Hiroki Arimura[1], and Kouichi Hirata[2]

[1] Graduate School of Information Science and Technology, Hokkaido University
Kita 14-jo Nishi 9-chome, Sapporo 060-0814, Japan
Tel.: +81-11-706-7678; Fax: +81-11-706-7890
{t-katou, arim}@ist.hokudai.ac.jp
[2] Department of Artificial Intelligence, Kyushu Institute of Technology
Kawazu 680-4, Iizuka 820-8502, Japan
Tel.: +81-948-29-7622; Fax: +81-948-29-7601
hirata@ai.kyutech.ac.jp

Abstract. In this paper, we study the problem of mining *frequent diamond episodes efficiently* from an input event sequence with sliding a window. Here, a diamond episode is of the form $a \mapsto E \mapsto b$, which means that every event of E follows an event a and is followed by an event b. Then, we design a polynomial-delay and polynomial-space algorithm PolyFreqDmd that finds all of the frequent diamond episodes without duplicates from an event sequence in $O(|\Sigma|^2 n)$ time per an episode and in $O(|\Sigma| + n)$ space, where Σ and n are an alphabet and the length the event sequence, respectively. Finally, we give experimental results on artificial event sequences with varying several mining parameters to evaluate the efficiency of the algorithm.

1 Introduction

It is one of the important tasks for data mining to discover frequent patterns from time-related data. For such a task, Mannila *et al.* [7] have introduced the *episode mining* to discover frequent *episodes* in an *event sequence*. Here, the episode is formulated as an *acyclic labeled digraphs* in which labels correspond to events and edges represent a temporal precedent-subsequent relation in an event sequence. Then, the episode is a richer representation of temporal relationship than a *subsequence*, which represents just a linearly ordered relation in *sequential pattern mining* (*cf.*, [3,9]). Furthermore, since the *frequency* of the episode is formulated by a *window* that is a subsequence of an event sequence under a fixed time span, the episode mining is more appropriate than the sequential pattern mining when considering the time span.

Mannila *et al.* [7] have designed an algorithm to construct episodes from a *parallel episode* as a set of events and a *serial episode* as a sequence of events. Note that their algorithm is general but inefficient. Then, the episode mining

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 172–183, 2009.
© Springer-Verlag Berlin Heidelberg 2009

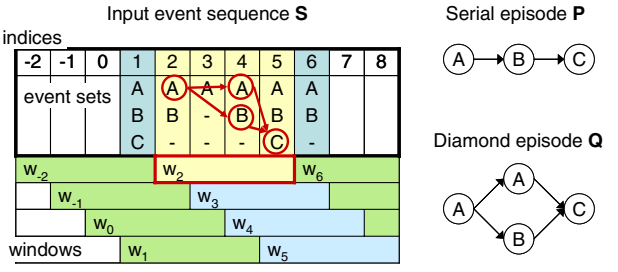


Fig. 1. (Left) An input sequence $\mathcal{S} = (S_1, \ldots, S_6)$ of length $n = 6$ over $\Sigma = \{A, B, C\}$ and their k-windows. (Right) Serial episode $P = A \mapsto B \mapsto C$ and a diamond episode $Q = A \mapsto \{A, B\} \mapsto C$. In the sequence $\mathcal{S}$, we indicate an occurrence (embedding) of Q in the second window W_2 in circles and arrows. See Example 1 and 2 for details.

has been developed by introducing the specific forms of episodes for every target area together with efficient algorithms [5,6,8].

As such specific forms of episodes, Katoh *et al.* have introduced *diamond episodes* [6] and *elliptic episodes* [5]. In Fig. 1, we show examples of an input event sequence, a serial episode, and a diamond episode over an alphabet. Both episodes have the special event types, a *source* and a *sink*. Then, by setting the source and the sink to the specified event types, we can find frequent episodes with the source as a premise and the sink as a consequence. In particular, from bacterial culture data [5,6], they have succeeded to find frequent diamond and elliptic episodes concerned with *the replacement of bacteria* and *the changes for drug resistance*, which are valuable from the medical viewpoint. Here, the source and the sink are set to the bacteria and another bacteria for the former episodes, and the sensitivity of antibiotic and the resistant of the same antibiotic for the latter episodes.

Note that the algorithms designed by Katoh *et al.* [5,6] are so-called *level-wise*; The algorithms first find the information for the occurrences of serial episodes in an event sequence, by scanning it just once. After regarding the serial episodes as itemsets, the algorithms then construct the frequent episodes by using the frequent itemset mining algorithm AprioriTid [1].

While the level-wise algorithms are sufficient to find frequent episodes efficiently in practice (in particular, appropriate to apply the bacterial culture data), it is difficult to give theoretical guarantee of the efficiency to the level-wise algorithms from the view of enumeration. In this paper, as a space-efficient episode mining algorithm, we newly design the *episode-growth* algorithm, called PolyFreqDmd, to enumerate frequent diamond episodes.

The algorithm PolyFreqDmd adopts the depth-first search, instead of the level-wise search. Then, the algorithm finds all of the frequent diamond episodes in an input sequence $\mathcal{S}$ without duplication in $O(|\Sigma|^2 n)$ time per an episode and in $O(|\Sigma| + n)$ space, where Σ and n are an alphabet and the length of $\mathcal{S}$, respectively. Hence, we can guarantee that the episode-growth algorithm enumerates

frequent diamond episodes in polynomial delay and in polynomial space. Further, we presents some practical optimization techniques for reducing the running time and the required space of the algorithm PolyFreqDmd.

This paper is organized as follows. In Section 2, we introduce diamond episodes and other notions necessary to the later discussion. In Section 3, we present the algorithm PolyFreqDmd and show its correctness and the time complexity. In Section 4, we give some experimental results from randomly generated event sequences to evaluate the practical performance of the algorithms. In Section 5, we conclude this paper and discuss the future works.

2 Diamond Episode

In this section, we prepare diamond episodes and the related notions necessary to later discussion. We denote the sets of all integers and all natural numbers by $\mathbf{Z}$ and $\mathbf{N}$, respectively. For a set S, we denote the cardinality of S by $|S|$. Let $\Sigma = \{1, \ldots, m\}$ $(m \geq 1)$ be a finite alphabet with the total order $\leq$ over $\mathbf{N}$. Each element $e \in \Sigma$ is called an *event*[1]. An *input event sequence* (*input sequence*, for short) $\mathcal{S}$ on Σ is a finite sequence $\langle S_1, \ldots, S_n \rangle \in (2^{\Sigma})^*$ of events $(n \geq 0)$, where $S_i \subseteq \Sigma$ is called the i-th *event set* for every $1 \leq i \leq n$. Then, we call n the *length* of $\mathcal{S}$ and denote it by $|\mathcal{S}|$, and define the *total size* of $\mathcal{S}$ by $||\mathcal{S}|| = \sum_{i=1}^{n} |S|$. Clearly, $||\mathcal{S}|| = O(|\Sigma|n)$, but the converse is not always true.

For a fixed input sequence $\langle S_1, \ldots, S_n \rangle \in (2^{\Sigma})^*$, a *position* or an *index* on $\mathcal{S}$ is any integer, where we allow an index out of $\mathcal{S}$ by defining that $S_i = \emptyset$ if i is either $i \leq 0$ or $i > n$. Let $1 \leq k \leq n$ be a fixed positive integer, called the *window width*. For any index $-k+1 \leq i \leq n$, we define the *k-window $W_i^{\mathcal{S},k}$ at position i in $\mathcal{S}$* by the contiguous subsequence of length k of $\mathcal{S}$ as follows: $W_i^{\mathcal{S},k} = w_{\mathcal{S}}(i,k) = \langle S_i, \ldots, S_{i+k-1} \rangle \in (2^{\Sigma})^k$. We denote the set $\{ W_i^{\mathcal{S},k} \mid -k+1 \leq i \leq n \}$ of all k-windows in $\mathcal{S}$ by $\mathbf{W}_{\mathcal{S},k}$. We simply write W_i and $\mathbf{W}$ instead of $W_i^{\mathcal{S},k}$ and $\mathbf{W}_{\mathcal{S},k}$ by omitting the scripts $\mathcal{S}$ and k, when they are clear from the context. Moreover, we may identify the set of all k-windows by the set $\{ -k+1 \leq i \leq n \mid i \in \mathbf{W}_{\mathcal{S},k} \} \subseteq \mathbf{Z}$ of their indices.

A *serial episode* over Σ of length $m \geq 0$ (or, *m-serial episode*) is a sequence $P = \langle a_1, \ldots, a_m \rangle \in \Sigma^*$ of events.

Definition 1. A *diamond episode* over Σ is either an event $a \in \Sigma$ (a 1-serial episode) or a triple $P = \langle a, E, b \rangle \in \Sigma \times 2^{\Sigma} \times \Sigma$ (called a *proper* diamond episode), where $a, b \in \Sigma$ are events and $E \subset \Sigma$ is a subset of Σ. Then, we call a, b, and E the *source*, the *sink*, and the *body* of P, respectively. For the body E, we denote the maximum element in E (w.r.t. Σ) by $\max(E)$.

To emphasize the chronological dependencies of events, we often write $(a_1 \mapsto \cdots \mapsto a_m)$ and $(a \mapsto E \mapsto b)$ for an m-serial episode $\langle a_1, \ldots, a_m \rangle$ and a diamond episode $\langle a, E, b \rangle$, respectively. Also we denote the classes of m-serial episodes,

[1] Mannila *et al.* [7] originally referred to each element $e \in \Sigma$ itself as an *event type* and an occurrence of e as an *event*. However, we simply call both of them as *events*.

proper diamond episodes and diamond episodes (over Σ) by $\mathcal{SE}_m$, $\mathcal{PDE}$ and $\mathcal{DE}$, respectively. Since any $(a \mapsto b) \in \mathcal{SE}_2$ and any $(a \mapsto b \mapsto c) \in \mathcal{SE}_3$ are equivalent to $(a \mapsto \emptyset \mapsto b)$ and $(a \mapsto \{b\} \mapsto c) \in \mathcal{PDE}$, respectively, we see that $\mathcal{SE}_1 \cup \mathcal{SE}_2 \cup \mathcal{SE}_3 \cup \mathcal{PDE} = \mathcal{DE}$.

Example 1. In Figure 1, we show examples of an event sequence $\mathcal{S} = (ABC, AB, A, AB, ABC, AB)$ of length $n = 6$, a serial episode $P = A \mapsto B \mapsto C$ and a diamond episode $Q = A \mapsto \{A, B\} \mapsto C$ on an alphabet of events $\Sigma = \{A, B, C\}$, where the body $\{A, C\}$ is written as a sequence AC.

Next, we introduce the concept of occurrences of episodes in a window. Then, we give the formal definition of the occurrences of episodes, which is consistent with the original definition by [7]. Let $P = e_1 \mapsto \cdots \mapsto e_n$ be a serial episode, $Q = e_{\mathrm{s}} \mapsto \{e_1, \ldots, e_m\} \mapsto e_{\mathrm{t}}$ a diamond episode and $W = \langle S_1 \cdots S_k \rangle \in \mathbf{W}_{\mathcal{S},k}$ a window, respectively. A serial episode $P = e_1 \mapsto \cdots \mapsto e_m$ *occurs in* an window $W = \langle S_1 \cdots S_k \rangle \in \mathbf{W}_{\mathcal{S},k}$, denoted by $P \sqsubseteq W$, iff there exists some mapping $h : \{1, \ldots, m\} \to \{1, \ldots, k\}$ satisfying (i) $1 \leq h(1) < \cdots < h(m) \leq k$, and (ii) $e_i \in S_{h(i)}$ holds for every $1 \leq i \leq m$.

Definition 2 (occurrence for a diamond episode). A diamond episode $P = e_{\mathrm{s}} \mapsto \{e_1, \ldots, e_m\} \mapsto e_{\mathrm{t}}$ $(m \geq 0)$ *occurs in* an window $W = \langle S_1 \cdots S_k \rangle \in \mathbf{W}_{\mathcal{S},k}$, denoted by $D \sqsubseteq W$, iff there exists some mapping $h : \{\mathrm{s}, \mathrm{t}, 1, \ldots, m\} \to \{1, \ldots, k\}$ satisfying (i) for every $i \in \{1, \ldots, m\}$, $1 \leq h(\mathrm{s}) < h(i) < h(\mathrm{t}) \leq k$ holds, and (ii) $e_i \in S_{h(i)}$ holds for every $i \in \{\mathrm{s}, \mathrm{t}, 1, \ldots, m\}$.

For a window W and an event $e \in \Sigma$, we denote the first and the last position in W at which e occurs by $st(e, W)$ and $et(e, W)$, respectively. The matching algorithm for diamond episodes will be studied in Section 3.

For an episode P, we define the *occurrence list* for P in $\mathcal{S}$ by $\mathbf{W}_{\mathcal{S},k}(P) = \{ -k + 1 \leq i \leq n \,|\, P \sqsubseteq W_i \}$, the set of occurrences of P in an input $\mathcal{S}$. We may call the element $i \in \mathbf{W}_{\mathcal{S},k}(P)$ a *label*. If $i \in \mathbf{W}_{\mathcal{S},k}(P)$, then we say that an episode P *occurs in* $\mathcal{S}$ at position i or at the i-th window.

Example 2. Consider an input event sequence $\mathcal{S} = (ABC, AB, A, AB, ABC, AB)$ in Figure 1. Then, if the window width k is 4, has nine 4-windows from W_{-2} to W_6 for all $-2 \leq i \leq 6$, i.e., $\mathbf{W}_{\mathcal{S},5} = \{ W_i \mid -2 \leq i \leq 6 \}$. Among them, the window list for a diamond episode $P = A \mapsto AB \mapsto C$ is $\mathbf{W}(P) = \{W_2, W_3\}$.

Lemma 1. *Let P be a partial diamond episode $(e_{\mathrm{s}} \mapsto E \mapsto e_{\mathrm{t}})$ and W a window in $\mathbf{W}_{\mathcal{S},k}$. Then, $P \sqsubseteq W$ iff for every $e \in E$, there exists some position p for e such that $e \in S_p$ and $st(e_{\mathrm{s}}, W) < p < et(e_{\mathrm{t}}, W)$ hold.*

Proof. (Only if-direction) If $P \sqsubseteq W$ then there exists some embedding h from P to W. By restricting h to the serial episode $(e_{\mathrm{s}} \mapsto \cdots \mapsto e_{\mathrm{t}})$, we obtain the claim. (If-direction) Suppose that for every $e \in E$, there exists a position p_e for e with $st(e_{\mathrm{s}}, W) < p < et(e_{\mathrm{t}}, W)$. Then, we can build a mapping h by $h(e_{\mathrm{s}}) = st(e_{\mathrm{s}}, W)$, $h(e_{\mathrm{t}}) = et(e_{\mathrm{t}}, W)$, and $h(e) = p_e$ for every $e \in E$. Then, the claim holds. □

Lemma 1 implies the following two important corollaries.

Corollary 1 (serial construction of diamond episodes). *Let P be a partial diamond episode $(e_s \mapsto E \mapsto e_t)$ $(m \geq 0)$ and W a window in $\boldsymbol{W}_{\mathcal{S},k}$. Then, $P \sqsubseteq W$ iff $(e_s \mapsto e \mapsto e_s) \sqsubseteq W$ for every $e \in E$.*

Corollary 2 (anti-monotonicity for diamond episodes). *Let $a, b \in \Sigma$ be events, and $E, F \subseteq \Sigma$ be event sets. For every frequency threshold σ and window width $k \geq 1$, if $E \supseteq F$, then $(a \mapsto E \mapsto b) \in \mathcal{F}_{\mathcal{S},k,\sigma}$ implies $(a \mapsto F \mapsto b) \in \mathcal{F}_{\mathcal{S},k,\sigma}$.*

Let $\mathcal{S}$ be an input sequence, $k \geq 1$ a window width and P a diamond episode $a \mapsto E \mapsto b$. The (absolute) *frequency* of P in $\mathcal{S}$ is defined by the number of k-windows $freq_{\mathcal{S},k}(P) = |\mathbf{W}_{\mathcal{S},k}(P)|$. A *minimum frequency threshold* is any integer $1 \leq \sigma \leq |\mathbf{W}_{\mathcal{S},k}|$. A diamond episode P is *σ-frequent in $\mathcal{S}$* if $freq_{\mathcal{S},k}(P) \geq \sigma$. Note that the frequency is an absolute value, while the support is a relative value. We denote by $\mathcal{F}_{\mathcal{S},k,\sigma}$ be the set of all σ-frequent diamond episodes occurring in $\mathcal{S}$.

Definition 3. Frequent Diamond Episode Mining Problem: Given an input sequence $\mathcal{S}$, a window width $k \geq 1$, and a minimum frequency threshold $\sigma \geq 1$, the task is to find all σ-frequent diamond episodes $P \in \mathcal{F}_{\mathcal{S},k,\sigma}$ occurring in $\mathcal{S}$ with window width k without duplicates.

In the remainder of this paper, we design an algorithm for efficiently solving the frequent diamond episode mining problem in the sense of enumeration algorithms [2,4]. Let N be the total input size and M the number of all solutions. An enumeration algorithm $\mathcal{A}$ is of *output-polynomial time*, if $\mathcal{A}$ finds all solutions $S \in \mathcal{S}$ in total polynomial time both in N and M. Also $\mathcal{A}$ is of *polynomial delay*, if the *delay*, which is the maximum computation time between two consecutive outputs, is bounded by a polynomial in N alone. It is obvious that if $\mathcal{A}$ is of polynomial delay, then so is of output-polynomial.

3 A Polynomial-Delay and Polynomial-Space Algorithm

In this section, we present a polynomial-delay and polynomial-space algorithm PolyFreqDmd for mining all frequent diamond episodes in a given input sequence. Let $\mathcal{S} = (S_1, \ldots, S_n) \in (2^{\Sigma})^*$ be an input sequence of length n and total input size $N = ||\mathcal{S}||$, $k \geq 1$ be the window width, and $\sigma \geq 1$ be the minimum frequency threshold.

In Fig. 2, we show an outline of our polynomial-delay and polynomial-space algorithm PolyFreqDmd and its subprocedure FreqDmdRec for mining frequent diamond episodes of $\mathcal{DE}$ appearing in an input sequence $\mathcal{S}$.

The algorithm PolyFreqDmd is a backtrack algorithm that searches the whole search space from general to specific using depth-first search over the class $\mathcal{FDE}$ of frequent diamond episodes. For every pair of events $(a, b) \in \Sigma$, PolyFreqDmd starts the depth-first search by calling the recursive procedure FreqDmdRec with the smallest (complete) diamond episode $D_{ab} = (a \mapsto \emptyset \mapsto b) \in \mathcal{DE}$ and with its occurrence window list $\mathbf{W}(D_{ab})$ occurs.

```
algorithm PolyFreqDmd(S, k, Σ, σ)
input: input event sequence S ∈ (2^Σ)* of length n, window width k > 0,
alphabet of events Σ, the minimum frequency 1 ≤ σ ≤ n + k;
output: frequent diamond episodes;
{
1  Σ0 := the set of all events apparing no less than σ windows (Σ0 ⊆ Σ);
2  foreach ( a ∈ Σ0 ) do
3      output a;
4      foreach ( b ∈ Σ0 ) do
5          D0 := (a ↦ ∅ ↦ b); //2-serial episode
6          W0 := the occurrence window list W_{S,k}(D0) for D0;
7          FreqDmdRec(D0, W0, S, k, Σ0, σ);
8      end for
}
procedure FreqDmdRec(D = (a ↦ E ↦ b), W, S, k, Σ, σ)
output: finds all frequent diamond episodes of the form a ↦ E ↦ b;
{
1  if ( |W| ≥ σ ) then
2      output D; //(*) output D if the depth is odd (alternating output);
3      foreach ( e ∈ Σ (e > max(E)) ) ) do
4          C = a ↦ (E ∪ {e}) ↦ b;
5          U := UpdateDmdOcc(W, e, D, k, S);    //Computing U = W_{S,k}(C)
6          FreqDmdRec(C, U, S, k, Σ, σ);
7      end for
8      //(*) output D if the depth is even (alternating output);
9  end if
}
```

Fig. 2. The main algorithm PolyFreqDmd and a recursive subprocedure FreqDmdRec for mining frequent diamond episodes in a sequence

By Corollary 2, in each iteration of FreqDmdRec, the algorithm tests if the current candidate $D = (a \mapsto E \mapsto b)$ is frequent. If so, FreqDmdRec output D, and furthermore, for every event $e \in \Sigma$ such that $e > \max(E)$, FreqDmdRec grows diamond episode D by adding the new event e to the body E. Otherwise, FreqDmdRec prunes the search below D and backtrack to its parent. We call this process the *tail expansion* for diamond episodes. For episodes P, Q, if Q is generated from P by adding new event e as above, then we say that P is a *parent* of Q, or Q is a *child* of P.

Lemma 2. *For any the window width $k > 0$ and any minimum support σ, the algorithm* PolyFreqDmd *enumerates all and only frequent diamond episodes from $\mathcal{S}$ without duplicates.*

Proof. Suppose that $Q = (a \mapsto E \cup \{e\} \mapsto b) \in \mathcal{DE}$ is a child of some $P = (a \mapsto E \mapsto b) \in \mathcal{DE}$ obtained by the tail expansion such that $e > \max(E)$. From Corollary 2, we see that any frequent Q can be obtained by expanding some

algorithm UpdateDmdOcc($D, e, W, k, \mathcal{S}$)
input: a parent serial episode $D = (a \mapsto E \mapsto b)$, a new event $e > \max(E)$, the old occurrence list W for D, $k \leq 1$, an input sequence $\mathcal{S}$;
output: the new occurrence list U for the child $C = (a \mapsto E \cup \{e\} \mapsto b)$;
{
  $V :=$ FindSerialOcc($P = (a \mapsto e \mapsto b), k, \mathcal{S}$);
  return $U := W \cap V$;
}
procedure FindSerialOcc($(a \mapsto e \mapsto b), k, \mathcal{S}$)
{ **return** the occurrence list $\mathbf{W}_{\mathcal{S},k}(P)$ for P in $\mathcal{S}$; }

Fig. 3. The algorithm UpdateDmdOcc for incremental update of the occurrence list

frequent parent P. Furthermore, since $e > \max(E)$, the parent P is unique for each Q. This means that the parent-child relationship forms a spanning tree $\mathcal{T}$ for all frequent diamond episodes in $\mathcal{DE}$. Since FreqDmdRec makes the DFS on $\mathcal{T}$ by backtracking, the result immediately follows. □

In the recursive procedure FreqDmdRec in Fig. 2, the procedure newly create a child episode $C = (a \mapsto E \cup \{e\} \mapsto b)$ from the parent $D = (a \mapsto E \mapsto b)$ by tail expansion with $e \in \Sigma$ at Line 4. Then, at Line 5, it computes the new occurrence window list $U = W_{\mathcal{S},k}(C)$ for C in $\mathcal{S}$. To compute the new list U, we can use a native procedure that scans all k-windows in $\mathcal{S}$ one by one for checking occurrences of C.

Lemma 3. *There is an algorithm that computes the occurrence of a* 3*-serial episode* $P = a \mapsto b \mapsto c$ *in a given window* W *of width* k *in* $O(||W_i||) = O(|\Sigma|k)$ *time, where* $||W_i|| = \sum_{j=i}^{i+k-1} |S_j|$.

From Lemma 3, this naive algorithm requires $O(|\Sigma|kmn)$ time, where k is the window width, $m = ||D||$ is the episode size, and $n = |\mathcal{S}|$ is the input length.

In Fig. 3, we show an improved algorithm UpdateDmdOcc that computes the new occurrence list $U = W_{\mathcal{S},k}(C)$ in $O(|\Sigma|kn)$ time, by dropping the factor of $m = ||C||$, with incrementally updating the old list W for the parent D. To see the validity of the improved algorithm, we require two properties, called the *serial construction* for $\mathcal{DE}$ shown in Corollary 1 and the *downward closure property* for $\mathcal{DE}$ shown in Lemma 4 below. Here, Lemma 4 is an extension of the downward closure property for itemsets [1].

Lemma 4 (downward closure property). *Let* $a, b \in \Sigma$ *and* $E \subseteq \Sigma$. *Then, for any input sequence* $\mathcal{S}$ *and any* $k \geq 1$*, the following statement holds:*

$$\boldsymbol{W}_{\mathcal{S},k}(a \mapsto (E_1 \cup E_2) \mapsto b) = \boldsymbol{W}_{\mathcal{S},k}(a \mapsto E_1 \mapsto b) \bigcap \boldsymbol{W}_{\mathcal{S},k}(a \mapsto E_2 \mapsto b).$$

From Lemma 3 and Lemma 4, we see the correctness of the improved algorithm UpdateDmdOcc in Fig. 3, and have the next lemma. Note in the following that the computation time of UpdateDmdOcc does not depends on the size

```
procedure FastFindSerialOcc(P = (a↦e↦b), k, S = ⟨S_1, ... S_n⟩)
input: serial episode P = (a↦e↦b), window width k > 0, an input sequence S;
output: the occurrence list W for P;
{
    W := ∅; (x, y, z) := (0, 0, 0);
    for (i := −k + 1, ..., n ) do
        last := i − 1; end := i + k
        while x < end and (not (x > last and e ∈ S_x)) do x := x + 1;
        while y < end and (not (y > x and e ∈ S_y)) do y := y + 1;
        while z < end and (not (z > y and b ∈ S_z)) do z := z + 1;
        if ( last < x < y < z < end ) then W := W ∪ {i};
            //(x, y, z) is the lexicographically first occurrence of P in W_i;
    end for
    return W;
}
```

Fig. 4. An improved algorithm FirstFindSerialOcc for computing the occurrence list of a serial episode

$m = ||C||$ of the child episode. If we implement the procedure FindSerialOcc by an algorithm of Lemma 3, we have the next result.

Lemma 5. *The algorithm* UpdateDmdOcc *in Fig. 3, given the old list W for the parent diamond episode D and a newly added event e, computes the new occurrence list $U = W_{\mathcal{S},k}(C)$ for a new child C in $O(kN) = O(|\Sigma|kn)$ time, where $n = |\mathcal{S}|$ and $N = ||\mathcal{S}||$ are the length and the total size of input $\mathcal{S}$, respectively.*

Next, we present a faster algorithm for implementing the procedure FindSerialOcc for serial episodes than that of Lemma 3. In Fig. 4, we show the faster algorithm FastFindSerialOcc that computes $\mathbf{W}(P)$ for a 3-serial episode $P = a \mapsto e \mapsto b$ by a single scan of an input sequence $\mathcal{S}$ from left to right.

Lemma 6. *The algorithm* FastFindSerialOcc *in Fig. 4 computes the occurrence list of a 3-serial episode $P = a \mapsto b \mapsto c$ in an input sequence $\mathcal{S}$ of length n in $O(N) = O(|\Sigma|n)$ time regardless window width k, where $N = ||\mathcal{S}||$.*

Corollary 3. *Equipped with* FastFindSerialOcc *in Fig. 4, the modified algorithm* UpdateDmdOcc *computes $U = W_{\mathcal{S},k}(C)$ for a child $C \in \mathcal{DE}$ from the list $W = W_{\mathcal{S},k}(D)$ for the parent $D \in \mathcal{DE}$ and $e \in \Sigma$ in $O(N) = O(|\Sigma|n)$ time, where $n = |\mathcal{S}|$ and $N = ||\mathcal{S}||$.*

During the execution of the algorithm FreqDmdRec, the subprocedure FindSerialOcc (or FastFindSerialOcc) for updating occurrence lists are called many times with the same arguments $((a \mapsto e \mapsto b), k, \mathcal{S})$ $(e \in \Sigma)$. In the worst case, the number of calls may be $|\Sigma|$ times in the search paths. Therefore, we can achieve the reduction of the number of calls for FindSerialOcc by memorizing the results of the computation in a hash table $TABLE$.

```
1  global variable: a hash table TABLE : Σ → 2^{−k+1,...,n};
2  initialization: TABLE := ∅;
3  procedure LookupSerialOcc((a↦e↦b), k ∈ N, S)
{
4      if (TABLE[e] = UNDEF) then
5          V := FindSerialOcc((a↦e↦b), k, S);
6          if |V| ≥ σ then TABLE := TABLE ∪ {⟨ e, V⟩ };
7      end if;
8      return TABLE[e];
9  }
```

Fig. 5. Practical speed-up of FindSerialOcc using dynamic programming

In Fig. 5, we show the codes for practical speed-up method using dynamic programming. Then, we modify PolyFreqDmd in Fig. 2 and UpdateDmdOcc in Fig. 3 as follows:

- Before Line 5 of PolyFreqDmd, insert Line 2 (**initialization**) in Fig. 5.
- Replace the call of FindSerialOcc$((a \mapsto e \mapsto b), k, \mathcal{S})$ in FreqDmdRec by the call of LookupSerialOcc$((a \mapsto e \mapsto b), k, \mathcal{S})$ in Fig. 5.

This modification does not change the behavior of procedures PolyFreqDmd, FreqDmdRec and UpdateDmdOcc. Moreover, this makes the total number of the calls of FindSerialOcc to be bounded above by $|\Sigma|^3$, while it uses $O(|\Sigma|n)$ space in main memory. In Section 4 below, we know this technique will be useful in practice.

The running time of the algorithm FreqDmdRec in Fig. 2 mainly depends on the time $T(m, N)$ for the subprocedure UpdateDmdOcc at Line 5 to compute the occurrence list $U = \mathbf{W}_{\mathcal{S},k}(D)$ of a candidate $D \in \mathcal{DE}$ in $\mathcal{S}$, where $m = ||D||$ and $N = ||\mathcal{S}||$.

Unfortunately, if the height of the search tree is $d = \Theta(m) = \Theta(|\Sigma|)$, then the straightforward execution of the algorithm FastFindSerialOcc in Fig. 4 yields the delay of $O(d \cdot |\Sigma| \cdot T(n, N))$, where factor d comes from that it takes at least d recursive calls to come back to the root from the leaf of depth d. We can remove this factor $d = \Theta(m)$ by using a technique called *alternating output* in backtracking [10], which can be realized by replacing Line 2 and Line 8 in the algorithm FreqDmdRec with the corresponding lines (*) in the comments.

Theorem 1. *Let $\mathcal{S}$ be any input sequence of length n. For any window width $k \geq 1$ and minimum frequency threshold $\sigma \geq 1$, the algorithm* PolyFreqDmd *in Fig. 2 finds all σ-frequent diamond episodes D in $\mathcal{DE}$ occurring in $\mathcal{S}$ without duplicates in $O(|\Sigma|N) = O(|\Sigma|^2 n)$ delay (time per frequent episode) and $O(mn + N) = O(|\Sigma|n)$ space, where $N = ||\mathcal{S}||$ and $m = ||D||$ is the maximum size of frequent episodes.*

Corollary 4. *The frequent diamond episode mining problem is solvable in linear delay w.r.t. the total input size and in polynomial space.*

```
1  C = a↦(E ∪ {e})↦b;
2  Δ := FindSerialOcc(P = (a↦e↦b), k, S);
3  W := W − Δ;
4  FreqDmdRec(C, U, S, k, Σ, σ);
5  W := W ∪ Δ;
6  C = a↦(E − {e})↦b;
```

Fig. 6. The diffset technique in PolyFreqDmd

Finally, we can reduce the space complexity of the algorithm PolyFreqDmd by using the *diffset* technique introduced by Zaki [11] for itemset mining, which can be realized by replacing Line 5 and Line 6 of PolyFreqDmd with the code in Fig. 6. Hence, we can reduce the space complexity in Theorem 1 to $O(m+n) = O(|\Sigma| + n)$.

4 Experimental Results

In this section, we give the experimental results for the following combinations of the algorithms given in Section 3, by applying to the randomly generated event sequences $\mathcal{S} = (S_1, \ldots, S_n)$ over an alphabet $\Sigma = \{1, \ldots, s\}$, where each event set S_i $(i = 1, \ldots, n)$ is generated by a uniform distribution with letter probability $0 \leq p \leq 1/|\Sigma|$ and stopping probability $1 - p$.

DF : PolyFreqDmd (Fig. 2) with FindSerialOcc (Fig. 3).
DF-SWO : DF with alternative output (SWO) (Fig. 2 with (*)).
DF-FFS : DF with fast update by FastFindSerialOcc (FFS) (Fig. 4).
DF-DIFF : DF with diffset technique (DIFF) (Fig. 6).
DF-DP : DF-FFS with dynamic programming (DP) (Fig. 5).

All experiments were run in a PC (AMD Mobile Athlon64 Processor 3000+, 1.81GHz, 2.00GB memory) with 32-bit x86 instruction set. Without saying explicitly, we assume that the length of the sequence is $n = |\mathcal{S}| = 2000$, the alphabet size is $s = |\mathcal{S}| = 30$, the probability of each event is $p = 0.1$, the window width is $k = 10$, the minimum frequency threshold is $\sigma = 0.4$.

Fig. 7 shows the running time and the number of solutions of the algorithms DF, DF-FFS and DF-DP for the input length n, where $s = 20$, $k = 10$ and $\sigma = 0.1n$. Then, we know that DF-FFS is twice as faster as DF and DF-DP is one hundred times as faster as DF. On the other hand, we cannot find any difference between DF-SWO, DF-DIFF and DF on this data set, although the first two techniques are useful in technical improvements. Moreover, the running time of these algorithms seems to be linear in the input size and thus expected to scales well on large datasets.

Fig. 8 shows the running time for the number of outputs, where $n = 10,000$, $k = 30$ and $\sigma = 0.3n$. Then, we see that the slope is almost constant and thus the delay is just determined by the input size as indicated by Theorem 1.

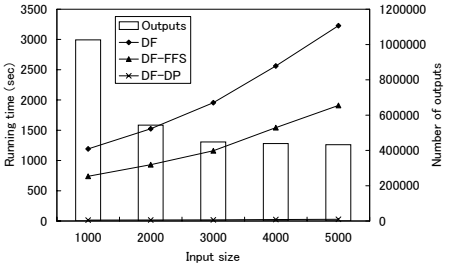


Fig. 7. Running time for the input length n, where $s = 20$, $k = 10$ and $\sigma = 0.1n$

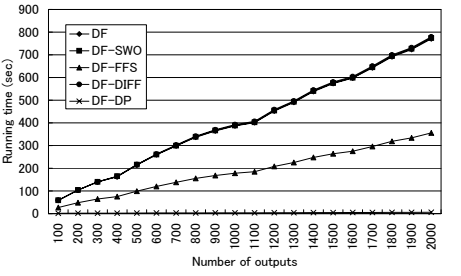


Fig. 8. Running time for the number of outputs, where $n = 10,000$, $k = 30$ and $\sigma = 0.3n$

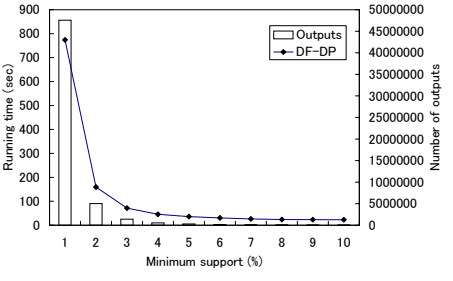


Fig. 9. Running time for the minimum support $0.5n \leq \sigma \leq 5n$ with span $0.5n$, where $n = 2,000$ and $k = 10$

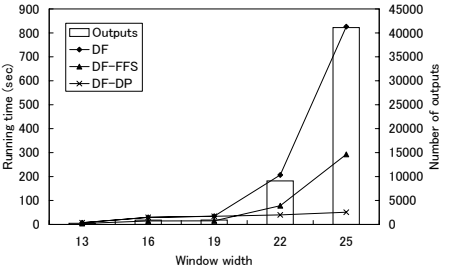


Fig. 10. Running time for the window width $13 \leq k \leq 25$, where $n = 2,000$ and $\sigma = 0.4n$

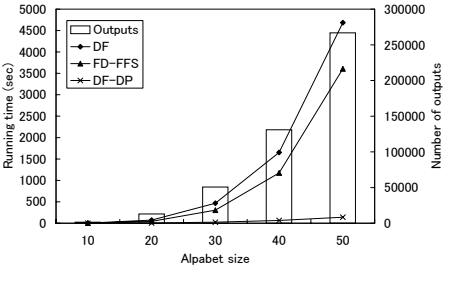


Fig. 11. Running time for the alphabet size $10 \leq |\Sigma| \leq 50$ with span 10, where $n = 2,000$, $\sigma = 0.4n$

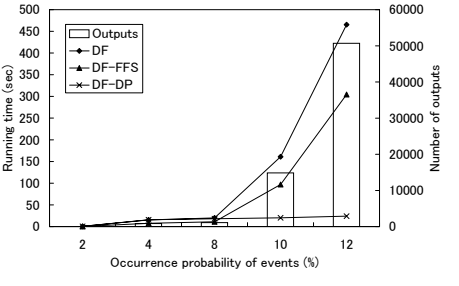


Fig. 12. Running time for the occurrence probability of events $0.02 \leq p \leq 0.12$ with span 0.02, where $n = 2,000$ and $\sigma = 0.4n$

Fig. 9 shows the running time of DF-DP, our fastest algorithm, with varying the minimum support $0.5n \leq \sigma \leq 5.0n$ with the input size $n = 2000$. We see that the number of outputs, and thus, the running time increase as σ decreases.

Figs. 10, 11 and 12 show the running time of the algorithms DF, DF-FFS and DF-DP with varying the window width $13 \leq k \leq 25$, the size of alphabet

$10 \leq |\Sigma| \leq 50$ and the event probability $0.02 \leq p \leq 0.12$, respectively. Then, we see that DF-DP outperforms other algorithms in most cases. The performance of DF-DP is stable in most datasets and the parameter settings. We also see that DF-FFS is from 20% to 60% faster than DF.

Overall, we conclude that the proposed algorithm FindDmdMain with the practical speed-up technique by dynamic programming in Fig. 5 (DF-DP) is quite efficient on the data sets used these experiments. The fast linear-time update by FastFindSerialOcc (DF-FFS) achieves twice speed-up.

5 Conclusion

This paper studied the problem of frequent diamond episode mining, and presented an efficient algorithm PolyFreqDmd that finds all frequent diamond episodes in an input sequence in polynomial delay and polynomial space in the input size. We have further studied several techniques for reducing the time and the space complexities of the algorithm.

Possible future problems are extension of PolyFreqDmd for general fragments of DAGs [7,8], and efficient mining of closed patterns [2,3,8,11] for diamond episodes and their generalizations. Also, we plan to apply the proposed algorithm to bacterial culture data [5,6].

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proc. 20th VLDB, pp. 487–499 (1994)
2. Arimura, H.: Efficient algorithms for mining frequent and closed patterns from semi-structured data. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS, vol. 5012, pp. 2–13. Springer, Heidelberg (2008)
3. Arimura, H., Uno, T.: A polynomial space and polynomial delay algorithm for enumeration of maximal motifs in a sequence. In: Deng, X., Du, D.-Z. (eds.) ISAAC 2005. LNCS, vol. 3827, pp. 724–737. Springer, Heidelberg (2005)
4. Avis, D., Fukuda, K.: Reverse search for enumeration. Discrete Applied Mathematics 65, 21–46 (1996)
5. Katoh, T., Hirata, K.: Mining frequent elliptic episodes from event sequences. In: Proc. 5th LLLL, pp. 46–52 (2007)
6. Katoh, T., Hirata, K., Harao, M.: Mining frequent diamond episodes from event sequences. In: Torra, V., Narukawa, Y., Yoshida, Y. (eds.) MDAI 2007. LNCS (LNAI), vol. 4617, pp. 477–488. Springer, Heidelberg (2007)
7. Mannila, H., Toivonen, H., Verkamo, A.I.: Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery 1, 259–289 (1997)
8. Pei, J., Wang, H., Liu, J., Wang, K., Wang, J., Yu, P.S.: Discovering frequent closed partial orders from strings. IEEE TKDE 18, 1467–1481 (2006)
9. Pei, J., Han, J., Mortazavi-Asi, B., Wang, J., Pinto, H., Chen, Q., Dayal, U., Hsu, M.-C.: Mining sequential patterns by pattern-growth: The PrefixSpan approach. IEEE Trans. Knowledge and Data Engineering 16, 1–17 (2004)
10. Uno, T.: Two general methods to reduce delay and change of enumeration algorithms, NII Technical Report, NII-2003-004E (April 2003)
11. Zaki, M.J., Hsiao, C.-J.: CHARM: An efficient algorithm for closed itemset mining. In: Proc. 2nd SDM, pp. 457–478. SIAM, Philadelphia (2002)

A Statistical Approach for Binary Vectors Modeling and Clustering

Nizar Bouguila[1] and Khalid Daoudi[2]

[1]Concordia Institute for Information Systems Engineering, Concordia University, Montreal, Canada, Qc, H3G 2W1
bouguila@ciise.concordia.ca
[2] CNRS-IRIT, Université Paul Sabatier, F-31062, Toulouse, France
daoudi@irit.fr

Abstract. This paper presents an approach for Binary feature selection. Our selection technique is based on a principled statistical model using a finite mixture of distributions. In contrast with classic feature selection algorithms that have been proposed in supervised settings, where training data are available and completely labeled, our approach is fully unsupervised. Through some applications, we found that our feature selection model improves the clustering results.

1 Introduction

Learning techniques and in particular statistical methods are widely used in data mining applications. One of the fundamental problems when using statistical approaches, is the quality of the data to model. The data may contain a large amount of irrelevant or low-quality information. Indeed, if the data are too few comparing to its dimensionality or the measurement dimensions are too noisy or irrelevant, then a statistical model may not be accurate and may suffer from several problems such as overfitting [1,2] and its performance may degrade. Feature selection is a well-studied approach to resolve this problem and its goals are improving computational efficiency and clustering effectiveness.

Feature selection, as a classical problem, has been the subject of intensive research efforts for the past several decades and different approaches have been proposed [3,4,5]. Feature selection approaches, however, have been devoted especially to Gaussian data, and little work has been done in the case of non-Gaussian [6,7,8], and particularly binary [9], data. Indeed, the literature is full of the travail associated with supervised feature selection techniques which can be categorized as filter-based, wrapper-based and hybrid approaches [3]. Filter approaches filter out irrelevant features using an algorithm different from the learning algorithm and wrapper models use the learning algorithm itself for the selection [4]. Some cognitive approaches have been proposed (See [10], for instance), also.

In this work, we are interested in a difficult problem which is unsupervised binary feature selection. Indeed, in many cases, neither the class labels nor the number of clusters are known and the problem of feature selection become more challenging, since the selections of the features and the number of clusters have

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 184–195, 2009.
© Springer-Verlag Berlin Heidelberg 2009

to be performed in conjunction [11,12]. To explore the feature selection problem for unlabeled binary data vectors, we propose a theoretically sound model-based approach, using a finite multivariate Bernoulli mixture model, which supports reasoning and generalization in the presence of varying context information. The learning of this model is based on maximum a posteriori (MAP) approach. Besides, we give a closed form expression to approximate its stochastic complexity which we use for model selection.

The rest of the paper is organized as follows. Section 2 presents our clustering and feature selection model using finite multivariate Bernoulli mixture. Section 3 gives the parameters estimation approach and the stochastic complexity principle for the selection of the number of clusters. We show our experimental results in Section 4. Finally, Section 5 ends the paper with some concluding remarks.

2 The Statistical Feature Weighting and Selection Model

Let $\mathcal{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N\}$ denotes a data set composed of binary vectors representing N images (or documents). Each vector $\boldsymbol{X}_n, n = 1, \ldots, N$ is D-dimensional and of the form $(X_{n1}, \ldots, X_{nD})$, where each variable $X_{nd} \in \{0, 1\}, d = 1, \ldots, D$, and describes the presence or absence of a given feature d in the image (or document). By supposing that $\mathcal{X}$ is composed of M clusters, each vector $\boldsymbol{X}_n$ can be modeled as a finite mixture of multivariate Bernoulli distributions:

$$p(\boldsymbol{X}_n|\Theta_M) = \sum_{j=1}^{M} p_j p(\boldsymbol{X}_n|\boldsymbol{\pi}_j) = \sum_{j=1}^{M} p_j \prod_{d=1}^{D} \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1-X_{nd}} \tag{1}$$

where $\Theta_M = \{\{\boldsymbol{\pi}_j\}, \boldsymbol{P}\}$ is the set of parameters defining the mixture model, $\boldsymbol{\pi}_j = (\pi_{j1}, \ldots, \pi_{jD})$ and $\boldsymbol{P} = (p_1, \ldots, p_M)$ is the mixing parameters vector, $0 \leq p_j \leq 1$, $\sum_{j=1}^{M} p_j = 1$.

As we have mentioned in the previous section, often in the case of high-dimensional data, many features are irrelevant and can deteriorate the clustering performance. Thus, these irrelevant features should be removed or associated with small weights. A common used approach is to suppose that a given feature is irrelevant if its distribution follows a common density across clusters and then is independent of the class labels [12,13]. Using this definition our model can be written as follows

$$p(\boldsymbol{X}_n|\Theta_M, \Phi, \boldsymbol{\Lambda}) = \sum_{j=1}^{M} p_j \prod_{d=1}^{D} \left[\left(\pi_{jd}^{X_{nd}} (1-\pi_{jd})^{1-X_{nd}} \right)^{\phi_d} \left(\lambda_d^{X_{nd}} (1-\lambda_d)^{1-X_{nd}} \right)^{1-\phi_d} \right] \tag{2}$$

where $\Phi = (\phi_1, \ldots, \phi_D)$ is a set of binary parameters and $\phi_d = 1$ is feature d is relevant and 0, otherwise, $\boldsymbol{\Lambda} = (\lambda_1, \ldots, \lambda_D)$ are the parameters of a multivariate Bernoulli distribution considered as a common background model to explain non-salient features. By defining the feature saliency as the probability that feature d is relevant $\rho_d = p(\phi_d = 1)$, straightforward manipulations of Eq. 2 give us

$$p(\boldsymbol{X}_n|\Theta) = \sum_{j=1}^{M} p_j \prod_{d=1}^{D} \left[\rho_d \pi_{jd}^{X_{nd}} (1-\pi_{jd})^{1-X_{nd}} + (1-\rho_d) \lambda_d^{X_{nd}} (1-\lambda_d)^{1-X_{nd}} \right] \quad (3)$$

where $\Theta = \{\Theta_M, \{\rho_d\}, \boldsymbol{\Lambda}\}$. Notes that the previous model takes into account the fact that different features may have different effects on clustering (i.e creating clusters) and that unimportant irrelevant feature d with low probability ρ_d may be removed. Indeed, the added parameters can be viewed as additional degrees of freedom for the model to avoid overfitting the data.

3 Unsupervised Model Learning

3.1 Parameters Estimation

An important step when using finite mixtures for modeling is the estimation of the associated parameters. A standard procedure is to estimate the parameters by maximizing the likelihood function:

$$p(\mathcal{X}|\Theta) = \prod_{n=1}^{N} \sum_{j=1}^{M} p_j \prod_{d=1}^{D} \left[\rho_d \pi_{jd}^{X_{nd}} (1-\pi_{jd})^{1-X_{nd}} + (1-\rho_d) \lambda_d^{X_{nd}} (1-\lambda_d)^{1-X_{nd}} \right] \quad (4)$$

The main approaches for parameters estimation are the maximum likelihood (ML) and the maximum a posteriori (MAP) [1]. In the case of discrete data, and binary data in particular, it is generally better to apply smoothing approaches by assuming priors over the parameters which prevent overfitting especially in the presence of small number of observations [16]. Thus, we have used the MAP approach given as the following

$$\hat{\Theta} = \arg\max_{\Theta} \{\log p(\mathcal{X}|\Theta) + \log p(\Theta)\} \quad (5)$$

where $p(\Theta)$ is the prior distribution and is taken as the product of the priors of the different model's parameters.

We know that $\boldsymbol{P}$ is defined on the following simplex $\{(p_1, \ldots, p_M) : \sum_{j=1}^{M} p_j = 1\}$. Thus, a natural choice, as a prior, for $\boldsymbol{P}$ is the Dirichlet distribution which is closed under multinomial sampling (i.e the Dirichlet is a conjugate prior for the multinomial distribution):

$$p(\boldsymbol{P}) = \frac{\Gamma(\sum_{j=1}^{M} \eta_j)}{\prod_{j=1}^{M} \Gamma(\eta_j)} \prod_{j=1}^{M} p_j^{\eta_j - 1} \quad (6)$$

Besides, we take Beta distributions as priors for π_{jd}, λ_d and ρ_d, since these parameters are defined on the compact support [0,1], then

$$p(\pi_{jd}) = \frac{\Gamma(\sigma_{jd} + \omega_{jd})}{\Gamma(\sigma_{jd})\Gamma(\omega_{jd})} \pi_{jd}^{\sigma_{jd}-1} (1-\pi_{jd})^{\omega_{jd}-1} \quad (7)$$

[1] Another estimation approach, for binary data, based on classification maximum likelihood has been proposed and discussed in [14,15].

$$p(\lambda_d) = \frac{\Gamma(\mu_d + \nu_d)}{\Gamma(\mu_d)\Gamma(\nu_d)} \lambda_d^{\mu_d - 1}(1 - \lambda_d)^{\nu_d - 1} \tag{8}$$

$$p(\rho_d) = \frac{\Gamma(\alpha_d + \beta_d)}{\Gamma(\alpha_d)\Gamma(\beta_d)} \rho_d^{\alpha_d - 1}(1 - \rho_d)^{\beta_d - 1} \tag{9}$$

where $\{\{\eta_j\}, \{\sigma_{jd}\}, \{\omega_{jd}\}, \{\alpha_d\}, \{\beta_d\}\, \{\mu_d\}, \{\nu_d\}\}$ is the set of hyperparameters, which are all positive, and can be viewed as a set of confidence measures of our prior probabilities. Then,

$$p(\Theta) = \frac{\Gamma(\sum_{j=1}^{M} \eta_j)}{\prod_{j=1}^{M} \Gamma(\eta_j)} \prod_{j=1}^{M} p_j^{\eta_j - 1} \times \prod_{j=1}^{M} \prod_{d=1}^{D} \frac{\Gamma(\sigma_{jd} + \omega_{jd})}{\Gamma(\sigma_{jd})\Gamma(\omega_{jd})} \pi_{jd}^{\sigma_{jd} - 1}(1 - \pi_{jd})^{\omega_{jd} - 1}$$
$$\times \prod_{d=1}^{D} \frac{\Gamma(\mu_d + \nu_d)}{\Gamma(\mu_d)\Gamma(\nu_d)} \frac{\Gamma(\alpha_d + \beta_d)}{\Gamma(\alpha_d)\Gamma(\beta_d)} \lambda_d^{\mu_d - 1}(1 - \lambda_d)^{\nu_d - 1} \rho_d^{\alpha_d - 1}(1 - \rho_d)^{\beta_d - 1} \tag{10}$$

By maximizing $\log p(\mathcal{X}|\Theta) + \log p(\Theta)$ (Eq. 5), we obtain the following

$$p_j = \frac{\sum_{n=1}^{N} p(j|\boldsymbol{X}_n) + (\eta_j - 1)}{N + M(\eta_j - 1)} \tag{11}$$

$$\pi_{jd} = \frac{\sum_{n=1}^{N} p(j|\boldsymbol{X}_n) \frac{\rho_d X_{nd} \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1 - X_{nd}}}{\rho_d \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1 - X_{nd}} + (1 - \rho_d) \lambda_d^{X_{nd}} (1 - \lambda_d)^{1 - X_{nd}}} + \sigma_{jd} - 1}{\sum_{n=1}^{N} p(j|\boldsymbol{X}_n) \frac{\rho_d \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1 - X_{nd}}}{\rho_d \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1 - X_{nd}} + (1 - \rho_d) \lambda_d^{X_{nd}} (1 - \lambda_d)^{1 - X_{nd}}} + \sigma_{jd} + \omega_{jd} - 2} \tag{12}$$

$$\lambda_d = \frac{\sum_{n=1}^{N} \sum_{j=1}^{M} p(j|\boldsymbol{X}_n) \frac{(1 - \rho_d) X_{nd} \lambda_d^{X_{nd}} (1 - \lambda_d)^{1 - X_{nd}}}{\rho_d \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1 - X_{nd}} + (1 - \rho_d) \lambda_d^{X_{nd}} (1 - \lambda_d)^{1 - X_{nd}}} + \mu_d - 1}{\sum_{n=1}^{N} \sum_{j=1}^{M} p(j|\boldsymbol{X}_n) \frac{(1 - \rho_d) \lambda_d^{X_{nd}} (1 - \lambda_d)^{1 - X_{nd}}}{\rho_d \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1 - X_{nd}} + (1 - \rho_d) \lambda_d^{X_{nd}} (1 - \lambda_d)^{1 - X_{nd}}} + \mu_d + \nu_d - 2} \tag{13}$$

$$\rho_d = \frac{\sum_{n=1}^{N} \sum_{j=1}^{M} p(j|\boldsymbol{X}_n) \frac{\rho_d \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1 - X_{nd}}}{\rho_d \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1 - X_{nd}} + (1 - \rho_d) \lambda_d^{X_{nd}} (1 - \lambda_d)^{1 - X_{nd}}} + \alpha_d - 1}{N + \alpha_d + \beta_d - 2} \tag{14}$$

where

$$p(j|\boldsymbol{X}_n) = \frac{p_j \prod_{d=1}^{D} \rho_d \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1 - X_{nd}} + (1 - \rho_d) \lambda_d^{X_{nd}} (1 - \lambda_d)^{1 - X_{nd}}}{\sum_{j=1}^{M} \left(p_j \prod_{d=1}^{D} \rho_d \pi_{jd}^{X_{nd}} (1 - \pi_{jd})^{1 - X_{nd}} + (1 - \rho_d) \lambda_d^{X_{nd}} (1 - \lambda_d)^{1 - X_{nd}} \right)} \tag{15}$$

and represents the *posterior* probability (i.e the conditional probability that $\boldsymbol{X}_n$ belongs to class j). Note that the previous equations are reduced to the ML estimates when all hyperparameters are set to one.

3.2 Model Selection

Different criteria have been proposed to determine how many clusters should be considered in the modeling process such as Akaike's information criterion [17], minimum description length [18], and minimum message length [19]. The majority of these criteria are based upon likelihood and their principal difference is in their penalty term [20,21]. In our case, we will use the stochastic complexity principle, for the selection of the model order M, which has been shown to be effective in the case of discrete and binary data modeling [22]. Stochastic complexity has been widely studied by Rissanen in a series of papers (See [23], for instance) and can be viewed as the shortest description length of the data and then allows to estimate the optimal statistical model to encode it. The stochastic complexity is defined as minus logarithm of the integrated (or marginal) likelihood given by

$$p(\mathcal{X}|M) = \int \pi(\Theta|\mathcal{X}, M)d\Theta = \int p(\mathcal{X}|\Theta, M)\pi(\Theta)d\Theta \tag{16}$$

$p(\mathcal{X}|M)$ is called also the evidence and can be viewed as an information theory measure. Computing the integrated likelihood is analytically intractable in practice and different approximations have been proposed [24]. A well known approximation to stochastic complexity is the Bayesian Information Criterion (BIC) [25], which is equivalent to the first version of the minimum description length (MDL) proposed by Rissanen [18]. A better approach that we adopt in this paper is the Cheeseman-Stutz approximation used in the AutoClass system [26] which suggests the use of the complete data evidence. The complete data for our model in Eq. 3 is

$$p(\mathcal{X}, \mathcal{Z}, \Phi|\Theta) = \prod_{n=1}^{N}\prod_{j=1}^{M}\left[p_j \prod_{d=1}^{D}\left(\rho_d \pi_{jd}^{X_{nd}}(1-\pi_{jd})^{1-X_{nd}}\right)^{\phi_d}\left((1-\rho_d)\lambda_d^{X_{nd}}(1-\lambda_d)^{1-X_{nd}}\right)^{1-\phi_d}\right]^{Z_{nj}} \tag{17}$$

where $\mathcal{Z} = (\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_N)$, $\boldsymbol{Z}_n = (Z_{n1}, \ldots, Z_{nM})$ with $Z_{nj} = 1$ if $\boldsymbol{X}_n$ belongs to class j and 0 otherwise. Thus, the complete data evidence is given by

$$\begin{aligned}
&p(\mathcal{X}, \mathcal{Z}, \Phi) = \int p(\mathcal{X}, \mathcal{Z}, \Phi|\Theta)\pi(\Theta)d\Theta \\
&= \frac{\Gamma(\sum_{j=1}^{M}\eta_j)}{\prod_{j=1}^{M}\Gamma(\eta_j)}\frac{\prod_{j=1}^{M}\Gamma(\eta_j + n_j)}{\Gamma(N + \sum_{j=1}^{M}\eta_j)} \\
&\times \prod_{j=1}^{M}\prod_{d=1}^{D}\left[\frac{\Gamma(\sigma_{jd}+\omega_{jd})}{\Gamma(\sigma_{jd})\Gamma(\omega_{jd})}\frac{\Gamma(\sigma_{jd} + \phi_d\sum_{i=1}^{N}Z_{nj}X_{nd})\Gamma(\omega_{jd} + \phi_d\sum_{i=1}^{N}Z_{nj}(1-X_{nd}))}{\Gamma(\sigma_{jd}+\omega_{jd}+\phi_d n_j)}\right] \\
&\times \prod_{d=1}^{D}\frac{\Gamma(\mu_d+\nu_d)}{\Gamma(\mu_d)\Gamma(\nu_d)}\frac{\Gamma(\mu_d + (1-\phi_d)\sum_{n=1}^{N}X_{nd})\Gamma(\nu_d + (1-\phi_d)\sum_{i=1}^{N}(1-X_{nd}))}{\Gamma(\mu_d+\nu_d+N)} \\
&\times \prod_{d=1}^{D}\frac{\Gamma(\alpha_d+\beta_d)}{\Gamma(\alpha_d)\Gamma(\beta_d)}\frac{\Gamma(\alpha_d + N\phi_d)\Gamma(\beta_d + (1-\phi_d)N)}{\Gamma(N+\alpha_d+\beta_d)}
\end{aligned} \tag{18}$$

where n_j is the number of vectors in cluster j.

Having Eq. 18 in hand, our complete algorithm can be summarized as follows:

Algorithm For each candidate value of M:

1. Set $\rho_t \leftarrow 0.5$, $t = 1, \ldots, T$ and initialization using k-Means algorithm.
2. Iterate the two following steps until convergence:
   (a) E-Step: Update $p(j|\boldsymbol{X}_n)$ using Eq. 15
   (b) M-Step: Update the p_j, π_{jt}, λ_t and ρ_t using Eqs. 11, 12, 13 and 14, respectively.
   (c) If $\rho_t \sim 1$ then prune λ_t. If $\rho_t \sim 0$, then prune π_{jt}, $j = 1, \ldots, M$.
3. Calculate the associated stochastic complexity ($-\log p(\mathcal{X}, \mathcal{Z}, \Phi)$) using Eq. 18.
4. Select the optimal model that yields the smallest stochastic complexity.

4 Experimental Results

4.1 Real Data

In the first experiment, we evaluate the performance of our selection model on the zoo data set available from the UCI Machine Learning Repository [27, the data set contributed by Richard Forsyth]. The data set describes 101 instances of animals in terms of 17 attributes ("animal name", 15 boolean attributes, and one numeric attribute called "legs" and "type" which indicates the class membership of the instance). Table 1 shows an example input vector for the instance "elephant". One animal, "frog" appears twice, so we eliminate one of them. The zoo data set contains 7 classes (each class represents a given type

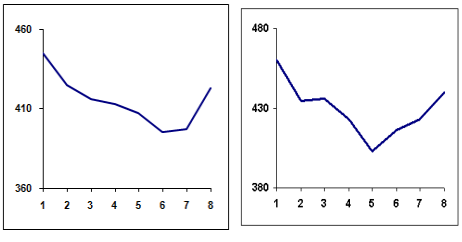
Table 1. The input vector for the elephant instance in Zoo data set

Label	Value
Hair	1
Feathers	0
Eggs	0
Milk	1
Airborne	0
Aquatic	0
Predator	0
Toothed	1
Backbone	1
Breathes	1
Venomous	0
Fins	0
Legs	4
Tail	1
Domestic	0
Catsize	1
Type	1

of animals) and the numbers of animals in each class are: 41, 20, 5, 13, 3, 8, 10, respectively. Besides, following [28], we eliminate the two attributes "animal name" and "type" and translate the numeric attribute, "legs", into six binary features corresponding to 0, 2, 4, 5, 6, and 8 legs, respectively.

Figure 1 shows the number of clusters found by our algorithm with and without feature selection. According to this figure, we obtain a better result using feature selection (6 clusters). In both cases, however, our algorithm fails to select the exact number of clusters which is 7 and does not identify the cluster representing type 5 because of its small number of instances (only 3 animals). This figure shows also that without feature selection, the algorithm selects only 5 clusters because it does not succeed to identify cluster representing type 3 (with only 5 animals). Tables 2 and 3 represent the confusion matrices with and without feature selection, respectively. According to these tables, it is clear that the feature selection process improves clustering. The resulted saliencies of the different features, with their standard deviation over 20 runs are shown in figure 2.

We also evaluated our model on the advertisement data set, available from the UCI machine learning repository [27]. The data set represents 3279 images and was collected to determine if images are advertisement or not according to a large number of their surrounding features. In order to represent each image in this data set as a binary vector, we removed the continuous features (3 features) and the missing value features (1 feature). After this preprocessing, each image



(a) (b)

Fig. 1. Number of clusters found for the zoo data set. (a) with feature selection, (b) without feature selection.

Table 2. Confusion matrix of the zoo data without feature selection

Class	Type1	Type2	Type3	Type4	Type5	Type6	Type7
Type1	41	0	0	0	0	0	0
Type2	0	20	0	0	0	0	0
Type3	1	0	0	1	0	2	1
Type4	0	0	0	13	0	0	0
Type5	0	0	0	1	0	1	1
Type6	0	0	0	0	0	8	0
Type7	0	0	0	1	0	1	8

Table 3. Confusion matrix of the zoo data with feature selection

Class	Type1	Type2	Type3	Type4	Type5	Type6	Type7
Type1	41	0	0	0	0	0	0
Type2	0	20	0	0	0	0	0
Type3	0	0	5	0	0	0	0
Type4	0	0	0	13	0	0	0
Type5	0	0	3	0	0	0	0
Type6	0	0	0	0	0	8	0
Type7	0	0	0	0	0	0	10

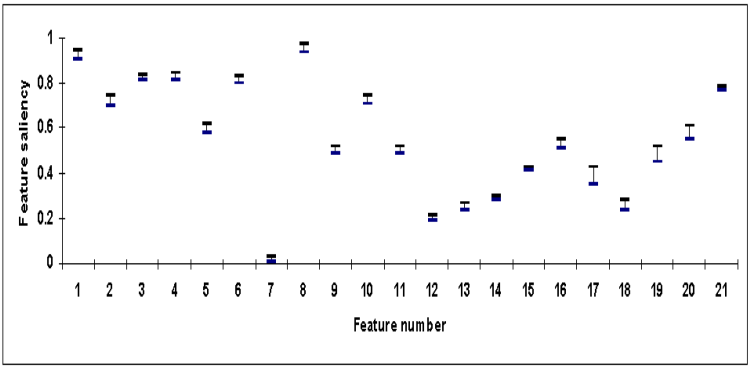


Fig. 2. Feature saliencies for the zoo data set

was represented by 1555-dimensional binary vector. We split the advertisement data set into a training set of 1779 vectors and a test set of 1500 vectors. We run our algorithm 20 times and the average error rate without feature selection was 10.33 ± 2.68 as opposed to 7.23 ± 2.03 obtained when we apply feature selection.

4.2 Binary Images Classification: Handwritten Digit Recognition

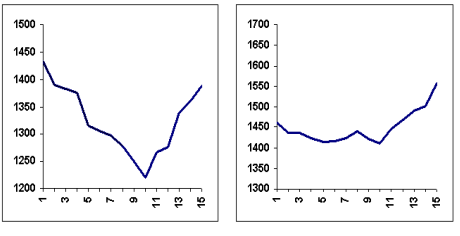
The ever-increasing quantity of digital data and images in particular presents a lot of opportunities to extract useful information. At the same time this growing content presents a lot of challenges [29]. Indeed, an important problem in computer vision is to obtain efficient summaries of image databases. In this application we use binary image classification to handle the handwriting digit recognition problem which has many application scenarios such as auto mail classification according to the zip code and signature dates recognition. Results were obtained on the well-known UCI database [27] which contains 5620 objects. The repartition of the different classes is given in table 4. The original images are processed to extract normalized bitmaps of handwritten digits. Each normalized bitmap includes a 32×32 matrix (each image is represented then by

Table 4. Repartition of the different classes

class	Number of objects
0	554
1	571
2	557
3	572
4	568
5	558
6	558
7	566
8	554
9	562



Fig. 3. Example of normalized bitmaps



(a) (b)

Fig. 4. Number of clusters found for the handwriting digit recognition problem. (a) with feature selection, (b) without feature selection.

1024-dimensional binary vector) in which each element indicates one pixel with value of white or black. Figure 3 shows an example of the normalized bitmaps.

In this application, the classification is done directly (i.e. we do not separate the data set into training and test sets). Note that all the results that we present here are obtained over 20 runs of our algorithm and are rounded. Figure 4 shows the number of clusters selected by our algorithm for the binary images data set. According to this figure it is clear that our algorithm finds the exact number of clusters (10) with and without feature selection. However, the feature selection process appear to improve the discrimination between models. Table 5 gives the confusion matrix when we use our feature selection model. The number of images misclassified was 287 (an error of 5.10 percent) as compared to 9.30 percent without feature selection (the number of misclassified images was 523, See table 6).

Table 5. Confusion matrix for handwritten digit recognition application using feature selection

	Class0	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8	Class9
Class0	517	0	0	0	17	0	9	0	11	0
Class1	0	543	8	0	12	0	0	8	0	0
Class2	0	11	526	0	5	0	0	0	0	15
Class3	0	0	0	543	13	0	16	0	0	0
Class4	0	1	0	0	550	0	0	0	0	17
Class5	0	0	7	0	0	543	0	0	0	8
Class6	11	0	0	0	0	0	532	0	15	0
Class7	9	5	0	6	2	0	0	539	0	5
Class8	5	0	11	0	12	0	7	0	519	0
Class9	11	5	0	0	8	9	0	0	8	521

Table 6. Confusion matrix for handwritten digit recognition application without feature selection

	Class0	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8	Class9
Class0	481	5	1	4	21	2	19	3	11	7
Class1	2	518	11	4	17	1	1	13	1	3
Class2	2	16	507	2	8	1	1	1	1	18
Class3	1	2	2	522	19	4	19	0	0	3
Class4	2	6	0	4	532	1	1	0	2	20
Class5	4	2	11	0	0	519	0	2	3	17
Class6	14	3	4	2	2	1	499	2	26	5
Class7	13	8	2	10	5	3	1	515	0	9
Class8	9	2	13	1	17	1	9	0	500	2
Class9	11	8	2	1	10	13	1	1	11	504

5 Conclusion

In this paper we have considered the problem of unsupervised learning of binary data vectors in the presence of irrelevant features. Our work is motivated by the need and the desire to handle automatically the clustering and selection problems for this kind of data which are widely present in many important applications. We have proposed an algorithm that iterates simultaneously between clustering and feature selection until finding a compact meaningful statistical representation of the data. We have also given a closed form expression to approximate the stochastic complexity of our model. Through some experimental results, we have shown that our binary selection model has good modeling capabilities.

Acknowledgment

The completion of this research was made possible thanks to the Natural Sciences and Engineering Research Council of Canada (NSERC), and a NATEQ Nouveaux Chercheurs Grant.

References

1. Raudys, S.J., Jain, A.K.: Small Sample Size Effects in Statistical Pattern Recognition: Recommendations for Practitioners. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(3), 252–264 (1991)
2. Kohavi, R., Sommerfield, D.: Feature Subset Selection Using the Wrapper Method: Overfitting and Dynamic Search Space Topology. In: Proc. of the First International Conference on Knowledge Discovery and Data Mining (KDD), pp. 192–197 (1995)
3. Liu, H., Yu, L.: Toward Integrating Feature Selection Algorithms for Classification and Selection. IEEE Transactions on Knowledge and Data Engineering 17(4), 491–502 (2005)
4. Blum, A.L., Langley, P.: Selection of Relevant Features and Examples in Machine Learning. Artificial Intelligence 97, 245–271 (1997)
5. Jain, A.K., Zongker, D.: Feature Selection: Evaluation, Application, and Small Sample Performance. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(2), 153–158 (1997)
6. Boutemedjet, S., Bouguila, N., Ziou, D.: A Hybrid Feature Extraction Selection Approach for High-Dimensional Non-Gaussian Data Clustering. In: IEEE Transactions on Pattern Analysis and Machine Intelligence (to appear, 2009)
7. Boutemedjet, S., Ziou, D., Bouguila, N.: Unsupervised Feature Selection for Accurate Recommendation of High-Dimensional Image Data. In: Advances in Neural Information Processing Systems (NIPS), pp. 177–184 (2007)
8. Boutemedjet, S., Bouguila, N., Ziou, D.: Unsupervised Feature and Model Selection for Generalized Dirichlet Mixture Models. In: Kamel, M.S., Campilho, A. (eds.) ICIAR 2007. LNCS, vol. 4633, pp. 330–341. Springer, Heidelberg (2007)
9. Wang, X., Kabán, A.: Finding Uninformative Features in Binary Data. In: Gallagher, M., Hogan, J.P., Maire, F. (eds.) IDEAL 2005. LNCS, vol. 3578, pp. 40–47. Springer, Heidelberg (2005)
10. Cardie, C.: A Cognitive Bias Approach to Feature Selection and Weighting for Case-Based Learners. Machine Learning 41(1), 85–116 (2000)
11. Dy, J.G., Brodley, C.E.: Feature Selection for Unsupervised Learning. Journal of Machine Learning Research 5, 845–889 (2004)
12. Law, M.H.C., Figueiredo, M.A.T., Jain, A.K.: Simultaneous Feature Selection and Clustering Using Mixture Models. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(9), 1154–1166 (2004)
13. Novovičová, J., Pudil, P., Kittler, J.: Divergence Based Feature Selection for Multimodal Class Densities. IEEE Transactions on Pattern Analysis and Machine Intelligence 18(2), 218–223 (1996)
14. Govaert, G.: Classification Binaire et Modèles. Revue de Statistique Appliquée 38(1), 67–81 (1990)
15. Govaert, G., Nadif, M.: Comparison of the Mixture and the Classification Maximum Likelihood in Cluster Analysis with Binary Data. Computational Statistics and Data Analysis 23(1), 65–81 (1996)
16. Ripley, B.D.: Pattern Recognition and Neural Networks. Cambridge University Press, Cambridge (1996)
17. Akaike, H.: A New Look at the Statistical Model Identification. IEEE Transaction on Automatic Control AC-19(6), 716–723 (1974)
18. Rissanen, J.: Modeling by Shortest Data Description. Automatica 14, 445–471 (1978)

19. Wallace, C.S.: Statistical and Inductive Inference by Minimum Message Length. Springer, Heidelberg (2005)
20. Bouguila, N., Ziou, D.: Unsupervised Selection of a Finite Dirichlet Mixture Model: An MML-Based Approach. IEEE Transactions on Knowledge and Data Engineering 18(8), 993–1009 (2006)
21. Bouguila, N., Ziou, D.: High-Dimensional Unsupervised Selection and Estimation of a Finite Generalized Dirichlet Mixture Model Based on Minimum Message Length. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(10), 1716–1731 (2007)
22. Kontkanen, P., Myllymaki, P., Silander, T., Tirri, H., Grunwald, P.: On Predective Distributions and Bayesian Networks. Statistics and Computing 10, 39–54 (2000)
23. Rissanen, J.: Stochastic Complexity. Journal of the Royal Statistical Society. Series B (Methodological) 49(3), 223–239 (1987)
24. Kass, R.E., Raftery, A.E.: Bayes Factors. Journal of the American Statistical Association 90, 773–795 (1995)
25. Schwarz, G.: Estimating the Dimension of a Model. Annals of Statistics 16, 461–464 (1978)
26. Cheeseman, P., Stutz, J.: Bayesian Classification (AutoClass): Theory and Results. In: Advances in Knowledge Discovery and Data Mining, ch. 6, pp. 153–180 (1995)
27. Blake, C.L., Merz, C.J.: Repository of Machine Learning Databases. University of California, Irvine, Dept. of Information and Computer Sciences (1998), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
28. Li, T.: A General Model for Clustering Binary Data. In: Proc. of the eleventh ACM SIGKDD international Conference on Knowledge Discovery in Data mining (KDD), pp. 188–197. ACM Press, New York (2005)
29. Boutemedjet, S., Ziou, D., Bouguila, N.: A Graphical Model for Content Based Image Suggestion and Feature Selection. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS (LNAI), vol. 4702, pp. 30–41. Springer, Heidelberg (2007)

Multi-resolution Boosting for Classification and Regression Problems

Chandan K. Reddy[1] and Jin-Hyeong Park[2]

[1] Department of Computer Science, Wayne State University,
Detroit, MI-48202, USA
reddy@cs.wayne.edu
[2] Integrated Data Systems Department, Siemens Corporate Research,
Princeton, NJ-08540, USA
jin-hyeong.park@siemens.com

Abstract. Various forms of boosting techniques have been popularly used in many data mining and machine learning related applications. Inspite of their great success, boosting algorithms still suffer from a few open-ended problems that require closer investigation. The efficiency of any such ensemble technique significantly relies on the choice of the weak learners and the form of the loss function. In this paper, we propose a novel multi-resolution approach for choosing the weak learners during additive modeling. Our method applies insights from multi-resolution analysis and chooses the optimal learners at multiple resolutions during different iterations of the boosting algorithms. We demonstrate the advantages of using this novel framework for classification tasks and show results on different real-world datasets obtained from the UCI machine learning repository. Though demonstrated specifically in the context of boosting algorithms, our framework can be easily accommodated in general additive modeling techniques.

1 Introduction

In the field of data mining, ensemble methods have been proven to be very effective for not only improving the classification accuracies but also in reducing the bias and variance of the estimated classifier. We choose to demonstrate our multi-resolution based framework using '*boosting*' algorithm, which is a standard additive modeling algorithm popular in data mining and machine learning domains. The Boosting meta-algorithm is an efficient, simple, and easy to manipulate additive modeling technique that can use potentially any weak learner available [8]. The most popular variant of boosting, namely the AdaBoost (Adaptive Boosting) in combination with trees has been described as the "best off-the-shelf classifier in the world" [3]. In simple terms, boosting algorithms combine weak learning models that are slightly better than random models. Recently, several researchers in other domains like computer vision, medical imaging have started using boosting algorithms extensively for real-time applications. Both classification and regression based boosting algorithms have been successfully

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 196–207, 2009.
© Springer-Verlag Berlin Heidelberg 2009

used in a wide variety of applications in the fields of computer vision [12], information retrieval [11], bioinformatics [9] etc. Inspite of their great success, boosting algorithms still suffer from a few open-ended issues such as the choice of the parameters for the weak learner. The framework proposed in this paper is more generally termed as "*Multi-resolution Boosting*", which can model any arbitrary function using the boosting methodology at different resolutions of either the model or the data. Here, we propose a novel boosting model that can take advantage of using the weak learners at multiple resolutions. This method of handling different resolutions and building effective models is similar to wavelet decomposition methods for multi-resolution signal analysis. In this work, we achieve this *multi-resolution* concept in the context of boosting algorithms by one of the following two ways:

- *Model-driven multi-resolution:* This is achieved by varying the complexity of the classification boundary. This approach will provide a systematic procedure that increases the complexity of the weak learner as the boosting iterations progress. This framework not only obtains weak learners in a systematic manner, but also reduces the over-fitting problem as discussed in Section 4.1 of this paper.
- *Data-driven multi-resolution:* This can be achieved by considering the data (not the model) at multiple resolutions during each iteration in the boosting algorithm. Our framework chooses the weak learners for the boosting algorithm that can best fit the current resolution and as the additive modeling iterations progress, the modeling resolution is increased. The amount of increase in the resolution follows from the theory of wavelet decomposition. Our algorithm provides the flexibility for dynamically choosing the weak learner compared to static learners with certain pre-specified parameters. This framework is discussed in Section 4.2 of this paper.

The main idea of the proposed framework is: *the use of Multi-resolution data (or model) driven fitting in the context of additive modeling using concepts that are similar to wavelet decomposition techniques.* The rest of the paper is organized as follows: Section 2 gives some relevant background on various boosting techniques and scale-space kernels. Section 3 shows the problem formulation in detail and discusses the concepts necessary to comprehend our algorithm. Section 4 describes both the model-driven and the data-driven multi-resolution boosting frameworks. Section 5 gives the experimental results of the proposed methods on real-world datasets and Section 6 concludes our discussion.

2 Relevant Background

Ensemble learning [4] is one of the most powerful modeling techniques that was found to be effective in a wide variety of applications in recent years. different ensemble techniques have been proposed in the literature and is still a very active area of research. Boosting is one of the most widely used algorithm that has

caught the attention of several researchers working in the areas of pattern recognition and machine learning [5]. A main advantage of boosting algorithms is that the weak learner can be a black-box which can deliver only the result in terms of accuracy and can potentially be any weak learner. This is a very desirable property of the boosting algorithms that can be applied in several applications for predictive modeling [8,6]. The additive model provides a reasonable flexibility in choosing the optimal weak learners for a desired task. In this paper, we propose a novel multi-resolution framework for choosing optimal weak learners during the iterations in boosting. This approach allows for effective modeling of the dataset at any given resolution [10]. In terms of analyzing (or modeling) a given dataset at different resolutions, our approach closely resembles wavelet decomposition techniques which are effective tools in the field of multi-resolution signal analysis [7]. In the model-driven multi-resolution boosting framework, the models are built by increasing the complexity during the boosting process. The data-driven multi-resolution, on the other hand, considers the data at different resolutions decomposition techniques which are effective tools in the field of multi-resolution signal analysis. The main advantages of using this multiple resolution framework in the context of boosting are that they:

- allow systematic hierarchical modeling of the final target model.
- provide more flexibility by allowing the user to stop at a reasonable resolution and thus avoid the over-fitting problem.
- require very few pre-defined user parameters.
- avoid the use of strong learners in the beginning stages of modeling and progressively use them towards the end.

3 Problem Formulation

Let us consider N i.i.d. training samples $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$ consisting of samples $(\mathcal{X}, \mathcal{Y}) = (x_1, y_1), (x_2, y_2), .., (x_N, y_N)$ where $\mathcal{X} \in \mathbb{R}^{N \times d}$ and $\mathcal{Y} \in \mathbb{R}^{N \times 1}$. For the case of binary classification problems, we have $y_i \in \{-1, +1\}$ and for regression problems, y_i takes any arbitrary real value. In other words, the univariate response $\mathcal{Y}$ is continuous for regression problems and discrete for classification problems. Now, we will discuss boosting algorithms applied to general classification problems. We choose to demonstrate the power of scale-space kernels in the context of Logitboost algorithm because of its popularity and its power of demonstrating the additive modeling nature.

Each boosting iteration performs the following three steps: (1) Computes response and weights for every datapoint. (2) Fits a weak learner to the weighted training samples and (3) Computes the error and updates the final model. In this way, the final model obtained by boosting algorithm is a linear combination of several weak learning models.

In the case of classification problems, the penalty function induced by the error estimation is given by:

$$L(y_i, F_t(x_i)) = I(y_i \neq F^{(t)}(x_i)) \tag{1}$$

where I denotes an indicator function which returns value 0, when $y_i \neq F^{(t)}(x_i)$ and 1 otherwise. In other words, the penalty term is 1 if the i^{th} sample is misclassified and 0 if it is correctly classified. Whether it is a classification or a regression problem, the main challenges in the boosting framework are the following: (i) The choice of the weak learner and (ii) The complexity of the weak learner. While choosing a weak learner model can be a complicated task in itself, tuning the right complexity for such a weak learner might be even more challenging. The multi-resolution framework proposed in this paper addresses the second issue.

The boosting framework discussed above works for classification problems and can be easily adapted to solve regression problems. In the case of regression problems, the penalty function is given by:

$$L(y_i, F^{(t)}(x_i)) = \|y_i - F^{(t)}(x_i)\|_p \tag{2}$$

where $\|\cdot\|_p$ indicates the L_p norm. We will consider $p = 2$ (namely, the Euclidean norm) in this paper. We formulate this multi-resolution boosting using the standard boosting algorithm with exponential L_2 norm loss function and demonstrate empirical results on classification problems. In our previous work [10], we have demonstrated the use of scale-space kernels in the data-driven boosting framework on several regression datasets.

Algorithm 1. Model-driven Multi-Resolution Boosting

Input: Data ($\mathcal{X}$), No. of samples (N), No. of iterations (T).
Output: Final model (F)
Algorithm:
Initialize the weight vector $W^{(1)}$ such that $w_i^1 = 1/N$ for $i = 1, 2, .., N$
$nplits = 1$
for $t = 1$ to T **do**
  $[\hat{f}_0, err_0] = Train(\mathcal{X}, W^{(t)}, nsplits)$
  $[\hat{f}_1, err_1] = Train(\mathcal{X}, W^{(t)}, nsplits + 2)$
  if $err_0 < err_1$ **then**
    $f_t = \hat{f}_0 \quad \epsilon_t = err_0$
  else
    $f_t = \hat{f}_1 \quad \epsilon_t = err_1$
    $nsplits = nsplits + 1$
  end if
  Compute $\alpha_t = \frac{1}{2} ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$
  Modify the training weight $w_i^{(t+1)}$ as follows:

$$w_i^{(t+1)} = \frac{w_i^{(t)} \cdot exp(-\alpha_t y_i f_t(x_i))}{z_t}$$

  where z_t is the normalization factor (chosen so that $\sum_{i=1}^{N} w_i^{(t+1)} = 1$)
end for
Output the final model $F(\mathcal{X}) = \sum_{t=1}^{T} \alpha_t f_t(\mathcal{X})$

4 Multi-resolution Boosting Framework

We will now describe both the model-driven and data-driven multi-resolution boosting algorithms. To demonstrate a reasonably wide applicability of the multi-resolution framework, we implement our framework using both the adaboost and logitboost algorithms. We show the model-driven multi-resolution algorithm using the adaboost framework for classification problems and the data-driven multi-resolution algorithm using the logitboost framework for regression problems. Though, we chose to demonstrate in this setting, the proposed framework is generic and can be applied to other additive modeling techniques used for solving classification and regression problems.

4.1 Model-Driven Multi-resolution Boosting

In the model driven boosting framework, the complexity of weak learner is modified as the boosting iterations progress. Changing the complexity of the weak model can be done in a very intuitive manner depending on the choice of the weak learner. For example, if decision trees are used as a weak learner, the resolution can be changed by changing the number of levels in the decision tree that is being considered. The initial boosting iterations use trees with only one level (or decision stumps) and later on the resolution can be increased by increasing the tree-depth. One has to note that the complexity of the modeling (or classification boundary) is significantly increased by changing the resolution. Algorithm 1 describes our model-driven multi-resolution boosting framework using the adaboost algorithm for a *binary classification* problem. The weight vector W is initialized to $1/N$ (uniform). The main algorithm runs for a predefined number (T) of iterations. The procedure $Train$ will obtain weak learner (and the corresponding training error) using the weights $W^{(t)}$. The number of splits ($nsplits$) is a parameter that determines the complexity of the model i.e. the more the number of splits in the weak learner, the more the complexity of the model. It is initialized to one at the beginning. As the iterations progress, the complexity of the weak learner is either retained or incremented depending upon the training error.

For every iteration, the training error of the current model is compared with the error of a slightly complex model (with $nsplits + 2$ nodes in the tree). If this new model performs well, then the complexity of the current model is increased ($nsplits = nsplits + 2$) and the re-weighting of the data points is computed using this new model. The weights are normalized (so that they sum to one) in every iteration. One can see that the algorithm appears to be working in a similar manner to the traditional Adaboost, except for the fact that the choice of the weak learner is made more systematically from simple to complex and is not chosen arbitrarily as done in the standard boosting procedure. In this manner, the algorithm increases the complexity of the weak learners chosen and the final weighted combinations of the selected weak learners are used as the final trained model. Hence, the model will have a very simple classification boundary

in the initial stages and the boundary becomes more and more complex as the iterations proceed.

4.2 Data-Driven Multi-resolution Boosting

In this section, we will describe the data-driven approach where we maintain the same complexity of the weak learner, but change the number of data points to be modeled during each boosting iteration. Algorithm 2 describes our data-driven multi-resolution boosting framework for a *regression problem.* As mentioned earlier, this approach is demonstrated using the logitboost algorithm. The initial model is set to null or to the mean value of the target values. The main program runs for a predefined number (T) of iterations. Initially, *res* is set to 1 indicating the simplest model possible (which will consider all the data points). The feature values are sorted independently by column-wise and the indices corresponding to each column are stored. As the iterations progress, the resolution considered for fitting the weak learner is retained or doubled depending on the error. In other words, depending on the error obtained at a given iteration, the resolution of the data is either maintained or increased for the next iteration. For every iteration, the residual r is computed depending on the difference between the target value ($\mathcal{Y}$) and the final model (F). By equating the first derivative of the loss function to zero, we will set the residual as the data to be modeled during the next iteration using another weak regressor. Using the quasi-Newton's method the data to be modeled in the next iteration will be set to $-(I + 2rr^T)^{-1} \cdot r$. The best multivariate Gaussian model will be fitted to this data at a given resolution.

Theorem 4.1. *During each boosting iteration, the minimum of the loss function is achieved by setting $f = r$ and the Newton's update is chosen by setting $f = -(I + 2rr^T)^{-1} \cdot r$.*

Proof. We will discuss the derivations for the first derivative and the second derivative and show the Newton updates in the case of the boosting for regression problems. Consider the following exponential loss function:

$$L(y, F, f) = exp(\parallel y - F - f \parallel^2)$$

For the Newton's update equation, we need to compute the first and second derivatives with respect to $f(x)$ and evaluate them at $f(x) = 0$.

$$s(x) = \frac{\partial L(y, F, f)}{\partial f(x)}|_{f(x)=0} = 2exp(\parallel r - f \parallel)(r - f)|_{f=0} = 2 \cdot exp(r^T r) \cdot r$$

Taking the derivative again, we have

$$H(x) = \frac{\partial^2 L(y, F, f)}{\partial f(x)^2}|_{f(x)=0} = 2exp(\parallel r - f \parallel^2) \cdot I$$
$$+ 4exp(\parallel r - f \parallel^2) \cdot (r - f) \cdot (r - f)^T|_{f=0} = 2exp(r^T r) \cdot (I + 2rr^T)$$

Hence, the inverse of the Hessian becomes

$$H^{-1}(x) = \frac{(I + 2rr^T)^{-1}}{2exp(r^T r)}$$

Finally, the Newton's update is given as follows:

$$F(x) = F(x) - H(x)^{-1}s(x) = F(x) - (I + 2rr^T)^{-1} \cdot r$$

Hence, we plug-in the value $-(I + 2rr^T)^{-1} \cdot r$ as the regression value to be modeled using the weak regressor. Also, we can notice that the minimum of the loss function can also be obtained by equating the first derivative to zero.

$$2exp(\parallel r - f \parallel)(r - f) = 0 \Rightarrow r = f$$

In other words, by modeling the residual directly using the weak regressor, the minimum of the loss function can be obtained. *End of Proof*

The details of the procedure $bestfit$ which obtains the best weak model at a given resolution of the data is described in the next section. The main reason for retaining the resolution of the next iteration is that sometimes there might be more than one significant component at that given resolution. One iteration can model only one of these components. In order to model the other component, one has to perform another iteration of obtaining the best weak model at the same resolution. Increasing the resolution for the next iteration might fail to model the component accurately. After ensuring that there are no more significant components at a given resolution, our algorithm will increase the resolution for the next iteration. Hence, the best weak model corresponding to current resolution or next higher resolution is obtained at every iteration and the model with the lowest error is added to the final model.

For every iteration, the best weak model is fit to the data based on a single feature value at a given resolution. This is performed using the $bestfit$ function in the algorithm. One way of achieving the multi-resolution in this context is to use scale-space kernel to model a subset of data and handling the data in a multi-resolution fashion. The procedure $bestgaussfit$ (instead of $bestfit$) performs this task for a particular value of resolution. Additive modeling with smooth and continuous kernels will result in smooth functions for classifier boundary and regression functions. Gaussian kernels are a simple and a trivial choice for scale-space kernels that are powerful universal approximators. Also, Gaussian kernels allow generative modeling of a target function which is a good choice for many applications like object detection. The basic idea is to slide a Gaussian window across all the datapoints corresponding to each feature at a given resolution. Algorithm 3 contains two loops. The outer loop ensures that the Gaussian fit has to be computed for each feature and the inner loop corresponds to the sliding Gaussian. In other words, depending on the given resolution (indicated by n datapoints), a Gaussian kernel containing n datapoints is moved across all the data points and the location where the minimal residual error is obtained.

The result f is obtained by fitting a Gaussian kernel computed using weighted median (μ) and standard deviation (σ) for the datapoints within this window.

Algorithm 2. Data-driven Multi-Resolution Boosting

Input: Data ($\mathcal{X}$), No. of samples (N), No. of iterations (T).
Output: Final Model (F)
Algorithm:
set $res = 1, \quad F = \emptyset$
for $i = 1 : d$ **do**
  $[\hat{\mathcal{X}}, idx(:, i)] =$sort$(\mathcal{X}(:, i))$
end for
for $t = 1 : T$ **do**
  $r = L(\mathcal{Y}, F)$
  $[\hat{f}_0, err_0] = bestfit(\hat{\mathcal{X}}, r, N, d, res, idx)$
  $[\hat{f}_1, err_1] = bestfit(\hat{\mathcal{X}}, r, N, d, res * 2, idx)$
  if $err_0 < err_1$ **then**
    $F = F + \hat{f}_0$
  else
    $F = F + \hat{f}_1$
    $res = res * 2$
  end if
end for
return F

Algorithm 3. *bestgaussfit*

Input: Sorted feature data ($\hat{\mathcal{X}}$), No. of samples(N), No. of samples to fit Gaussian (n), Residual vector (r), Sorting indices (idx).
Output: Best fit Regressor ($\hat{f}$), Error (Err_{min})
Algorithm:
$Err_{min} =$MAXDOUBLE
for $i = 1 : d$ **do**
  for $j = 1 : N - n + 1$ **do**
    $\hat{x} = \hat{\mathcal{X}}(:, j : j + n - 1)$
    $\hat{r} = r(idx(j : j + n - 1, i))$
    $wgt(1 : n) = abs(\hat{r}(1 : n))/sum(abs(r))$
    $\mu = E_{wgt}(\hat{x}) = wgt^T * \hat{x}$
    $\sigma =$ sqrt$(E_{wgt}((\mu - \hat{x})^2))$
    $f =$normpdf$(\hat{X}, \mu, \sigma)$
    $\beta =$sum$(\hat{r})/$sum$(f(j : j + n - 1))$
    $err = (r - \beta f)^T \cdot (r - \beta f)$
    if $err < Err_{min}$ **then**
      $Err_{min} = err$
      $\hat{f} = f$
    end if
    $f = min(\hat{f}(1 : d))$
  end for
end for
return $\{f, Err_{min}\}$

After obtaining the weak learner it must be scaled (scale factor is β) according to the target values. Finally, the error is computed between the weak learner and the target values. If the error with the new model is improved, the resolution is doubled (change at a logarithmic scale) or in other words, the number of datapoints considered to fit a Gaussian is halved. In fact, we can use any other heuristic to change the resolution more efficiently. Experimental results showed that this change of resolution is optimal and also this logarithmic change of resolution has nice theoretical properties as they mimic some of the wavelet decomposition methods.

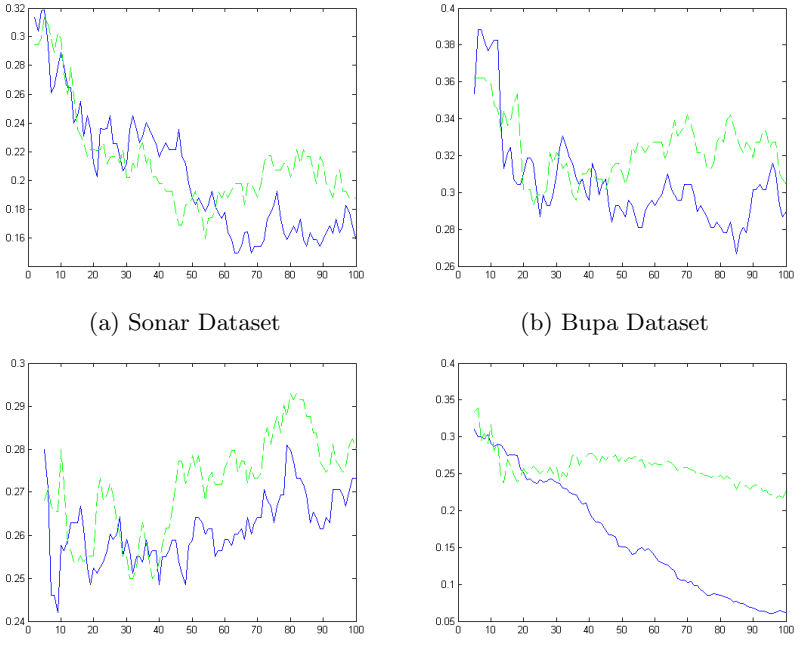
The multi-resolution aspect of our algorithm can be seen from the fact that the resolution of the data to be modeled is either maintained or increased as the number of iterations increase. In fact, one might interpret this approach as an improvement in the weak learner alone because the algorithm proposed here will obtain improved weak learner at every iteration and hence the overall boosting will have faster convergence. We consider that the main contribution of this paper is not just at the level of choosing a weak learner but it is at the junction between the choice of weak learner and the iterations in the boosting algorithm. Also, our algorithm obtains the weak models in a more systematic hierarchical manner. Most importantly, the increase in the resolution is monotonically non-decreasing, i.e. the resolution either remains the same or increased.

5 Experimental Results

We will now demonstrate our results on some real-world datasets. All experiments were run in MATLAB 7.0 and on a pentium IV 2.8 GHz machine. Six different real world binary classification datasets were chosen from the UCI machine learning repository [2]. Multi-class classification problems can also be performed using methods similar to [1]. Two different sets of experiments were conducted on these datasets to illustrate the power of multi-resolution boosting. In order to demonstrate the model-driven framework, decision trees at multiple resolutions (different number of levels in the decision tree) are considered, and in order to demonstrate the data-driven framework, Gaussian kernels are considered for fitting the data at multiple resolutions.

5.1 Results for Model-Driven Multi-resolution

Fig. 1 shows the test error results on different datasets during the boosting iterations. Comparisons are made between the standard Adaboost and the multi-resolution boosting framework. We can see that the error obtained using the multi-resolution boosting procedure is significantly lower compared to the standard procedure. This clearly illustrates the fact that the multi-resolution scheme is less prone to the over-fitting problem. Under this framework, during the initial iterations of boosting, decision stumps (trees with only one level of child nodes) are used. As the iterations proceed, more deeper trees (with levels greater than 2) are used for modeling. This way, a hierarchical approach is used for computing the classification boundary from low resolution to high resolution. Using a



(a) Sonar Dataset (b) Bupa Dataset

(c) Diabetes Dataset (d) Tic-Tac-Toe Dataset

Fig. 1. Test error during the boosting iterations on various datasets. The dashed line gives the error obtained using the standard Adaboost algorithm and the solid line gives the error obtained using the model-driven multi-resolution boosting algorithm.

high-resolution weak models will suffer from the problem of over-fitting. For example, by using a tree with many levels in the first few iterations in the boosting procedure might obtain a very complicated decision boundary which is prone to the over-fitting problem. Also, it will be expensive to use complex trees from the start of the boosting procedure when it is not required to have a complex decision boundary.

5.2 Results for Data-Driven Multi-resolution

We demonstrate the power of data-driven multi-resolution approach using scale-space kernels on binary classification problems. Additive modeling with smooth and continuous kernels will result in smooth functions for classifier boundary and regression functions. Since, obtaining the width of the kernel during the boosting process can be a challenging task, the use of scale-space kernels can resolve the problem by using adaptive step-sizes by a '*global-to-local*' fitting process. One cannot predetermine the reduction in the kernel width. In our multi-resolution framework, we choose to reduce it by halves using the concepts of wavelet decomposition methods which were well studied concepts in the context of handling image operations efficiently. We compare the performance of these scale-space

Table 1. Experimental results of Data-Driven Multi-Resolution boosting. Performance of scale-space kernels with other kernels on various real-world datasets. Test error along with the standard deviation using five-fold cross validation procedure is reported.

Dataset	Cancer	Ionosphere	Sonar	Bupa	Tic-Tac-Toe	Diabetes
Number of Samples	569	351	208	345	958	768
Number of Features	30	34	60	6	9	8
static kernel -n/2	0.1938±0.05	0.3647± 0.08	**0.6632± 0.13**	0.8785± 0.08	0.6947± 0.04	0.6765± 0.06
static kernel -n/4	0.1993 ±0.03	**0.333 ±0.08**	0.6697±0.07	0.9156 ± 0.11	0.5725± 0.02	0.6419±0.06
static kernel -n/8	0.244± 0.09	0.4118± 0.1	0.9148± 0.13	0.9453± 0.06	0.5644 ± 0.02	0.657±0.019
static kernel -8	0.7638 ±0.07	0.503 ±0.06	1.144 ± 0.15	0.9384± 0.11	0.5662± 0.03	0.7487±0.06
Dynamic kernel	0.1898± 0.03	0.3553 ± 0.06	0.7543± 0.09	**0.869 ±0.05**	0.5726± 0.03	0.6676±0.07
Exhaustive kernel	0.2325± 0.06	0.4243 ± 0.12	0.8068±0.30	0.9643± 0.12	0.5624±0.04	0.6546 ±0.07
Scale-space kernel	**0.1895 ± 0.04**	0.3371± 0.09	0.7125± 0.14	0.8962 ± 0.13	**0.5603± 0.038**	**0.6386± 0.05**

kernels with other static and dynamic kernels. Exhaustive kernel is the most expensive one which tries to fit a kernel of various widths during each iteration of boosting. Dynamic kernel (or random kernel) fits a kernel of random width during the boosting process. Static kernels will have static widths that do not change during the boosting process.

Compared to other static kernels of fixed width, the scale-space kernels do not suffer from the generalization problem as clearly illustrated by the results on the test data shown in Table 1. Scale-space kernels consistently perform better than the exhaustive or dynamic kernels. For some datasets, wider static kernels perform better than the scale-space kernels and for other datasets static kernels with lesser width perform better. However, scale-space kernels are competitive with the best possible kernels and can be generically used for any dataset. Overall, the scale-space kernels are less than twice as expensive as the static width kernels. One can also see that the results of the scale-space kernels are fairly robust compared to other kernels. This multi-resolution framework will provide a systematic hierarchical approach of obtaining the classification boundary in the context of additive modeling. One of the main reasons for using the scale-space framework is for faster convergence of the results by *dynamically choosing the weak regressors* during the boosting procedure. Choosing an optimal weak regressor by exploring all possibilities might yield a better result, but it will be computationally inefficient and infeasible for most of the practical problems. For such problems, scale-space kernels will give the users with a great flexibility of adaptive kernel scheme at a very low computational effort (also considering fast convergence). The fact that the scale-space kernels converge much faster than static kernels make them more suitable for additive modeling algorithms. To the best of our knowledge, this is the first attempt to use the concepts of scale-space theory and wavelet decomposition in the context of boosting algorithms for predictive modeling.

6 Conclusion

Recently, additive modeling techniques have received a great attention from several researchers working in a wide variety of applications in science and engineering. Choosing optimal weak learners and setting their parameters during the modeling have been a crucial and challenging task. In this paper, we

proposed a novel boosting algorithm that uses multi-resolution framework to obtain the optimal weak learner at every iteration. We demonstrated our results for logitboost based regression problems on real-world datasets. Advantages of our method compared to existing methods proposed in the literature is clearly demonstrated. As a continuation of this work, we would like to perform the generalization of the multi-resolution approach for other ensemble learning techniques.

References

1. Allwein, E., Schapire, R., Singer, Y.: Reducing multiclass to binary: a unifying approach for margin classifiers. J. Machine Learning Research 1, 113–141 (2001)
2. Blake, C.L., Merz, C.J.: UCI repository of machine learning databases. University of California, Irvine, Dept. of Information and Computer Sciences (1998), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
3. Breiman, L.: Arcing classifiers. The Annals of Statistics 26(3), 801–849 (1998)
4. Dietterich, T.G.: Ensemble methods in machine learning. In: First International Workshop on Multiple Classifier Systems, pp. 1–15 (2000)
5. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: International Conference on Machine Learning, pp. 148–156 (1996)
6. Friedman, J.H., Hastie, T., Tibshirani, R.: Additive logistic regression: A statistical view of boosting. Annals of Statistics 28(2), 337–407 (2000)
7. Graps, A.L.: An introduction to wavelets. IEEE Computational Sciences and Engineering 2(2), 50–61 (1995)
8. Hastie, T., Tibshirani, R., Friedman, J.: Boosting and Additive Trees. In: The Elements of Statistical Learning. Data Mining, Inference, and Prediction. Springer, New York (2001)
9. Hong, P., Liu, X.S., Zhou, Q., Lu, X., Liu, J.S., Wong, W.H.: A boosting approach for motif modeling using chip-chip data. Bioinformatics 21(11), 2636–2643 (2005)
10. Park, J.-H., Reddy, C.K.: Scale-space based weak regressors for boosting. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS, vol. 4701, pp. 666–673. Springer, Heidelberg (2007)
11. Schapire, R., Singer, Y., Singhal, A.: Boosting and rocchio applied to text filtering. In: Proceedings of ACM SIGIR, pp. 215–223 (1998)
12. Viola, P.A., Jones, M.J.: Robust real-time face detection. International Journal of Computer Vision 57(2), 137–154 (2004)

Interval Data Classification under Partial Information: A Chance-Constraint Approach

Sahely Bhadra[1], J. Saketha Nath[2], Aharon Ben-Tal[2], and Chiranjib Bhattacharyya[1]

[1] Dept. of Computer Science and Automation, Indian Institute of Science, Bangalore, India
{sahely,chiru}@csa.iisc.ernet.in
[2] MINERVA Optimization center, Faculty of Industrial Engg. and Management, Technion, Haifa, Israel
saketh@tx.technion.ac.il, abental@ie.technion.ac.il

Abstract. This paper presents a Chance-constraint Programming approach for constructing maximum-margin classifiers which are robust to interval-valued uncertainty in training examples. The methodology ensures that uncertain examples are classified correctly with high probability by employing chance-constraints. The main contribution of the paper is to pose the resultant optimization problem as a Second Order Cone Program by using large deviation inequalities, due to Bernstein. Apart from support and mean of the uncertain examples these Bernstein based relaxations make no further assumptions on the underlying uncertainty. Classifiers built using the proposed approach are less conservative, yield higher margins and hence are expected to generalize better than existing methods. Experimental results on synthetic and real-world datasets show that the proposed classifiers are better equipped to handle interval-valued uncertainty than state-of-the-art.

1 Introduction

In the recent past there has been a growing interest in analysis of interval-valued data in the learning community [1,2]. Data generated in many real world problems are subject to interval valued uncertainty. In such cases it maynot be possible to describe the data by a precise value but intervals may be a more proper description. For example, in the case of cancer diagnosis, a tumorous tissue is analyzed and features are computed for each cell nucleus. However, since the features vary among cells of a tissue, usually, the mean and worst-case (minimum/maximum) feature values of the tissues are considered[1]. Micro-array data, obtained by profiling experiments on genetic markers, are another such instances of noisy data. Past research has shown that handling uncertainty in

[1] Examples are the Wisconsin breast cancer diagnostic/prognostic datasets available at http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Original)

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 208–219, 2009.
© Springer-Verlag Berlin Heidelberg 2009

such applications by the representation as interval data leads to accurate learning algorithms [3,1]. Classification formulations which are capable of handling interval data have immense importance from a pragmatic perspective. This paper presents a maximum-margin classification formulation which uses means and bounding hyper-rectangles (support) of the interval-valued training examples in order to build the decision function. As shown in the paper, the proposed classifier is robust to interval uncertainty and is also not overly-conservative.

The idea is to model interval-valued uncertainty using Chance-Constrained Programming (CCP). The main contribution of the paper is to approximate the CCP as a Second Order Cone Program (SOCP) using Bernstein schemes [4]. SOCPs are well studied convex optimization problems with efficient interior point solvers (e.g. `SeDuMi` [5]). The key advantage of the Bernstein scheme is that no assumptions on distributions regarding the underlying uncertainty are made and only partial information like support and mean of the uncertain examples is required. Geometric interpretation of the SOCP formulation reveals that the classifier views each example with interval uncertainty as a region of intersection of its bounding hyper-rectangle and an ellipsoid centered at its mean. Thus the proposed classifier is far less conservative than the methods which utilize the bounding hyper-rectangle information alone. Since a classifier's conservativeness directly affects the classification margin achieved, the proposed classifier is expected to generalize better. Methodology of classifying uncertain test examples is discussed and error measures for evaluating performance of interval data classifiers are presented. Experimental results show that the proposed classifier outperforms state-of-the-art when evaluated using any of the discussed error measures.

The paper is organized as follows: in section 2, the main contributions of the paper are presented. In section 3, experiments on real-world and synthetic data are presented. Section 4 summarizes the work and concludes the paper.

2 Robust Classifiers for Interval-Valued Uncertainty

This section presents the main contribution of the paper, a novel maximum-margin formulation for interval data in section 2.1. A discussion on geometric interpretation of the proposed formulation is presented in section 2.2. The section concludes with a discussion on error measures which evaluate the performance of a classifier on interval data.

2.1 Maximum-Margin Formulation Using Bernstein Bounds

In this section, a maximum-margin classification formulation for interval data, which assumes the mean and the bounding hyper-rectangles are known for each example, is presented. It is also assumed that the features describing the data are independent. Let $\mathbf{X}_i = [X_{i1} \dots X_{in}]^\top$ be the random vector representing i^{th} training example (n denotes dimensionality of the data) and y_i denotes its label ($i = 1, \dots, m$). Let $\mathbf{a}_i = [a_{i1} \dots a_{in}]^\top$, $\mathbf{b}_i = [b_{i1} \dots b_{in}]^\top$ and $a_{ij} \leq X_{ij} \leq b_{ij}$,

so that $[\mathbf{a}_i, \mathbf{b}_i]$ represents the bounding hyper-rectangle of i^{th} example. Also let $\mathbf{E}[X]$ denote mean of the random variable X.

Consider the problem of constructing a maximum-margin classifier using the training example $\mathbf{X}_i$, which have interval-valued uncertainty. Let the discriminating hyperplane be denoted by $\mathbf{w}^\top \mathbf{x} - b = 0$. Then the constraints $y_i(\mathbf{w}^\top \mathbf{X}_i - b) \geq 1$ ensure that the training data is classified correctly. Slack variables, $\xi_i \geq 0$, can be introduced in order to handle outliers. Thus the classification constraints turn out to be $y_i(\mathbf{w}^\top \mathbf{X}_i - b) \geq 1 - \xi_i$. Since the constraints involve the random vector, X_i, they cannot be satisfied always. Hence, alternatively, one can ensure that the following chance-constraints are satisfied:

$$Prob\left(y_i(\mathbf{w}^\top \mathbf{X}_i - b) \leq 1 - \xi_i\right) \leq \epsilon \tag{1}$$

where $0 \leq \epsilon \leq 1$ is a small number denoting the upper bound on misclassification error made on an example and is a user-given parameter. Using these constraints the following maximum-margin formulation, similar in spirit to SVMs [6], can be written:

$$\begin{aligned} \min_{\mathbf{w},b,\xi_i} \quad & \tfrac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^m \xi_i \\ \text{s.t. } & Prob\left(y_i(\mathbf{w}^\top \mathbf{X}_i - b) \leq 1 - \xi_i\right) \leq \epsilon,\ \xi_i \geq 0,\ i = 1, \ldots, m \end{aligned} \tag{2}$$

In the following we will show that the CCP (2) can be approximated as an SOCP problem by using Bernstein bounds. To this end, the following theorem is presented, which specializes the Bernstein approximation schemes described in [4,7]:

Theorem 1. *Consider the following notation* ($\forall\ i = 1, \ldots, m,\ j = 1, \ldots, n$)*:*

$$\begin{aligned} l_{ij} &= \frac{b_{ij} - a_{ij}}{2} & m_{ij} &= \frac{b_{ij} + a_{ij}}{2} & \mu_{ij} &= \frac{\mathbf{E}[X_{ij}] - m_{ij}}{l_{ij}} \\ \mathbf{L}_i &= diag\left([l_{i1} \ldots l_{in}]\right) & \mathbf{m}_i &= [m_{i1} \ldots m_{in}]^\top & \mu_i &= [\mu_{i1} \ldots \mu_{in}]^\top \\ \mu_{ij}^{'} &= \mathbf{E}[X_{ij}] & \mu_i^{'} &= \left[\mu_{i1}^{'} \ldots \mu_{in}^{'}\right]^\top & \mathbf{\Sigma}_i &= diag\left([\sigma(\mu_{i1}) \ldots \sigma(\mu_{in})]\right) \end{aligned} \tag{3}$$

where $\sigma(\mu_{ij})$ *is given by:*

$$\sigma(\mu_{ij}) = \min\left\{\sigma \geq 0 \;\middle|\; \frac{\sigma^2}{2}t^2 + \mu_{ij}t - \log\left(\cosh(t) + \mu_{ij}\sinh(t)\right) \geq 0,\ \forall\ t\ \in\ \mathbb{R}\right\} \tag{4}$$

The chance-constraint (1), which represents the classification constraint for i^{th} *example, is satisfied if the following cone constraint, in variables* $\mathbf{w}, b, \xi_i, \mathbf{z}_i$*, holds:*

$$y_i(\mathbf{w}^\top \mu_i^{'} - b) + \mathbf{z}_i^\top \mu_i \geq 1 - \xi_i + \|\mathbf{z}_i\|_1 + \sqrt{2\log(1/\epsilon)}\,\|\mathbf{\Sigma}_i(y_i\mathbf{L}_i\mathbf{w} + \mathbf{z}_i)\|_2 \tag{5}$$

Proof. The chance-constraint (1) can be written as:

$$Prob\left(-y_i\mathbf{w}^\top\mathbf{X}_i + (1-\xi_i+y_ib) \geq 0\right) \leq \epsilon$$

Now, let variables $\mathbf{u}_i, \mathbf{v}_i$ be chosen such that:

$$\mathbf{u}_i + \mathbf{v}_i = -y_i\mathbf{w} \tag{6}$$

Since $\mathbf{a}_i \leq \mathbf{X}_i \leq \mathbf{b}_i$, we have that $\mathbf{v}_i^\top\mathbf{X}_i \leq \mathbf{v}_i^\top\mathbf{m}_i + \|\mathbf{L}_i\mathbf{v}_i\|_1$. Using this inequality, we have that the chance-constraint (1) is satisfied if:

$$Prob\left(\mathbf{u}_i^\top\mathbf{X}_i + u_{i0} \geq 0\right) \leq \epsilon \tag{7}$$

where $u_{i0} = 1-\xi_i+y_ib+\mathbf{v}_i^\top\mathbf{m}_i+\|\mathbf{L}_i\mathbf{v}_i\|_1$. Clearly, the advantage of introducing the variables $\mathbf{u}_i, \mathbf{v}_i$ (6) is to utilize the bounding hyper-rectangle information via the inequality $\mathbf{v}_i^\top\mathbf{X}_i \leq \mathbf{v}_i^\top\mathbf{m}_i + \|\mathbf{L}_i\mathbf{v}_i\|_1$ (also see lemma 2).

Using Markov inequality and independence of random variables $X_{ij}, j = 1,\ldots,n$, we have

$$Prob\left(\mathbf{u}_i^\top\mathbf{X}_i + u_{i0} \geq 0\right) \leq \exp\{\alpha u_{i0}\}\prod_j \mathbf{E}\left[\exp\{\alpha u_{ij}X_{ij}\}\right],\ \forall\ \alpha \geq 0 \tag{8}$$

The Key of modeling chance-constraint (7) now depends on how one upper-bounds the moment generating functions $\mathbf{E}\left[\exp\{tX_{ij}\}\right],\ t \in \mathbb{R}$. To this end, we use the following lemma:

Lemma 1. *Consider the notation in (3). Then,*

$$\mathbf{E}\left[\exp\{tX_{ij}\}\right] \leq \exp\left\{\frac{\sigma(\mu_{ij})^2 l_{ij}^2}{2}t^2 + \mu_{ij}^{'}t\right\}\ \forall\ t\ \in\ \mathbb{R} \tag{9}$$

Proof. The fact that $\exp\{tX_{ij}\}$ is a convex function gives the following inequality: $\exp\{tX_{ij}\} \leq \frac{b_{ij}-X_{ij}}{b_{ij}-a_{ij}}\exp\{ta_{ij}\} + \frac{X_{ij}-a_{ij}}{b_{ij}-a_{ij}}\exp\{tb_{ij}\}$. Taking expectation on both sides and re-writing the resulting inequality in terms of m_{ij}, l_{ij} gives:

$$\mathbf{E}\left[\exp\{tX_{ij}\}\right] \leq \exp\left\{m_{ij}t + h_{ij}(l_{ij}t)\right\} \tag{10}$$

where $h_{ij}(\beta) \equiv \log\left(\cosh(\beta) + \mu_{ij}\sinh(\beta)\right)$. Note that, $h_{ij}(0) = 0, h_{ij}^{'}(0) = \mu_{ij}$ and $h_{ij}^{''}(\beta) \leq 1\ ,\ \forall\ \beta$. This gives the inequality $h_{ij}(\beta) \leq \frac{1}{2}\beta^2 + \mu_{ij}\beta\ ,\ \forall\ \beta$. In fact, using $\sigma(\mu_{ij})$ as defined in (4), we have the tighter inequality, $h_{ij}(\beta) \leq \frac{\sigma(\mu_{ij})^2}{2}\beta^2 + \mu_{ij}\beta\ ,\ \forall\ \beta$. Using this inequality in (10), and noting that $\mu_{ij}^{'} = l_{ij}\mu_{ij} + m_{ij}$, we obtain (9). This completes the proof of Lemma 1. □

Using Lemma 1, from (8) we obtain: $\log\left[Prob\left(\mathbf{u}_i^\top\mathbf{X}_i + u_{i0} \geq 0\right)\right] \leq \alpha(u_{i0} + \mathbf{u}_i^\top\mu_i^{'}) + \frac{\alpha^2}{2}\|\mathbf{L}_i\mathbf{\Sigma}_i\mathbf{u}_i\|_2^2\ ,\ \forall\ \alpha \geq 0$. Since this inequality holds for all values of α, if we ensure that for certain α the right-hand side of the inequality is $\leq \log(\epsilon)$, then we would satisfy the chance-constraint (7). Choosing $\alpha^* = -\frac{u_{i0}+\mathbf{u}_i^\top\mu_i^{'}}{\|\mathbf{L}_i\mathbf{\Sigma}_i\mathbf{u}_i\|_2^2}$, which is

the one minimizing right-hand side of the inequality, we see that (7) and in turn (1) are satisfied if:

$$u_{io} + \mathbf{u}_i^\top \mu_i^{'} + \sqrt{2\log(1/\epsilon)}\|\mathbf{L}_i\mathbf{\Sigma}_i\mathbf{u}_i\|_2 \leq 0 \tag{11}$$

Substituting the value of u_{i0}, eliminating the variable $\mathbf{u}_i$ from (6), (11) and introducing the variable $\mathbf{z}_i = \mathbf{L}_i\mathbf{v}_i$ gives (5). This completes the proof of the theorem. □

The values of $\sigma(\mu_{ij})$ (4) can be calculated numerically (refer Appendix A). Using theorem 1 and CCP (2), a maximum-margin SOCP formulation for interval data which ensures that the probability of misclassification is less than ϵ, can be written as follows:

$$\begin{array}{ll} \min\limits_{\substack{\mathbf{w},b,\mathbf{z}_i,\\ \xi_i\geq 0}} & \dfrac{1}{2}\|\mathbf{w}\|_2^2 + C\sum\limits_{i=1}^{m}\xi_i \\ \text{s.t.} & y_i(\mathbf{w}^\top\mu_i^{'} - b) + \mathbf{z}_i^\top\mu_i \geq 1 - \xi_i + \|\mathbf{z}_i\|_1 + \kappa\left\|\mathbf{\Sigma}_i(y_i\mathbf{L}_i\mathbf{w} + \mathbf{z}_i)\right\|_2 \end{array} \tag{12}$$

where $\kappa = \sqrt{2\log(1/\epsilon)}$ and $\mu_i^{'}, \mu_i, \mathbf{L}_i, \mathbf{\Sigma}_i$ are as given in (3). As mentioned earlier, C and ϵ are user-given parameters.

2.2 Geometric Interpretation of the Formulation

In this section, a geometrical interpretation for the proposed formulation (12) is presented. To this end, consider the following lemma:

Lemma 2. *Consider the notation in (3) and let* $\mathbf{S}_i = \mathbf{L}_i^2\mathbf{\Sigma}_i^2, \kappa = \sqrt{2\log(1/\epsilon)}$. *Suppose* $\mathcal{E}(\mu_i^{'}, \mathbf{S}_i, \kappa)$ *represents the ellipsoid* $\left\{\mathbf{x} = \mu_i^{'} + \kappa\mathbf{L}_i\mathbf{\Sigma}_i\mathbf{u} \ : \ \|\mathbf{u}\|_2 \leq 1\right\}$ *and* $\mathcal{R}(\mathbf{a}_i, \mathbf{b}_i)$ *represents the hyper-rectangle* $\{\mathbf{x} \ : \ \mathbf{a}_i \leq \mathbf{x} \leq \mathbf{b}_i\}$. *Consider the problem of correctly classifying points belonging to the intersection of* $\mathcal{E}(\mu_i^{'}, \mathbf{S}_i, \kappa)$ *and* $\mathcal{R}(\mathbf{a}_i, \mathbf{b}_i)$*:*

$$y_i(\mathbf{w}^\top\mathbf{x} - b) \geq 1 - \xi_i, \ \forall \ \mathbf{x} \ \in \ \mathcal{E}(\mu_i^{'}, \mathbf{S}_i, \kappa) \cap \mathcal{R}(\mathbf{a}_i, \mathbf{b}_i) \tag{13}$$

The continuum of constraints represented in (13) is satisfied if and only if the constraint (5) holds.

Proof. The constraint (13) hold if and only if:

$$1 - \xi_i + y_i b + \left(\max_{\mathbf{x} \ \in \ \mathcal{E}(\mu_i^{'}, \mathbf{S}_i, \kappa) \cap \mathcal{R}(\mathbf{a}_i, \mathbf{b}_i)} -y_i\mathbf{w}^\top\mathbf{x}\right) \leq 0$$

Note that, $\max_{\mathbf{x} \ \in \ \mathcal{E}(\mu_i^{'}, \mathbf{S}_i, \kappa) \cap \mathcal{R}(\mathbf{a}_i, \mathbf{b}_i)} \left(-y_i\mathbf{w}^\top\mathbf{x}\right)$ is the support function of the set $\mathcal{E}(\mu_i^{'}, \mathbf{S}_i, \kappa) \cap \mathcal{R}(\mathbf{a}_i, \mathbf{b}_i)$ (denoted by $I_{\mathcal{E}(\mu_i^{'}, \mathbf{S}_i, \kappa) \cap \mathcal{R}(\mathbf{a}_i, \mathbf{b}_i)}(-y_i\mathbf{w})$). Since

support function of intersection of two sets is the infimal convolution of support functions of the individual sets (see section 16, [8]), we have that $I_{\mathcal{E}(\mu_i^{'},\mathbf{S}_i,\kappa)\cap\mathcal{R}(\mathbf{a}_i,\mathbf{b}_i)}(-y_i\mathbf{w}) = \inf\left\{I_{\mathcal{E}(\mu_i^{'},\mathbf{S}_i,\kappa)}(\mathbf{u}_i)+I_{\mathcal{R}(\mathbf{a}_i,\mathbf{b}_i)}(\mathbf{v}_i) \mid \mathbf{u}_i+\mathbf{v}_i=-y_i\mathbf{w}\right\}$. Thus we have:

$$(13) \Leftrightarrow 1-\xi_i+y_ib+\inf\left\{I_{\mathcal{E}(\mu_i^{'},\mathbf{S}_i,\kappa)}(\mathbf{u}_i)+I_{\mathcal{R}(\mathbf{a}_i,\mathbf{b}_i)}(\mathbf{v}_i) \mid \mathbf{u}_i+\mathbf{v}_i=-y_i\mathbf{w}\right\}\leq 0$$

$$\Leftrightarrow \exists\ \mathbf{u}_i,\mathbf{v}_i \ni \mathbf{u}_i+\mathbf{v}_i=-y_i\mathbf{w},$$

$$1-\xi_i+y_ib+\left\{I_{\mathcal{E}(\mu_i^{'},\mathbf{S}_i,\kappa)}(\mathbf{u}_i)+I_{\mathcal{R}(\mathbf{a}_i,\mathbf{b}_i)}(\mathbf{v}_i)\right\}\leq 0 \qquad (14)$$

Now it is easy to see that $I_{\mathcal{E}(\mu_i^{'},\mathbf{S}_i,\kappa)}(\mathbf{u}_i)=\mathbf{u}_i^\top\mu_i^{'}+\kappa\|\mathbf{L}_i\mathbf{\Sigma}_i\mathbf{u}_i\|_2$ and $I_{\mathcal{R}(\mathbf{a}_i,\mathbf{b}_i)}(\mathbf{v}_i)=\mathbf{v}_i^\top\mathbf{m}_i+\|\mathbf{L}_i\mathbf{v}_i\|_1$. Substituting these values in (14) and noting that $\mu_i^{'}=\mathbf{L}_i\mu_i+\mathbf{m}_i$, gives (6, 11). Since the constraints (6, 11) are equivalent to (5) (see proof of theorem 1), we have that (13) $\Leftrightarrow$ (5). This completes the proof. □

The above lemma shows that the proposed classifier (12) views each interval data example as the intersection of the bounding hyper-rectangle and an ellipsoid centered at its mean with semi-axis lengths proportional to $l_{ij},\sigma(\mu_{ij})$. In this way the proposed formulation takes into account both the mean and bounding hyper-rectangle informations. Note that, lemma 2 theoretically proves that the proposed classifier is always less conservative (pessimistic) than classifiers which use the bounding hyper-rectangle information alone [3] (this is because $\mathcal{E}(\mu_i^{'},\mathbf{S}_i,\kappa)\cap\mathcal{R}(\mathbf{a}_i,\mathbf{b}_i)\subset\mathcal{R}(\mathbf{a}_i,\mathbf{b}_i)$). In fact, it is easy to see that classifiers which utilize the bounding hyper-rectangle information alone and classifiers which utilize the mean information alone are special cases of the proposed formulation (with $\epsilon=0$ and $\epsilon=1$ respectively).

It is interesting to note the effect of $\sigma(\mu_{ij})$ (4) on the proposed classifier. As mentioned earlier, the semi-axis lengths of the uncertainty ellipsoid are proportional to $\sigma(\mu_{ij})$. Table 3 shows that as μ increases from 0 to 1, $\sigma(\mu)$ decreases from 1 to 0. In other words, as the mean of example shifts from center to a corner of the bounding hyper-rectangle, the size of the uncertainty ellipsoid decreases. This is very intuitive because, in one extreme case where mean lies at a corner, the datapoint is deterministic and in the other extreme case, where mean is at center of the hyper-rectangle, the uncertainty of the datapoint is maximum. This phenomenon is also illustrated in figure 1, where the bounding hyper-rectangle and the uncertainty region at various positions of the mean are shown. It can be seen that as the mean moves towards a corner, not only does the uncertainty region move but also its size decreases. However, a classifier which does not employ the mean information [3], always views the example as the whole hyper-rectangle. Thus the proposed classifier is robust to interval-valued uncertainty, as well as not overly-conservative.

2.3 Classification with Uncertain Examples

This section presents the methodology for labeling interval-valued test examples and discusses error measures for evaluating the performance of interval data

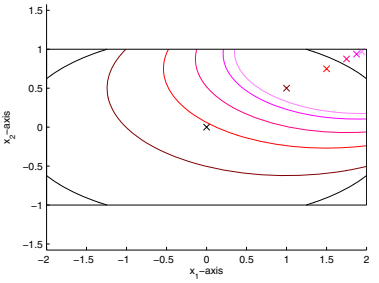


Fig. 1. Figure showing bounding hyper-rectangle and uncertainty sets for different positions of mean. Mean and boundary of uncertainty set marked with same color.

classifiers. Depending on the form in which the examples are available, the labeling methodologies summarized in table 1 can be employed. Here, y_i^{pr} denotes the predicted label for test example $\mathbf{X}_i$ (also refer (3) for notation). Once a test example is labeled using the appropriate methodology, the overall misclassification error for the given test dataset can be calculated as the percentage of examples in which y_i^{pr} and y_i do not agree:

$$\textbf{NomErr} = \frac{\sum_i 1_{y_i^{pr} \neq y_i}}{\#\ test\ examples} \times 100 \tag{15}$$

Note that, the proposed formulation can be solved when the training examples are in any of the 3 forms shown in table 1. In case of form 2, the support and mean information are readily available and in case of form 3 these partial information can be easily estimated from the replicates. In case of form 1, since no mean information is available the proposed formulation can be solved using $\epsilon = 0$, which as discussed in section 2.2 is the maximum-margin classifier built using support information alone.

Based on the discussion in section 2.2, another interesting error measure can be derived. Given an uncertain test example X_i with label y_i, one can calculate the value of $\epsilon = \epsilon_{opt} = \exp\left\{-\frac{(\mathbf{w}^\top \mu_i^{'} - b)^2}{2(\mathbf{w}^\top \mathbf{L}_i^2 \mathbf{\Sigma}_i^2 \mathbf{w})}\right\}$ for which the uncertainty ellipsoid $\mathcal{E}(\mu_i^{'}, \mathbf{S}_i, \kappa)$ touches the discriminating hyperplane, $\mathbf{w}^\top \mathbf{x} - b = 0$. Additionally, if true label, y_i, of the example is same as the predicted label (y_i^{pr}), then the

Table 1. Table summarizing ways of representing interval-valued uncertainty in examples and corresponding label prediction methodologies

S.No.	Form of examples	Labeling Methodology
1	$\mathcal{R}(\mathbf{a}_i, \mathbf{b}_i)$ are given	$y_i^{pr} \equiv \text{sign}(\mathbf{w}^\top \mathbf{m}_i - b)$
2	$\mathcal{R}(\mathbf{a}_i, \mathbf{b}_i), \mu_i^{'}$ are given	$y_i^{pr} \equiv \text{sign}(\mathbf{w}^\top \mu_i^{'} - b)$
3	Set of replicates $\mathbf{r}_{i1}, \mathbf{r}_{i2}, \ldots$	y_i^{pr} is label of majority of $\mathbf{r}_{ij}$. Label of $\mathbf{r}_{ij}$ is $\text{sign}(\mathbf{w}^\top \mathbf{r}_{ij} - b)$

proof of theorem 1 shows that the true probability of misclassification of the test example is less than or equal to ϵ_{opt}. This leads to the following definition of error on each test example:

$$\mathbf{OptErr}_i = \begin{cases} 1 & \text{if } y_i \neq y_i^{pr} \\ \epsilon_{opt} & \text{if } y_i = y_i^{pr} \text{ and } \exists\ \mathbf{x} \in \mathcal{R}(\mathbf{a}_i, \mathbf{b}_i) \ni y_i(\mathbf{w}^\top \mathbf{x} - b) < 0 \\ 0 & y_i(\mathbf{w}^\top \mathbf{x} - b) \geq 0\ \forall\ \mathbf{x} \in \mathcal{R}(\mathbf{a}_i, \mathbf{b}_i) \end{cases} \tag{16}$$

The overall error, **OptErr**, can be calculated as percentage of $\mathbf{OptErr}_i$ over all test examples:

$$\mathbf{OptErr} = \frac{\sum_i \mathbf{OptErr}_i}{\#\ test\ examples} \times 100 \tag{17}$$

Note that, both **NomErr** and **OptErr** can be estimated for any hyperplane classifier and are not specific to the proposed classifier. Experimental results show that the proposed classifier achieves lower **NomErr** and **OptErr** when compared to existing interval data classification methods.

3 Numerical Experiments

In this section, experimental results which compare performance of the proposed interval data classifier (12) (denoted by **IC-MBH**) and the maximum-margin classifier which utilizes bounding hyper-rectangle information alone [3] (denoted by **IC-BH**):

$$\begin{aligned} \min_{\mathbf{w},b,\xi_i} \quad & \tfrac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^m \xi_i \\ \text{s.t. } & y_i(\mathbf{w}^\top \mathbf{m}_i - b) \geq 1 - \xi_i + \|\mathbf{L}_i\mathbf{w}\|_1,\ \xi_i \geq 0,\ i = 1, \ldots, m \end{aligned} \tag{18}$$

are presented. Note that the only difference between (18) and the one proposed in [3] is minimization of $\|\mathbf{w}\|_2$ in the objective rather than minimization of $\|\mathbf{w}\|_1$, which implies maximum-margin classification rather than sparse classification. We have done this in order to achieve a fair comparison of the methodologies. Traditional classifiers like SVMs cannot handle interval-valued data. However, in cases where the means of the uncertain examples are known or in cases where uncertainty is represented using replicates (e.g. form 2 and 3 in table 1 respectively), SVMs can be trained by considering mean of each example as a training datapoint or by considering each replicate as a training datapoint. Henceforth, let these classifiers be denoted by **IC-M** and **IC-R** respectively. Hence, wherever applicable, we compare the performance of the proposed classifier with SVM based methods also.

Experiments were performed on synthetic datasets and two real-world datasets: micro-array data[2] [1] and Wisconsin Diagnostic Breast Cancer (WDBC) dataset[3]. Synthetic datasets were generated using the following methodology: a) nominal

[2] Available at `http://www.ncbi.nlm.nih.gov/geo/` with accession number GSE2187.
[3] Available at `http://mlearn.ics.uci.edu/MLSummary.html`

(true) examples were generated using Gaussian mixture models b) uncertainty was introduced into each nominal point using standard finite-supported distributions (whose parameters are chosen randomly) c) replicates for each nominal example were produced by sampling the chosen noise distribution. The synthetic datasets are named using dimension of the dataset and are subscripted with the distribution used for generating replicates (e.g. synthetic data of dimensionality n with Uniform, truncated skew-Normal and truncated Beta noise distributions are denoted by $\mathbf{n_U}$, $\mathbf{n_{SN}}$ and $\mathbf{n}_\beta$ respectively). In each case, a synthetic test dataset was also generated independently.

The micro-array dataset defines a 4 category classification task, namely that of identifying four kinds of drugs: Azoles ($\mathcal{A}$), Fibrates ($\mathcal{F}$), Statins ($\mathcal{S}$) and Toxicants ($\mathcal{T}$). Instead of handling a multi-class problem we have defined six binary classification tasks using "one versus one" scheme. As a preprocessing step we have reduced the dimension of the problem to 200 by feature selection using Fisher score. In case of both synthetic and micro-array data, the means and bounding hyper-rectangles were estimated from the replicates provided for each training example.

The task of WDBC is to classify "benign" and "malignant" tumours based on 10 features computed from tumour cell nuclei. However, since the measurements are not the same over all tumour cells, the mean, standard-error and maximum values of the 10 features are provided. From this information the bounding hyper-rectangles and means for each training example are estimated.

In section 3.1, we compare classification margins ($2/\|\mathbf{w}\|_2$) achieved by **IC-BH** and **IC-MBH**, which represent state-of-the-art and the proposed interval data classifier respectively. The key results of the paper are presented in section 3.2. These results compare the **NomErr** (15) and **OptErr** (17) obtained with various classifiers.

3.1 Comparison of Margin

In this section, the margins ($2/\|\mathbf{w}\|_2$) achieved by **IC-BH** and **IC-MBH** at a fixed value of the C parameter are compared. Figure 2 summarizes the results.

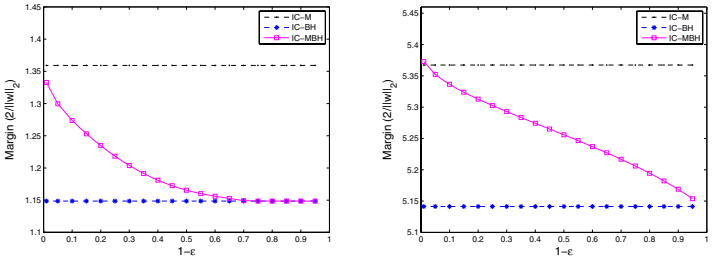


Fig. 2. Figure comparing margins achieved by **IC-M, IC-BH** and **IC-MBH** at various ϵ values ($\mathbf{2_U}$ on left and $\mathbf{10_U}$ on right)

Note that, at all values of ϵ, the classification margin with **IC-MBH** is higher than that with **IC-BH**. Also, as the value of ϵ or dimensionality of the data increases, difference in the margins achieved by **IC-BH** and **IC-MBH** also increases. The explanation for this is clear from the geometric interpretation presented in section 2.2. According to Structural Risk minimization principle of Vapnik [6], higher margin implies better generalization. Hence the proposed classifier is expected to achieve good generalization for interval data. As a baseline for comparison, the margin achieved by the SVM trained using means of the examples, **IC-M**, is also shown in the figure. Since **IC-M** does not take into account the interval uncertainty and assumes the mean to be the "true" training example, it always achieves higher margin than **IC-BH, IC-MBH**. The trend shown in figure 2 remained the same for higher dimensions and with different noise distributions ($\mathbf{n_{SN}}, \mathbf{n}_\beta$).

3.2 Comparison of Generalization Error

This section presents results which compare the performance of **IC-M, IC-R, IC-BH** and **IC-MBH** when evaluated using the error measures **NomErr** (15) and **OptErr** (17). Experiments were done on the synthetic and real-world datasets described in section 3. In all cases, the hyper-parameters (C and/or ϵ) for each classifier were tuned using a 3-fold cross-validation procedure. The results are summarized in table 2. In case of synthetic datasets, the reported values represent the mean testset error achieved with the tuned hyper-parameters when trained with 10 different training sets each generated from the same synthetic data template. In case of the real-world datasets, the values represent cross-validation error with tuned hyper-parameters averaged over three cross-validation experiments. Hence the error values reported in the table represent a good estimate of the generalization error of the respective classifiers. Clearly, **NomErr** and **OptErr** are least for **IC-MBH**; confirming that **IC-MBH** achieves good generalization for interval data. Moreover, in case of many

Table 2. Table comparing **NomErr** and **OptErr** obtained with **IC-M, IC-R, IC-BH** and **IC-MBH**

Dataset	IC-M		IC-R		IC-BH		IC-MBH	
	NomErr	**OptErr**	**NomErr**	**OptErr**	**NomErr**	**OptErr**	**NomErr**	**OptErr**
$\mathbf{10_U}$	32.07	59.90	44.80	65.70	51.05	53.62	**20.36**	**52.68**
$\mathbf{10}_\beta$	46.46	54.78	48.02	53.52	46.67	49.50	**46.18**	**49.38**
$\mathcal{A}$ vs. $\mathcal{F}$	00.75	46.47	00.08	46.41	55.29	58.14	**00.07**	**39.68**
$\mathcal{A}$ vs. $\mathcal{S}$	09.02	64.64	08.65	68.56	61.69	61.69	**06.10**	**39.63**
$\mathcal{A}$ vs. $\mathcal{T}$	12.92	73.88	**07.92**	81.16	58.33	58.33	11.25	**40.84**
$\mathcal{F}$ vs. $\mathcal{S}$	01.03	34.86	00.95	38.73	28.21	49.25	**00.05**	**27.40**
$\mathcal{F}$ vs. $\mathcal{T}$	06.55	55.02	05.81	58.25	51.19	60.04	**05.28**	**35.07**
$\mathcal{S}$ vs. $\mathcal{T}$	10.95	64.71	**05.00**	70.76	69.29	69.29	**05.00**	**30.71**
WDBC	55.67	**37.26**	×	×	**37.26**	45.82	47.04	45.84

datasets, the proposed classifier outperforms the existing classifiers in terms of the **OptErr** error measure.

4 Conclusions

This paper presents a novel maximum-margin classifier which achieves good generalization on data having interval-valued uncertainty. The key idea was to employ chance-constraints in order to handle uncertainty. The main contribution was to derive a tractable SOCP formulation, which is a safe approximation of the resulting CCP, using Bernstein schemes. The formulation ensures that the probability of misclassification on interval data is less than a user specified upper-bound (ϵ). Also, the geometric interpretation shows that the classifier views each training example as the region of intersection of its bounding hyper-rectangle and an ellipsoid centered at its mean.

The proposed classifier is robust to interval-valued uncertainty and is also not overly-conservative. As shown in the paper, this amounts to achieving higher classification margins and in turn better generalization than methods which employ the bounding hyper-rectangle information alone. As the results showed, the average error with the proposed classifier, in case of many synthetic and real-world datasets, is less than half of that with the existing methods.

The Bernstein relaxations schemes presented in this paper not only aid in approximating the original CCP as a convex program, but also open avenues for efficient approximations of other CCP-based learning formulations (e.g. [9] and its variants). By employing rich partial information, the Bernstein schemes lead to less conservative relaxations. Hence exploitation of the Bernstein schemes in the context of learning is a good direction for research.

Acknowledgments

SB and CB are supported by a Yahoo! faculty grant.

A Computation of $\boldsymbol{\sigma(\mu)}$

In this section, we present details of the numerical procedure for computing $\sigma(\mu)$. Consider the following claim:

Claim. Let $\sigma(\mu)$ be as defined in (4). Then, $\sqrt{1-\mu^2} \leq \sigma(\mu) \leq 1$.

Proof. Recalling the definition of $\sigma(\mu)$, we have:

$$\sigma(\mu) = \min \{\sigma \geq 0 \mid f(t;\sigma,\mu) \geq 0,\ \forall\ t\ \in\ \mathbb{R}\}$$

where $f(t;\sigma,\mu) \equiv \frac{\sigma^2}{2}t^2 + \mu t - \log(\cosh(t) + \mu \sinh(t))$. Let $f^{'}(t;\sigma,\mu) = g_1(t) - g_2(t)$ where $g_1(t) \equiv \sigma^2 t + \mu$ and $g_2(t) \equiv \frac{\sinh(t)+\mu\cosh(t)}{\cosh(t)+\mu\sinh(t)}$. Now, if $g_1^{'}(0) < g_2^{'}(0)$, then there exists a neighbourhood around $t = 0$ where $f^{'}(t) < 0$ (since $f^{'}(0) =$

Table 3. Table showing values of σ as a function of $\mu \in [0, 1]$ at 20 equal increments

1.0000	0.9995	0.9979	0.9958	0.9914	0.9876	0.9827	0.9745	0.9627	0.9560
0.9481	0.9356	0.9176	0.8927	0.8686	0.8538	0.8279	0.7812	0.6986	0.0000

0). Also in this neighbourhood $f(t) < 0$ because $f(0) = 0$. Thus $g_1^{'}(0) \geq g_2^{'}(0)$ is a necessary condition for $f \geq 0$. In other words, $\sigma(\mu) \geq \sqrt{1-\mu^2}$. Also, from proof of lemma 2 we have that $\sigma(\mu) \leq 1$. This completes the proof. □

Note that, the function f strictly increases with the value of σ and by claim A we have that $\sqrt{1-\mu^2} \leq \sigma(\mu) \leq 1$. Thus one can have a simple binary search algorithm for computing σ. The algorithm starts with $\sigma_0^l \equiv \sqrt{1-\mu^2}$ and $\sigma_0^u \equiv 1$. At every iteration, $i \geq 1$, $\sigma_i \equiv \frac{\sigma_{i-1}^l + \sigma_{i-1}^u}{2}$ and it is checked whether $f_i^{min} \equiv (\min_t f(t; \sigma_i, \mu)) \geq 0$. If $f_i^{min} \geq 0$, then $\sigma_i^u \equiv \sigma_i$, else $\sigma_i^l \equiv \sigma_i$. This is repeated until a relevant stopping criteria is met. Also, as the proof of claim A suggests, for fixed values of σ, μ, the function f has only one minimum wrt. t (this is because $g_2(t)$ is concave above $t = t^*$ and convex below $t = t^*$). Hence checking whether $f_i^{min} \geq 0$ for a fixed value σ_i is also easy. The values of σ as a function of $\mu \in [0, 1]$ are shown in table 3. Since the function $\sigma(\mu)$ is symmetric wrt. μ, we have $\sigma(\mu) = \sigma(-\mu)$.

References

1. Natsoulis, G., Ghaoui, L.E., Lanckriet, G.R.G., Tolley, A.M., Leroy, F., Dunlea, S., Eynon, B.P., Pearson, C.I., Tugendreich, S., Jarnagin, K.: Classification of a Large Microarray Data Set: Algorithm Comparison and Analysis of Drug Signatures. Genome Research 15, 724–736 (2005)
2. Silva, F.C.D., de A.T. de Carvalho, F., de Souza, R.M.C.R., Silva, J.Q.: A Modal Symbolic Classifier for Interval Data. In: King, I., Wang, J., Chan, L.-W., Wang, D. (eds.) ICONIP 2006. LNCS, vol. 4233, pp. 50–59. Springer, Heidelberg (2006)
3. Ghaoui, L.E., Lanckriet, G.R.G., Natsoulis, G.: Robust Classification with Interval Data. Technical Report UCB/CSD-03-1279, Computer Science Division, University of California, Berkeley (2003)
4. Nemirovski, A., Shapiro, A.: Convex Approximations of Chance Constrained Programs. SIAM Journal of Optimization 17(4), 969–996 (2006)
5. Sturm, J.F.: Using SeDuMi 1.02, A MATLAB Toolbox for Optimization over Symmetric Cones. Optimization Methods and Software 11–12, 625–653 (1999)
6. Vapnik, V.: Statistical Learning Theory. John Wiley and Sons, New York (1998)
7. Ben-Tal, A., Nemirovski, A.: Selected Topics in Robust Convex Optimization. Mathematical Programming 112(1) (2007)
8. Rockafellar, R.T.: Convex Analysis. Princeton University Press, Princeton (1970)
9. Lanckriet, G.R., Ghaoui, L.E., Bhattacharyya, C., Jordan, M.I.: A Robust Minimax Approach to Classification. JMLR 3, 555–582 (2003)

Negative Encoding Length as a Subjective Interestingness Measure for Groups of Rules

Einoshin Suzuki

Department of Informatics, ISEE, Kyushu University
suzuki@i.kyushu-u.ac.jp

Abstract. We propose an interestingness measure for groups of classification rules which are mutually related based on the Minimum Description Length Principle. Unlike conventional methods, our interestingness measure is based on a theoretical background, has no parameter, is applicable to a group of any number of rules, and can exploit an initial hypothesis. We have integrated the interestingness measure with practical heuristic search and built a rule-group discovery method CLARDEM (Classification Rule Discovery method based on an Extended-Mdlp). Extensive experiments using both real and artificial data confirm that CLARDEM can discover the correct concept from a small noisy data set and an approximate initial concept with high "discovery accuracy".

1 Introduction

The most serious problem in rule discovery would be the interestingness problem: typically many rules are discovered but most of them are uninteresting [4,13]. Solutions for this problem can be classified into the objective approach [11,12,13], which uses only data as input, and the subjective approach [3,4], which uses also user-supplied information in addition to data. In both approaches, an interestingness measure [3,11,12,13], which is a function for estimating the degree of the interestingness of a rule, is actively studied.

Despite the numerous studies on interestingness measures, few of them have a theoretical background, are parameter-free, can discover a group of rules which are mutually related, and can exploit an initial hypothesis. We attribute the reasons to the subjective nature of interestingness and the high time complexity. [3,4,11,12] are exceptions for some of them but none satisfy these four conditions. Moreover, as far as we know, no study has ever made a systematic investigation on the discovered rules under noisy data and incorrect user-supplied information.

The Minimum Description Length Principle (MDLP) is a principle that the best hypothesis that can be inferred from data is the one that has the shortest "(code length of the hypothesis) + (code length of the data using the hypothesis)" [5,8,9,15]. The MDLP is based on a solid theoretical framework, has a clear interpretation, is robust to noise, and requires no parameter specification. In association rule discovery, the MDLP has been applied to the problem of discovering frequent itemsets [11]. However, the discovered patterns are still large in number and are unrelated. Moreover, the method belongs to the objective

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 220–231, 2009.
© Springer-Verlag Berlin Heidelberg 2009

approach thus a theoretical framework which can be integrated for exploiting user-supplied information is unknown.

We restrict our attention to the classification rule [4], which has a class label in its conclusion and has been well-studied due to its importance. It is not obvious how to apply the MDLP for classification to the classification-rule discovery problem since a classifier can be applied to any example unlike a typical group of rules. Moreover, the standard MDLP cannot exploit an initial hypothesis and the MDLP extended for this purpose [14] has problems such as a redundancy in its encoding method. In summary, the MDLP has problems to be used in developing a method with a theoretical background for discovering a group of rules which are mutually related by exploiting user-supplied information. To resolve these problems we formalize the discovery problem of interesting classification rules as an estimation problem of a partial decision list, extend the MDLP for classification so that it can exploit an initial hypothesis, invent an encoding method, and use the negative encoding length as our interestingness measure.

2 Preliminaries

2.1 MDL for Classification

A data set D consists of n examples $d_1, d_2, \ldots, d_n$. Each example d_i is described with m attributes $a_1, a_2, \ldots, a_m$ as an attribute value vector $(v_{i1}, v_{i2}, \ldots, v_{im})$ and belongs to one of M classes, of which labels are represented by $c_1, c_2, \ldots, c_M$. A classifier is a function which outputs a class label given an attribute value vector. We call the process of learning a classifier from D classification.

As a principle for preferring a classifier in classification, the MDLP states that the best classifier T_{MDL} is given as follows [5,8,9,15].

$$T_{\mathrm{MDL}} \equiv \arg\min_{T} \left(-\log P(T) - \log P(D|T)\right) \tag{1}$$

where $P(T)$ and $P(D|T)$ represent the probability that T occurs and the conditional probability that D occurs given T, respectively. Consider the problem of encoding T as a binary string. According to the coding theory [10], the length of the code string for T using an optimally efficient code is $-\log P(T)$. Similarly, $-\log P(D|T)$ may be regarded as the length of the code string for D encoded using T. In the MDLP for classification, these code lengths are calculated in a problem where the receiver has D except for the class labels. The sender first sends T then the class labels of examples in D using T.

It is straightforward to show T_{MDL} coincides with the maximum a posteriori hypothesis. The MDLP can be interpreted as assigning priors to theories based on a compact coding: $P(T)$ is defined by the encoding method for $-\log P(T)$.

2.2 Preliminaries for Encoding

Firstly we consider a problem of sending a binary string of length x which consists of y binary 1s and $(x - y)$ binary 0s. A common method first sends the number

y of binary 1s with code length $\log(x+1)$ then specifies the positions of binary 1s [8,15]. The required code length is denoted with $\Theta(x,y)$.

$$\Theta(x,y) \equiv \log(x+1) + \log \binom{x}{y}$$

For example, "1110010" is sent with a code length $\Theta(7,4) = 8.13$ bits. Note that we do not have to generate the binary message for our purpose.

If we know that $y > 0$, the number y of binary 1s can be sent with code length $\log x$. The required code length in this case is denoted with $\Theta_0(x,y)$.

$$\Theta_0(x,y) \equiv \log x + \log \binom{x}{y}$$

Likewise, we consider a problem of sending a string of length x described with M symbols where the ith symbol occurs x_i times. The sender first sends the numbers $x_1, x_2, \ldots, x_{M-1}$ then specifies the positions except that of the last symbol. We denote the required code length with $H(x,(x_1,x_2,\ldots,x_M),M)$.

$$H(x,(x_1,x_2,\ldots,x_M),M) \equiv \sum_{i=1}^{M-1} \log \left(x+1-\sum_{j=1}^{i-1} x_j \right) + \log \left(\frac{x!}{x_1!x_2!\ldots x_M!} \right) \tag{2}$$

For example, "AACBBAB" is sent in $H(7,(3,3,1),3) = 12.45$ bits.

Lastly we consider a problem of sending a positive integer x under the assumption that $x = y$ is most likely and the occurrence probability $P(i)$ of $x = i$ is given by $P(y)(1/2)^{|y-i|}$. This setting may be interpreted as the length for sending i is longer than that for sending y by $|y-i|$ bits. Since $P(1)+P(2)+\cdots = 1$, the length $-\log\left[P(y)(1/2)^{|y-x|}\right]$, which is required to send x and is denoted with $\Lambda(x,y)$, is given as follows.

$$\Lambda(x,y) \equiv \log\left[3-\left(\frac{1}{2}\right)^{y}\right] + |y-x| \tag{3}$$

2.3 Classification-Rule Discovery Problem

We call an assignment $a = v$ of a value v to an attribute a an atom. A literal is defined as either a single atom or a conjunction of multiple atoms. An example $(v_{i1}, v_{i2}, \ldots, v_{im})$ is said to satisfy a literal δ if every atom in δ is included in $\{a_1 = v_{i1}, a_2 = v_{i2}, \ldots, a_m = v_{im}\}$. We define a distribution rule r as $r \equiv \rho(r) \to (P_1, P_2, \ldots, P_M)$, where its premise $\rho(r)$ is a literal and its conclusion is a probabilistic distribution $P_1, P_2, \ldots, P_M$ over the classes $1, 2, \ldots, M$.

A partial decision list T, which may be interpreted as a decision list without the default class label, consists of μ distribution rules $r_1, r_2, \ldots, r_\mu$ i.e. $T \equiv r_1, r_2, \ldots, r_\mu$. For a partial decision list $r_1, r_2, \ldots, r_\mu$, a rule r_j is said to cover an example e iff. (if and only if) e does not satisfy $\rho(r_i)$ $(i = 1, 2, \ldots, j-1)$ but satisfies $\rho(r_j)$. We believe that a partial decision list is adequate as the

representation of a hypothesis as it represents a group of rules which are mutually related in a separate-and-conquer manner. The set of examples each of which is satisfied by a distribution rule in a partial decision list T is denoted with $D(T)$.

A null partial decision list B consists of ν distribution rules $b_1, b_2, \ldots, b_\nu$ without conclusions i.e. $B \equiv b_1, b_2, \ldots, b_\nu$. We believe that a null partial decision list is adequate as the representation of an initial hypothesis since it is easier to be obtained from domain experts or textbooks. A partial decision list (or a null partial decision list) which satisfies $\mu = 0$ (or $\nu = 0$) is denoted with $\emptyset$ and is called a null hypothesis.

Partial classification [1] has been mainly studied in the context of decision making. As the objective of a data mining process is not usually restricted to prediction, neither a utility function nor a cost function is adequate for evaluating the goodness of our partial decision list T. In a domain where there is a ground truth i.e. for our case a "correct" partial decision list T_{true}, we define, as an evaluation index, the discovery accuracy $E(\mathcal{M})$ of a rule-group discovery method $\mathcal{M}$ as $E(\mathcal{M}) \equiv \frac{\Upsilon(T_{\mathcal{M}}=T_{\text{true}})}{\Upsilon}$, where Υ is the total number of different trials and $\Upsilon(T_M = T_{\text{true}})$ is the number of trials in each of which the hypothesis $T_{\mathcal{M}}$ returned by $\mathcal{M}$ is equivalent to T_{true}.

[Classification-rule Discovery Problem] Given a data set D and a null partial decision list B as an initial hypothesis, discover a partial decision list T. The goodness of a discovery method $\mathcal{M}$ is evaluated with its discovery accuracy $E(\mathcal{M})$ if a correct partial decision list T_{true} is known.

3 Our Method CLARDEM

3.1 Incorporating Background Knowledge

The MDLP for classification (1) cannot handle an initial hypothesis B thus cannot be applied to our discovery problem directly. We have extended the original MDLP for classification so that T is inferred from D and B. The best hypothesis T_{EMDL} chosen by our extended MDLP is stated as follows.

$$T_{\text{EMDL}} \equiv \arg\min_T \left(-\log P(T) - \log P(D|T) - \log P(B|T)\right)$$

A unique feature of our method is the term $-\log P(B|T)$, which allows us to consider B rigorously. We calculate the code length $L(T)$ in a problem setting where the receiver has D except for the class labels. The sender first sends T, then the class labels of examples in D using T, and B using T.

$$L(T) \equiv -\log P(T) - \log P(D|T) - \log P(B|T) \tag{4}$$

Note that the smaller $L(T)$ is the more interesting T is thus the negative code length $-L(T)$ can be considered as our interestingness measure.

We assume that B and D are independent because B is typically given by the user and not inferred from D. In this case, T_{EMDL} is shown to coincide with the maximum a posteriori hypothesis i.e. $T_{\text{EMDL}} = \arg\min_T(-\log P(T) - \log P(D, B|T)) = \arg\max_T P(T|D, B)$.

3.2 Encoding Method

Here we propose how to calculate (4). A hypothesis T is sent by first sending the number μ of distribution rules in T then the premise $\rho(r_i)$ of each r_i in T. The conclusion of r_i is sent as the class labels in $D(T)$ in the message of $-\log P(D|T)$. μ is sent with code length $\Lambda(\mu, 0)$ given by (3). $\rho(r_i)$ is sent by specifying the attributes in the premise and their values. Let $\kappa(a_j)$ and $|x|$ represent the number of possible values of an attribute a_j and the number of atoms in a literal x.

$$-\log P(T) = \Lambda(\mu, 0) + \sum_{i=1}^{\mu} \left[\Theta_0\left(m, |\rho(r_i)|\right) + \sum_{a_j \text{ in } \rho(r_i)} \log \kappa(a_j) \right]$$

The initial hypothesis B is sent by first sending the number ν of distribution rules without conclusions in B then each distribution rule b_i without conclusions using T. The former is sent with code length $\Lambda(\nu, \mu)$. We say that a conjunction x of atoms is more general than a conjunction y of atoms iff. each atom in x is found in y and y has at least one atom which does not exist in x, and denote with $x \sqsupset y$. For instance, $a1 = v1, a3 = v3 \sqsupset a1 = v1, a2 = v2, a3 = v3$, where $a1, a2, a3$ are attributes and $v1, v2, v3$ are their values. For the latter, we consider four distinctive cases: 1. $\rho(b_i) = \rho(r_i)$, 2. $\rho(b_i) \sqsupset \rho(r_i)$, 3. $\rho(r_i) \sqsupset \rho(b_i)$, and 4. other cases. The sender sends ν flags for indicating the corresponding case of (b_i, r_i) using (2). For the cases 2. and 3., the attributes in $\rho(r_i)$ are used to specify those in $\rho(b_i)$. Below s.t. represents "such that" and $\nu_1(B,T),\ \nu_2(B,T),\ \nu_3(B,T),\ \nu_4(B,T)$ are the respective numbers of the four cases.

$$\begin{aligned}
&-\log P(B|T) \\
&= \Lambda(\nu, \mu) + H(\nu, (\nu_1(B,T), \nu_2(B,T), \nu_3(B,T), \nu_4(B,T)), 4) \\
&\quad + \sum_{i \text{ s.t. } \rho(b_i) = \rho(r_i)} 0 + \sum_{i \text{ s.t. } \rho(b_i) \sqsupset \rho(r_i)} \Theta_0\left(|\rho(r_i)|, |\rho(r_i)| - |\rho(b_i)|\right) \\
&\quad + \sum_{i \text{ s.t. } \rho(r_i) \sqsupset \rho(b_i)} \left[\Theta_0\left(m - |\rho(r_i)|, |\rho(b_i)| - |\rho(r_i)|\right) + \sum_{\substack{a_j \text{ in } \rho(b_i) \\ \text{but not in } \rho(r_i)}} \log \kappa(a_j) \right] \\
&\quad + \sum_{i \text{ for othercases}} \left[\Theta_0\left(m, |\rho(b_i)|\right) + \sum_{a_j \text{ in } \rho(b_i)} \log \kappa(a_j) \right]
\end{aligned}$$

The class labels in D is sent using T: they are decomposed into those covered by each r_i and those in $D \setminus D(T)$. For the former, we use (2) with a small modification to avoid inconveniences[1]. Let $n(T,i), n_j(T,i)$ be the number of examples covered by the i-th rule in T and the number of examples of class j covered by the i-th rule in T, respectively. Let $j_{\mathrm{NTH}(T,i,d)}$ be the d-th most

[1] For instance, $H(8,(4,2,1,1),4) \neq H(8,(1,1,2,4),4)$ and $H(8,(3,3,1,1),4) < H(8,(1,1,2,4),4)$.

numerous class for its number $n_{j_{\mathrm{NTH}(T,i,d)}}(T,i)$ of examples covered by the i-th rule in T so $n_{j_{\mathrm{NTH}(T,i,1)}}(T,i) \geq n_{j_{\mathrm{NTH}(T,i,2)}}(T,i) \geq \cdots \geq n_{j_{\mathrm{NTH}(T,i,M)}}(T,i)$. We assume that the message which specifies the new order of the class labels $c_{j_{\mathrm{NTH}(T,i,1)}}, c_{j_{\mathrm{NTH}(T,i,2)}}, \ldots, c_{j_{\mathrm{NTH}(T,i,M)}}$ has a fixed size and omit counting its code length for simplicity. For the latter, we assign the code length $-\log M$, which is the longest code length for an event with M possible states, to each class label. This assignment represents the indifference of a partial decision list to its uncovered examples. It helps us avoid obtaining a counterintuitive hypothesis of which rules try to "get rid of" examples to have a $D \setminus D(T)$ which is nearly homogeneous with the majority class. We omit the reason due to lack of space.

$$-\log P(D|T) = \sum_{i=1}^{\mu} H(n(T,i), (n_{j_{\mathrm{NTH}(T,i,1)}}(T,i), \ldots, n_{j_{\mathrm{NTH}(T,i,M)}}(T,i)), M) + \sum_{e \notin D(T)} \log M$$

3.3 Desirable Properties

Studying (4) for two similar hypotheses T and T' reveals that (4) exhibits attractive properties. This fact is important because it differentiates (4) from many empirical interestingness measures which are designed to exhibit attractive properties. Due to space constraint, we just show the following without proof.

Theorem 1. *Let $\mu(T_0)$ be the number of distribution rules in a hypothesis T_0. Let two distinct hypotheses T and T' satisfy*

$$-\log P(T) - \log P(B|T) = -\log P(T') - \log P(B|T')$$
$$\mu(T) = \mu(T').$$

If T is more accurate than T' and covers the same number of examples for each rule, i.e.

$$\forall i \; n_{j_{\mathrm{NTH}(T,i,1)}}(T,i) > n_{j_{\mathrm{NTH}(T',i,1)}}(T',i)$$
$$\forall i \forall d \neq 1 \; n_{j_{\mathrm{NTH}(T,i,d)}}(T,i) \leq n_{j_{\mathrm{NTH}(T',i,d)}}(T',i)$$
$$\forall i \; n(T,i) = n(T',i)$$

then T is judged better with our interestingness measure i.e. $L(T) < L(T')$.

3.4 Practical Heuristic Search

Since an exhaustive search for all possible partial decision lists is prohibitive due to its time-inefficiency, CLARDEM applies three heuristic search methods then outputs the partial decision list with the minimum code length. The first two methods are hill climbing from B and $\emptyset$ where a step is an addition/deletion of a rule/atom, where an added rule has a single atom in its premise.

Separate-and-conquer is frequently used for learning a rule-based classifier (e.g. [7]). Here we use a modified version which never returns a hypothesis with a longer code length. It is a double-loop algorithm which searches rules with sequential covering in its outer loop. It searches conjunctions of atoms as premises of the rule with greedy search which checks up to conjunctions of m atoms. Below we show its pseudo-code, where $r_\mu(T'')$ represents the μ-th rule of T''.

```
algorithm Separate-and-conquer
  T = ∅, min = ∞, μ = 1, T' = T
  do // outer loop
    SacInnerLoop(μ, min, T, T', f), μ = μ + 1
  while(f == TRUE) // outer loop
  output T
procedure SacInnerLoop(μ, min, T, T', f)
  f = FALSE, T'' = T', ρ(r_μ(T'')) = TRUE
  for π = 1,...,m // decide the π-th atom in the μ-th rule
    min' = ∞
    foreach attribute a_i
      If a_i does not exist in ρ(r_μ(T''))
        foreach value v_ij of a_i
          ρ(r_μ(T'')) = ρ(r_μ(T'')) ∧ (a_i = v_ij)
          If L(T'') < min' // update the best hypothesis T' with μ rules
            min' = L(T''), T' = T''
          If L(T'') < min // update the best hypothesis T
            min = L(T''), T = T'', f = TRUE
          Delete ∧(a_i = v_ij) from ρ(r_μ(T''))
    T'' = T'
```

In the hill climbing method from B, an addition of a rule, which has a single atom in its premise, at each step takes $O(mn\kappa_{\mathrm{MAX}})$, where κ_{MAX} represents the maximum number of values that an attribute can take. A deletion of a rule at each step takes $O(n\mu_{\mathrm{MAX}})$, where μ_{MAX} represents the maximum number of rules in a hypothesis during the search. An addition of an atom at each step takes $O(mn\kappa_{\mathrm{MAX}}\mu_{\mathrm{MAX}})$. A deletion of an atom at each step takes $O(n\pi_{\mathrm{MAX}}\mu_{\mathrm{MAX}})$, where π_{MAX} represents the maximum number of atoms in a premise during the search. We assume that the number of search steps is $O(|\mu - \nu|)$, $O(\mu) = O(\nu) = O(\mu_{\mathrm{MAX}})$, and $O(\pi_{\mathrm{MAX}}) = O(1)$. Thus the time complexity is given by $O(mn\kappa_{\mathrm{MAX}}\mu^2)$. The same result holds even if the starting point is $\emptyset$. For our Separate-and-conquer, the time complexity is given by $O(m^2 n\kappa_{\mathrm{MAX}}\mu^2)$.

4 Experiments

4.1 Application to Benchmark Data Sets

We use for comparison MDL, which a method based on the MDLP. It employs $L'(T) \equiv -\log P(T) - \log P(D|T)$ as its coding length for T and is equivalent

Table 1. Characteristics of data sets and their initial hypotheses, where rec., prec., κ_{M}, π_{M} represent recall, precision, κ_{MAX}, and π_{MAX}, respectively

	data set				initial hypothesis					data set				initial hypothesis			
name	n	m	κ_{M}	M	ν	π_{M}	rec.	prec.	name	n	m	κ_{M}	M	ν	π_{M}	rec.	prec.
golf	14	5	3	3	4	2	85.7	100.0	ttt	958	10	3	2	18	5	99.0	100.0
spon.	76	46	12	12	13	4	96.1	97.3	car	1728	7	4	4	87	5	99.5	96.9
p.-op.	90	9	5	3	2	2	26.7	83.3	kr-kp	3196	37	7	2	14	12	89.8	99.7
vote	435	17	3	2	5	2	98.4	97.4	mush.	8124	22	12	2	9	3	100.0	99.8
soyb.	683	36	19	19	36	8	97.2	98.0	nurse.	12960	9	5	5	352	7	95.6	99.5

Table 2. Performance on benchmark data sets, where the best method represents the heuristic search method that returned the best result. HC1, HC2, S, and # nodes represent hill climbing from a null hypothesis, hill climbing from the initial hypothesis, the separate-and-conquer method, and the number of the searched nodes, respectively

	Discovered hypothesis					Search		
name	method	μ	π_{MAX}	recall	precision	best method	# nodes	time
golf	CLARDEM	4	2	85.7	100.0	HC2 S	180	0.01s
	MDL	0	-	0.0	-	HC1 HC2	186	0.01s
	e-Jmeasure	5	2	100.0	100.0	HC1 HC2 S	508	0.10s
sponge	CLARDEM	13	4	96.1	97.3	HC2	16626	1.38s
	MDL	5	3	100.0	69.7	HC2	37803	5.73s
	e-Jmeasure	16	4	100.0	97.4	HC2	49008	5.01s
post-operative	CLARDEM	3	2	97.8	73.9	HC2	705	0.03s
	MDL	1	1	92.2	72.3	HC1 HC2 S	629	0.03s
	e-Jmeasure	21	3	100.0	86.7	HC2	15680	0.71s
vote	CLARDEM	5	2	98.4	97.4	HC2	1776	0.09s
	MDL	2	1	88.7	98.7	HC1 HC2 S	2463	0.14s
	e-Jmeasure	9	3	100.0	98.4	HC1	13774	0.79s
soybean	CLARDEM	37	8	100.0	96.5	HC2	33156	6.26s
	MDL	18	5	100.0	84.2	HC2	160838	36.71s
	e-Jmeasure	37	9	100.0	96.2	HC2	75551	10.38s
tic-tac-toe	CLARDEM	18	5	99.0	100.0	HC2	4046	0.57s
	MDL	11	3	100.0	100.0	HC2	8200	1.00s
	e-Jmeasure	19	5	100.0	100.0	HC2	8550	1.01s
car	CLARDEM	87	5	99.8	97.9	HC2	7766	4.83s
	MDL	25	4	100.0	96.5	HC2	61026	29.14s
	e-Jmeasure	87	5	100.0	97.6	HC2	24487	11.92s
kr-vs-kp	CLARDEM	15	12	100.0	99.7	HC2	61412	26.92s
	MDL	10	12	100.0	99.7	HC2	86656	46.88s
	e-Jmeasure	16	12	100.0	99.9	HC2	193132	31.90s
mushroom	CLARDEM	9	3	100.0	99.8	HC2	20538	23.05s
	MDL	9	3	100.0	99.8	HC2	18964	21.01s
	e-Jmeasure	9	3	100.0	99.8	HC2	19083	4.42s
nursery	CLARDEM	354	7	100.0	99.5	HC2	70822	14m
	MDL	113	7	100.0	98.2	HC2	1200555	191m
	e-Jmeasure	351	7	100.0	99.7	HC2	395409	73m

Table 3. Discovered hypotheses from the vote data set

CLARDEM	e-Jmeasure
`physician = n`	`physician = y, synfuels = n,`
`-> (245, 2)/247`	`immigration = y -> (0, 76)/76`
`missile = y, synfuels = y`	`physician = n -> (245, 2)/247`
`-> (6, 1)/7`	`education = n, salvador = n -> (5, 1)/6`
`adoption = y, synfuels = y`	`missile = ? -> (0, 3)/3`
`-> (6, 2)/8`	`education = ?, adoption = ? -> (2, 0)/2`
`physician = y, synfuels = n`	`water = ? -> (0, 8)/8`
`-> (3, 135)/138`	`adoption = n -> (4, 70)/74`
`physician = y, missile = n`	`satellite = n -> (11, 0)/11`
`-> (3, 25)/28`	`adoption = y -> (0, 8)/8`

to our method for other points. We also use e-Jmeasure, which is an extension of the J-measure [12] to evaluate the goodness of T with the amount $\Gamma(T)$ of information compressed by T, where $\Gamma(T) \equiv \sum_{i=1}^{\mu} \sum_{j=1}^{M} n_j(T,i)\Big(-\log P(c_j) + \log \frac{n_j(T,i)}{n(T,i)}\Big)$. We exclude ad-hoc methods such as those based on frequent itemsets because such a method requires parameters such as support and confidence thresholds, and lacks of a theoretical background and a clear interpretation.

We first apply the three methods to ten benchmark data sets from [2] to investigate their tendencies except discovery accuracies as there is no ground truth. An initial hypothesis is generated by deleting the default class label of the decision list obtained with C4.5rules [6]. We show the characteristics of the data sets and the initial hypotheses in Table 1.

The results of the experiments and the names of the data sets are shown in Table 2. We see that the number μ of the distribution rules in the output hypothesis often increases in the order of MDL, CLARDEM, and e-Jmeasure. Theses results make sense as MDL has a preference bias for $\emptyset$, CLARDEM for the initial hypothesis, and e-Jmeasure for hypotheses which compress a large amount of information. These reasons explain that recall and precision often improve in this order, though their differences are often small.

In terms of search, we see that the method chosen as best most frequently is the hill climbing from the initial hypothesis (HC2). We attribute the reason to the excellence of C4.5rules [6]. As CLARDEM has a preference bias for the initial hypothesis, HC2 is always chosen as the best method. For computation time, CLARDEM is the fastest among the three methods for most of the cases. This result may be explained by the fact that the discovered hypotheses are often most similar to the initial hypotheses. As MDL has a preference bias for an empty hypothesis, the similarity is often the least hence it was the slowest. The number of the searched nodes gives a rough estimate of the computation time for the same data sets (e.g. nursery) thus it will be used as an index.

Due to lack of space we just show examples of the discovered hypotheses from the vote data set. MDL discovered a simple one with two rules, where class 1 and class 2 correspond to democrat and republican, respectively.

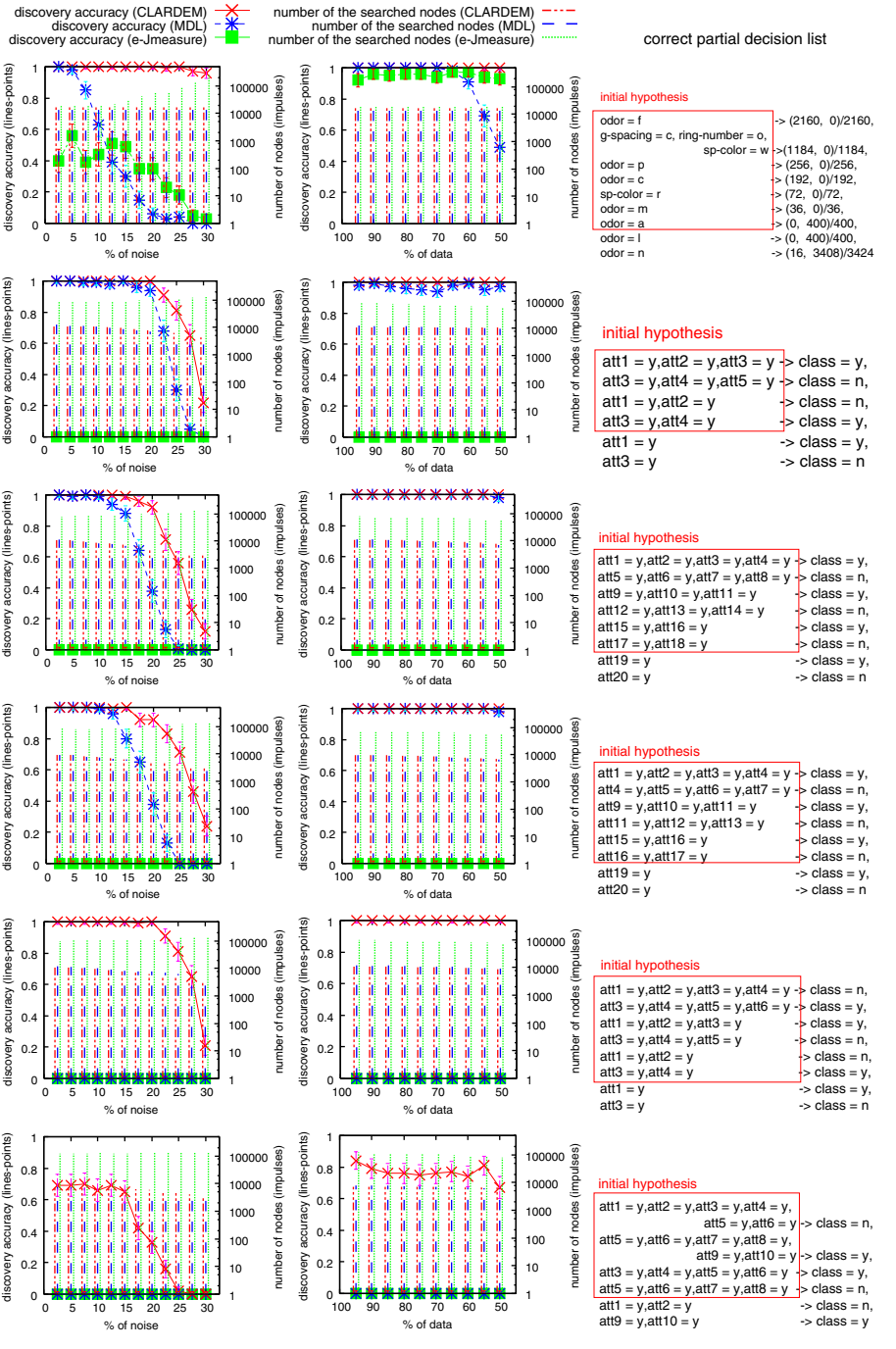


Fig. 1. Results of experiments for robustness with the mushroom data set and five artificial data sets, where class = y and class = n represent (1,0) and (0, 1), respectively

```
physician = n -> (245,  2)/247, synfuels = n -> (3,  136)/139
```

CLARDEM discovered the initial hypothesis and e-Jmeasure a complex one, which are shown in Table 3. Their preference biases explain these results.

4.2 Robustness of the Three Methods

We report the robustness of the methods to noisy data sets and incorrect initial hypotheses, where each result is an average performance on 100 data sets. For Mushroom, $T_{\rm true}$ is assumed to be the hypothesis generated by C4.5rules minus the default class label. Artificial data sets of $n = 1000$, $M = \kappa = 2$, $m = 32$ with 5, 10, ..., 30 % of random noise in the class labels are generated using hand-coded concepts. We have also generated small data sets with $n = 950, 900, \ldots, 500$ without noise. Class labels of uncovered examples are set randomly.

We consider problems of completing approximate initial hypotheses, which fits the nature of the partial decision list. The results of the experiment with correct concepts and the incorrect initial hypotheses are shown in Figure 1, where we also show $\pm$ 1.5*(standard deviations) for discovery accuracies. We see that CLARDEM is almost always the best method due to its capability of exploiting the initial hypothesis even if it is approximate. MDL is often the second method while e-Jmeasure is almost always the worst. We think e-Jmeasure always shows discovery accuracy 0 % for artificial data sets because it tries to compress the "random" parts not covered by $T_{\rm true}$. Anyway CLARDEM is also the best method for mushroom, which has no random part. CLARDEM shows high discovery accuracies even if the initial hypothesis is complex and contains strongly related rules. The numbers of the searched nodes show that CLARDEM and MDL are often one order of magnitude faster than e-Jmeasure.

5 Conclusions

Compression and learning are known to be highly related with each other [5]. The MDLP [5,9] is considered to be among the most successful works along this philosophy due to its performance and theoretical foundation. This paper has presented the first attempt to apply the MDLP and hence the philosophy of data compression to the discovery problem for a group of classification rules.

There are many evidences that the MDLP for classification is robust against noise [8,15]. Our method inherits this nice property and in addition can borrow strength from an initial hypothesis, which are shown through extensive experiments. Our method is adequate for discovering groups of rules even from a small amount of noisy data and an approximate initial hypothesis.

Acknowledgments

This work was partially supported by the grant-in-aid for scientific research on fundamental research (B) 18300047 from the Japanese Ministry of Education, Culture, Sports, Science and Technology.

References

1. Baram, Y.: Partial Classification: The Benefit of Deferred Decision. IEEE Trans. Pattern Analysis and Machine Intelligence 20(8), 769–776 (1998)
2. Blake, C., Merz, C.J., Keogh, E.: UCI Repository of Machine Learning Databases, `http://www.ics.uci.edu/~mlearn/MLRepository.html`
3. Jaroszewicz, S., Simovici, D.A.: Interestingness of Frequent Itemsets Using Bayesian Networks as Background Knowledge. In: Proc. Tenth ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD), pp. 178–186 (2004)
4. Padmanabhan, B., Tuzhilin, A.: Small is Beautiful: Discovering the Minimal Set of Unexpected Patterns. In: Proc. KDD, pp. 54–63 (2000)
5. Grünwald, P.D.: The Minimum Description Length Principle. MIT Press, Cambridge (2007)
6. Quinlan, J.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
7. Quinlan, J.R.: Learning Logical Definitions from Relations. Machine Learning 5(3), 239–266 (1990)
8. Quinlan, J.R., Rivest, R.L.: Inferring Decision Trees Using the Minimum Description Length Principle. Information and Computation 80(3), 227–248 (1989)
9. Rissanen, J.: Stochastic Complexity in Statistical Inquiry. World Scientific, Singapore (1989)
10. Shannon, C.: A Mathematical Theory of Communication. Bell System Technical Journal 27, 379–423, 623–656 (1948)
11. Siebes, A., Vreeken, J., van Leeuwen, M.: Item Sets that Compress. In: 2006 SIAM Conference on Data Mining (SDM), pp. 393–404 (2006)
12. Smyth, P., Goodman, R.M.: An Information Theoretic Approach to Rule Induction from Databases. IEEE TKDE 4(4), 301–316 (1992)
13. Tan, P.-N., Kumar, V., Srivastava, J.: Selecting the Right Interestingness Measure for Association Patterns. In: Proc. KDD, pp. 32–41 (2002)
14. Tangkitvanich, S., Shimura, M.: Learning from an Approximate Theory and Noisy Examples. In: Proc. AAAI, pp. 466–471 (1993)
15. Wallace, C.S., Patrick, J.D.: Coding Decision Trees. Machine Learning 11(1), 7–22 (1993)

The Studies of Mining Frequent Patterns Based on Frequent Pattern Tree

Show-Jane Yen[1], Yue-Shi Lee[1], Chiu-Kuang Wang[2], Jung-Wei Wu[1], and Liang-Yu Ouyang[2]

[1] Department of Computer Science & Information Engineering, Ming Chuan University, 5 De Ming Rd., Gui Shan District, Taoyuan County 333, Taiwan
[2] The Graduate Institute of Management Science , Tamkang University, 151 Ying-chuan Road, Tamsui, Taipei County, Taiwan 25137, R.O.C

Abstract. Mining frequent patterns is to discover the groups of items appearing always together excess of a user specified threshold. Many approaches have been proposed for mining frequent pattern. However, either the search space or memory space is huge, such that the performance for the previous approach degrades when the database is massive or the threshold for mining frequent patterns is low. In order to decrease the usage of memory space and speed up the mining process, we study some methods for mining frequent patterns based on frequent pattern tree. The concept of our approach is to only construct a FP-tree and traverse a subtree of the FP-tree to generate all the frequent patterns for an item without constructing any other subtrees. After traversing a subtree for an item, our approach merges and removes the subtree to reduce the FP-tree smaller and smaller. We propose four methods based on this concept and compare the four methods with the famous algorithm FP-Growth which also construct a FP-tree and recursively mines frequent patterns by building conditional FP-tree.

Keywords: Frequent Pattern, Frequent Itemset, Data Mining, Knowledge Discovery, Transaction Database.

1 Introduction

The definitions about frequent patterns are described as follows [1, 2, 3, 4, 5, 6]. A transaction database consists of a set of transactions (e.g., Table 1). A transaction is a set of items purchased by a customer at the same time. A transaction *t contains* an itemset *X* if every item in *X* is in *t*. The *support for an itemset* is defined as the ratio of the total number of transactions which contain this itemset to the total number of transactions in the database. The *support count for an itemset* is the total number of transactions which contain the itemset. A frequent pattern or a frequent itemset is an itemset whose support is no less than a certain user-specified *minimum support* threshold. An itemset of length *k* is called a *k*-itemset and a frequent itemset of length *k* a frequent *k*-itemset.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 232–241, 2009.
© Springer-Verlag Berlin Heidelberg 2009

In order to avoid generating large number of candidate itemsets and scanning the transaction database repeatedly to count supports for candidate itemsets, Han et al. proposed an efficient algorithm *FP-Growth* [4, 5], which is another innovative approach of discovering frequent patterns in a transaction database. This algorithm creates a frequent pattern tree structure which is called *FP-tree*. A frequent pattern tree (*FP-tree*) is an extended prefix-tree for storing compressed, crucial information about frequent patterns. The frequent pattern tree structure gives a good solution to store a whole transaction database. *FP-Growth* algorithm requires only two full I/O scans of the dataset to build a FP-tree in main memory and then recursively mines frequent patterns from this structure by building *conditional FP-trees*. However, this massive creation of conditional trees makes this algorithm not scalable to mine large datasets.

In this paper, we introduce a new study for discovery of frequent patterns based on the *FP-tree* [4, 5]. Our approach is different from *FP-Growth* algorithm [4, 5] which needs to recursively generate conditional *FP-trees* such that a large amount of memory space needs to be used. Our approach discovers frequent patterns by traversing the FP-tree without constructing any subtree, and applies merging techniques on the tree, which makes the *FP-tree* become smaller and smaller. By this way, our approach can dramatically condense the kernel memory space and reduce the search space without losing any frequent patterns.

2 Frequent Pattern Generation

In this section, we introduce and describe our approach in details, and give a running example (Table 1) to illustrate our algorithm TFP (mining frequent patterns by Traversing Frequent Pattern tree). TFP reduces *FP-tree* progressively and can be divided into three phases: The first phase is *construction phase* **in** which a *FP-tree* is constructed. The second phase is *frequent pattern generation phase* which generates candidate itemsets about a specific item from the constructed *FP-tree* or reduced *FPtree* and finds frequent patterns for the item. After generating frequent itemsets about the specific item from the (reduced) FP-tree, the last phase is *merge phase* which moves or merges subtrees in the (reduced) FP-tree. The last two phases need to be repeatedly executed until the reduced FP-tree only contains a node except the root node.

In the first phase, our approach TFP constructs a FP-tree by scanning original transaction database twice. The FP-tree construction method is similar to FP-Growth algorithm [4, 5]. The difference is that the root node of the FP-tree contains a header table. Each entry of the header table contains a frequent item and an empty link. After finishing the construction phase for Table 1, the constructed FP-tree is shown in Figure 1.

Table 1. A Transaction Database

TID	Items	TID	Items	TID	Items	TID	Items
1	BGDCA	2	ACHED	3	ADEBM	4	CEFBN
5	BANOP	6	BCQRG	7	BCHIG	8	LEFKA
9	BFMNO	10	CFPGR	11	BDAHI	12	DEAKL
13	MDCGO	14	CFPQJ	15	ADEFI	16	JEABD
17	BKEFC	18	CDLAB				

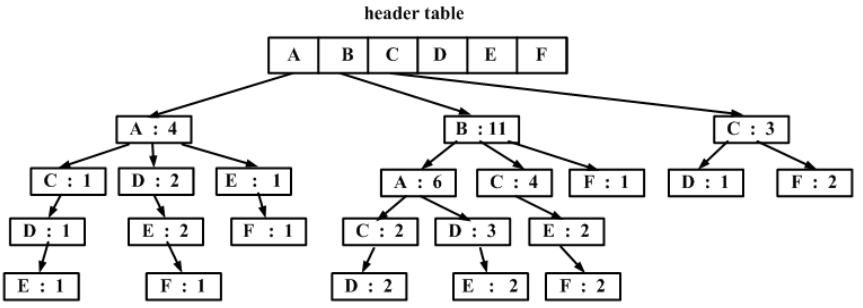


Fig. 1. The constructed FP-tree for Table 1 with minimum support 25%

2.1 Frequent Pattern Generation Phase

After constructing a *FP-tree* with a header table on the root, the next phase is to generate candidate itemsets for a specific item and find frequent itemsets about this item. The first proposed method is called Candidate Generation 1 (CG1) algorithm which generates candidate itemsets for an item. CG1 traverses FP-tree to generate candidate itemsets for a specific frequent item. The child node N_h of the root, which contains frequent item I_h with the highest support among all the frequent items, is firstly chosen as starting node to be traversed. This traversal will generate all the frequent itemsets about item I_h. CG1 applies Depth First Search (DFS) algorithm to traverse the subtree starting from the node N_h.

Meanwhile each node visited is conserved into temporary buffer for recording the passing path. When a node N_c with frequent item I_c is visited, CG1 keeps the path from starting node N_h to the current node N_c and generates all the combinations about I_h and I_c as candidate itemsets from this path. The count of each generated itemset is accumulated by adding the count of N_c to it.

After traversing the subtree with starting node N_h, the supports for all the candidate itemsets about I_h can be computed and the frequent itemsets about item I_h can be obtained. The algorithm for CG1 is shown in Algorithm CG1, in which CIS is used to store candidate itemsets with their support counts by the form: {Candidate Itemset: Support count}. next_child(N_t) is a function that systematically returns the child node of N_t. CandBuf is a local data structure for storing itemsets. Initially, N_h is a child node of the root, which contains item I_h with the highest support currently. CG1 can output the set of candidate itemsets about I_h and their support counts.

After generating all the frequent itemsets about I_h, the subtrees under the node N_h is merged to the subtrees of the root. The merging algorithm will be described in Section 2.2. Our algorithm then chooses the next frequent item with highest support, and performs candidate generation method to traverse the subtree of the reduced FP-tree starting from the node containing this frequent item. After merging and removing the subtree rooted at the node containing the item with the highest support, our algorithm continues to choose the next frequent item with highest support until the reduced FP-tree only contains a node except the root node.

Algorithm CG1: Candidate Generation 1

```
CG1(Nh, Ih, Nt, PathBuf)
{
Nc = next_child(Nt);
While(Nc != NULL)
  {
     Ic = Nc.item;
     Sc = Nc.count;
     Insert_to_CIS({Ih Ic}, Sc);
     CandBuf = all combinations for the
items in PathBuf;
          For each C in CandBuf
               Insert_to_CIS(C ∪ {IhIc},
Sc);
     CGI(Nh, Ih, Nc, PathBuf ∪ {Ic});
     Nc = next_child(Nt);
   }
}

Insert_to_CIS(Cand, Sup)
{
   If {Cand, X} ∉ CIS // X means any number
       CIS = CIS ∪ {Cand: Sup};
   Else
       Add Sup to the count of Cand, i.e.,
X, in CIS;
}
```

CG1 traverses a subtree to generate candidate itemsets about a certain frequent item and compute supports directly from (reduced) FP-tree. The smaller the support of an item is, the fewer the candidates generated by traversing the subtree starting at this item is. However, there may be many combinations generated to search and compute their supports. In order to avoid the costly generation of a large number of combinations, we propose another candidate generation method CG2 to generate fewer candidates for each traversal.

CG2 also chooses the child node N_h of the root, which contains frequent item I_h with the highest support, and traverses the subtree rooted at node N_h twice. For the first traversal, CG2 generates candidate 2-itemset I_hI_c and compute its support by accumulating the count of node N_c to it when a node N_c containing item I_c is visited. After traversing the subtree rooted at N_h, all the frequent 2-itemsets about I_h can be generated.

For the second traversal, like CG1, CG2 keeps the passing path from N_h to the current visited node N_c and generates candidate k-itemsets ($k \geq 3$) about I_h and I_c by performing the combinations of the items between N_h and N_c of this path. The count of each generated itemset is accumulated by adding the count of N_c to it. If the 2-itemset I_hI_c is not a frequent 2-itemset, the node N_c can be ignored and CG2 does nothing for this node, that is, all the candidate itemsets about I_hI_c need not be generated. After the second traversal, the supports for all the candidate k-itemsets ($k \geq 3$) about I_h can be computed and the frequent k-itemsets about item I_h can be obtained. The algorithm for CG2 is shown in Algorithm CG2, in which CIS2 is used to store candidate 2-itemsets with their support count by the form: {Candidate 2-Itemset: Support count}. CISK is used to store candidate k-itemsets ($k \geq 3$) with their support count by the form: {Candidate k-Itemset: Support count}. The other parameters are the same as CG1.

Algorithm CG2: Candidate Generation 2

```
CG2(N_h, I_h)
{
   GFreq2(N_h, I_h, N_h);
   For Each {Cand, X} ∈ CIS2
   {
      If X < min_sup
         Remove {Cand, X} from CIS2
   }
   GCandK(N_h, I_h, N_h, NULL);
}

Insert_to_CISK(Cand, Sup)
{
   If {Cand, X} ∉ CISK // X means any
number
     CISK = CISK ∪ {Cand: Sup};
   Else
     Add Sup to the count of Cand,
i.e., X, in CISK;
   }

GFreq2(N_h, I_h, Nt)
{
   Nc = next_child(Nt);
   While(Nc != NULL)
   {
        Ic = Nc.item;
        Sc = Nc.count;
        Insert_to_CIS2({IhIc}, Sc);
        GFreq2 (N_h, I_h, Nc);
        Nc = next_child(Nt);
   }
}

Insert_to_CIS2(Cand, Sup)
{
If {Cand, X} ∉ CIS2 // X means any
number
   CIS2 = CIS2 ∪ {Cand: Sup};
Else
   Add Sup to the count of Cand, i.e.,
X, in CIS2;
}
```

2.2 Merge Phase

After generating frequent itemsets about an item, our approach merges the subtrees under the node N_h containing this item to the subtrees under the root. And then the subtree rooted at N_h is removed from the (reduced) FP-tree. After merging and removing, only a subtree needs to be traversed to generate frequent itemsets for the other item. By the way, the size of the (reduced) FP-tree also can be reduced. In this section, we propose two merging algorithms Simple Merge (SM) and Compact Merge (CM) to merge these subtrees after finishing generating frequent itemsets for a certain item.

After traversing the subtree rooted at N_h and generating the frequent itmsets about the item contained in N_h, SM directly moves the subtrees rooted at the children of node N_h to the subtrees under the root. For each child node N_m containing item I_m of

node N_h, if the link of I_m in header table is null, SM sets the link to point to N_m. Otherwise, node N_m is merged to node N_m' which the link of I_m in header table points to and the count of N_m is added to the count of node N_m', since the two nodes have the same item I_m. The subtrees rooted at the children of node N_m are moved to be the subtrees under the node N_m'.

Since SM moves the subtrees rooted at the children of node N_h to become the subtrees under the child nodes of the root, the reduced *FP-tree* is still similar to the previous one. Therefore, many duplicate candidate itemsets can be generated to be counted when candidate generation algorithm is applied. We propose another merging algorithm CM to make the reduced *FP-tree* as small as possible.

The same as SM, CM traverses each subtree rooted at the children of N_h. For a subtree rooted at N_m containing item I_m, if the link of I_m in header table is null, CM sets the link to point to N_m. Otherwise, the count of N_m is added to the count of node N_m' which the link of I_m points to, since the two nodes have the same item I_m. CM continues to visit each child node of N_m and checks if there is a child node of N_m', which contains the same item with the visited node. If a child node of N_m' contains the same item I_c with the visited node N_c, then the count of N_c is added to the count of the child node and CM recursively performs the checking for the children of node N_c. Otherwise, the subtree rooted at N_c is moved to become a subtree under the node N_m' and N_c becomes a child of N_m'. And then CM continues to traverse the other subtrees rooted at the children of N_h.

3 Experimental Results

For our experiments, we evaluate the performance of our algorithms TFP(CG1+SM), TFP(CG1+CM), TFP(CG2+SM), TFP(CG2+CM) and FP-Growth algorithm [4, 5]. The source code of *FP-Growth* was downloaded from [8]. All our experiments were conducted on Intel®Pentium®4 CPU 3.00GHz with 2.99 GHz, 1.99GB memory

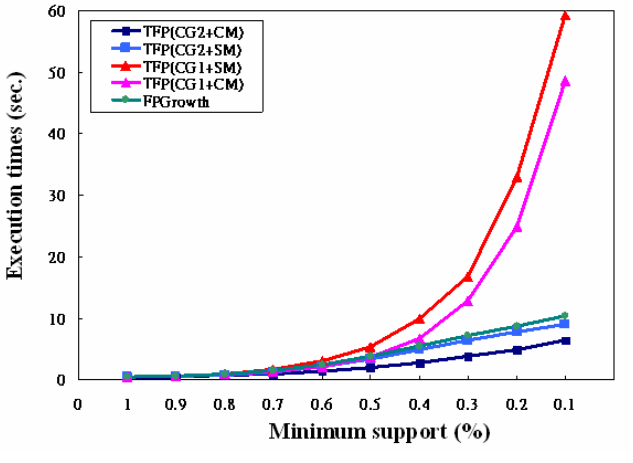


Fig. 2. Execution times on dataset T5I2D100K

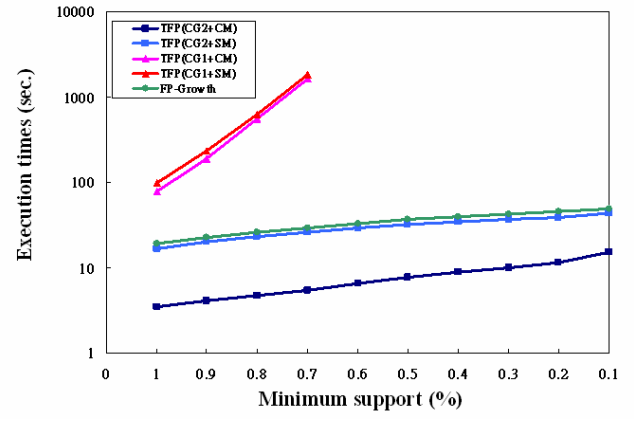


Fig. 3. Execution times on dataset T10I4D100K

using C Programming Language running Microsoft windows XP environment. Synthetic datasets are generated from IBM Data Generator [7] with parameters as follows: T-variable means average size of the transactions, I-variable means average size of the maximal potentially frequent itemsets and D-variable means the number of transactions. In the experiments, we generate two datasets: T5I2D100K, T5I2D500K, T10I4D100K and T10I4D500K to compare FP-Growth and Our algorithms.

Figures 2 and 3 show the execution times for the five algorithms on dataset T5I2D100K and T10I4D100K, respectively, with minimum support from 0.1% to 1%. From Figures 2 and 3, we can see that TFP(CG1+SM) and TFP(CG1+CM) are much worse than the other algorithms since there are many combinations generated. It is costly to search and count the large number of combinations. FP-Growth need not generate candidate itemsets, but needs to recursively generate conditional FP-trees. Moreover, the nodes with the same item need to be linked together in the constructed

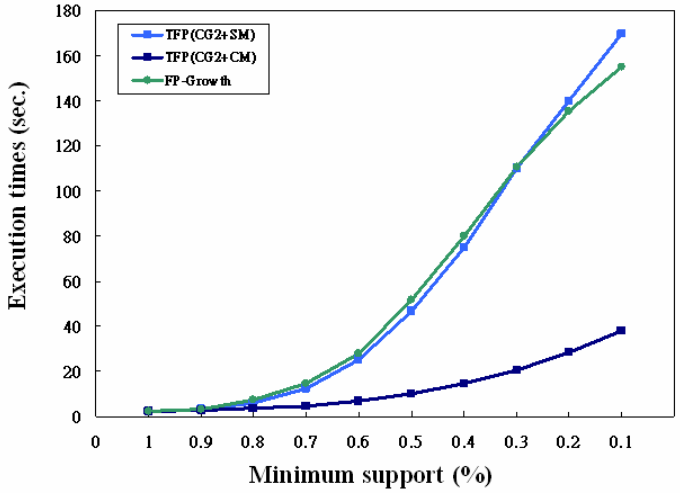


Fig. 4. Execution times on dataset T5I2D500K

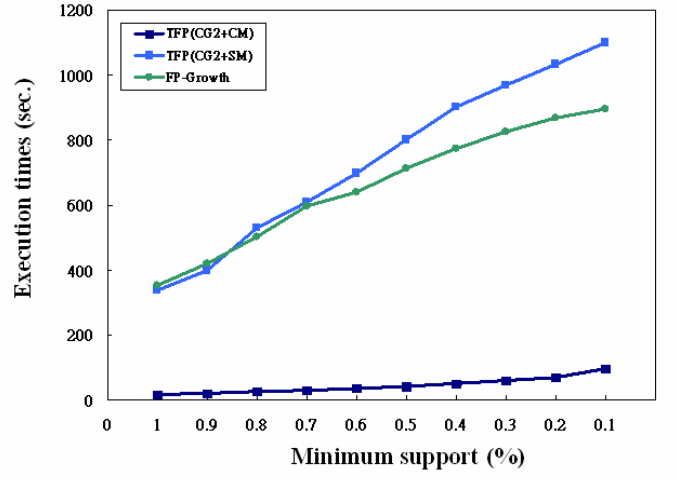


Fig. 5. Execution times on dataset T10I4D500K

FP-tree and conditional FP-trees. Because CG2 can generate fewer combinations to be computed for each traversal than CG1 and CM merges the subtrees much more compact than SM, TFP(CG2+CM) is much more efficient than the other algorithms. Therefore, the following experiments only compare the three algorithms TFP(CG2+SM), TFP(CG2+CM) and *FP-Growth*.

Figures 4 and 5 show the execution times for the three algorithms on dataset T5I2D100K and T10I4D100K, respectively, with minimum support from 0.1% to 1%. In Figure 6, TFP(CG2+SM) slightly outperforms *FP-Growth* when minimum support is higher. However, *FP-Growth* slightly outperforms TFP(CG2+SM) as the minimum support decreases. In general, the performances of the two algorithms are similar. TFP(CG2+CM) also significantly outperforms the other two algorithms.

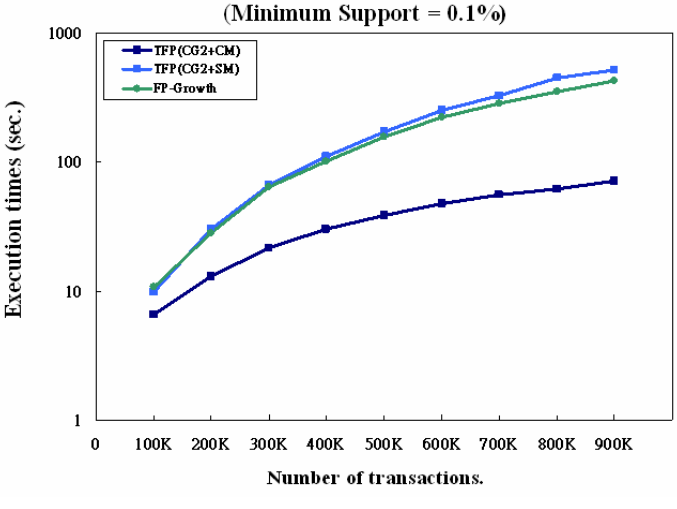


Fig. 6. Execution times on dataset T5I2

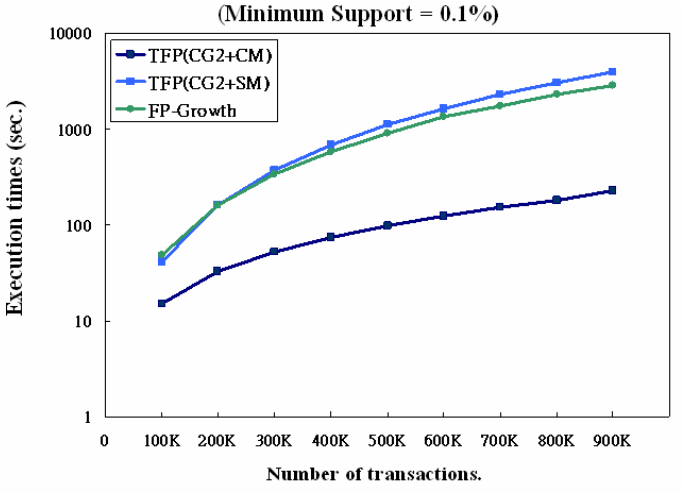


Fig. 7. Execution times on dataset T10I4

The scale up analysis is shown in Figures 6 and 7. The execution times slightly increases as the number of transactions increases for the three algorithms. However, the shape of TFP(CG2+CM) in Figure 6 is more smooth and stable than the other two algorithms, when the number of transactions increases by 100K. From our experiments, we can conclude that TFP(CG2+CM) is the most efficient and stable among all the algorithms based on *FP-tree* structure.

4 Conclusions

This paper studies frequent pattern mining algorithm based a frequent pattern tree and proposes four algorithms. Our approach only needs to construct a FP-tree and traverse each subtree of the FP-tree to generate all the frequent patterns for an item without constructing any other subtrees. Although the candidate itemsets need to be generated, there are only few candidates generated in each traversal for TFP(CG2+CM) algorithm. Through merging and removing subtrees and choosing the item with the highest support, our approach can generate frequent itemsets for an item by only traversing a subtree. Since there is no extra trees constructed and the frequent patterns generated for an item only need to traverse a subtree, the experimental results show that TFP(CG2+CM) outperforms FP-Growth significantly.

References

1. Agarwal, R., Aggarwal, C., Prasad, V.: A tree projection algorithm for generation of frequent itemsets. Parallel and distributed Computing 61, 350–371 (2000)
2. Agarwal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc. 1994 Int. Conf. Very Large Data Bases, Santiago, Chile, pp. 487–499 (September 1994)

3. Chen, M., Park, J., Yu, P.: An Effective Hash-Based Algorithm for Mining Association Rules. Proceedings of ACM SIGMOD 24(2), 175–186 (1995)
4. Han, J., Mao, R., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data mining and knowledge discovery 8, 53–87 (2004)
5. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: ACM-SIGMOD, Dallas (2000)
6. Yen, S.J., Chen, A.L.P.: A Graph-Based Approach for Discovering Various Types of Association Rules. IEEE Transactions on Knowledge and Data Engineering (TKDE) 13(5), 839–845 (2001)
7. IBM Almaden. Quest synthetic data generation code, `http://www.almaden.ibm.com/cs/quest/syndata.html`
8. `http://appsrv.cse.cuhk.hk/~kdd/program.htm`

Discovering Periodic-Frequent Patterns in Transactional Databases

Syed Khairuzzaman Tanbeer, Chowdhury Farhan Ahmed, Byeong-Soo Jeong, and Young-Koo Lee

Department of Computer Engineering, Kyung Hee University
1 Seochun-dong, Kihung-gu, Youngin-si, Kyunggi-do, 446-701, Republic of Korea
{tanbeer,farhan,jeong,yklee}@khu.ac.kr

Abstract. Since mining frequent patterns from transactional databases involves an exponential mining space and generates a huge number of patterns, efficient discovery of user-interest-based frequent pattern set becomes the first priority for a mining algorithm. In many real-world scenarios it is often sufficient to mine a small interesting representative subset of frequent patterns. Temporal periodicity of pattern appearance can be regarded as an important criterion for measuring the interestingness of frequent patterns in several applications. A frequent pattern can be said periodic-frequent if it appears at a regular interval given by the user in the database. In this paper, we introduce a novel concept of mining periodic-frequent patterns from transactional databases. We use an efficient tree-based data structure, called Periodic-frequent pattern tree (PF-tree in short), that captures the database contents in a highly compact manner and enables a pattern growth mining technique to generate the complete set of periodic-frequent patterns in a database for user-given periodicity and support thresholds. The performance study shows that mining periodic-frequent patterns with PF-tree is time and memory efficient and highly scalable as well.

Keywords: Data mining, knowledge discovery, frequent pattern, interesting pattern, periodic-frequent pattern.

1 Introduction

Mining frequent patterns [1], [2], [4], [6] from transactional databases has been actively and widely studied in data mining and knowledge discovery techniques such as association rule, sequential pattern, classification, and clustering. Since the rationale behind mining the support metric-based frequent patterns is to find the set of patterns that appear frequently in a database, a huge number of patterns are normally generated and most of which might be found insignificant depending on application or user requirement. Moreover, the computation cost in finding such number of patterns may not be trivial. As a result, several techniques to mine constraint-based and/or user interest-based frequent patterns [9], [10], [5] have been proposed recently to reduce the desired result set by effectively

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 242–253, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Table 1. A transactional database

Id	Transaction	Id	Transaction	Id	Transaction	Id	Transaction	Id	Transaction
1	*a c d e*	3	*a c e*	5	*a c e f*	7	*b c d e*	9	*a b c d*
2	*a d e f*	4	*c d e*	6	*b f*	8	*b c d e*	10	*a b e f*

and efficiently applying early pruning techniques. Uses of several interesting parameters such as closed [3], K-most [5], demand-driven [10], maximum length [9] are found useful in literature in discovering frequent patterns of special interest. The other important criterion for identifying the interestingness of frequent patterns might be the shape of occurrence, i.e., whether they occur periodically, irregularly, or mostly in specific time interval in the database.

In a retail market, among all frequently sold products, the user may be interested only on the regularly sold products compared to the rest. Besides, for improved web site design or web administration an administrator may be interested on the click sequences of heavily hit web pages. Also, in genetic data analysis the set of all genes that not only appear frequently but also co-occur at regular interval in DNA sequence may carry more significant information to scientists. As for stock market, the set of high stocks indices that rise periodically may be of special interest to companies and individuals. In the above examples, we observe that the occurrence periodicity plays an important role in discovering some interesting frequent patterns in a wide variety of application areas. We define such a frequent pattern that appears maintaining a similar period/interval in a database as a *periodic-frequent pattern*.

Let us consider the transactional database of Table 1 with ten transactions. The support of the patterns "*e*", "*ae*", "*cd*", "*ce*", "*b*", and "*de*" in the database are respectively 8, 5, 5, 6, 5, and 5. Even though these patterns may be frequent in the database, some of them may not be periodic-frequent because of non-similar occurrence periods. For example, "*b*" and "*ae*" appear more frequently at a certain part of the database (i.e., "*b*" at the end and "*ae*" at the beginning of database) than the rest part. In contrast, patterns "*e*", "*cd*", "*ce*", "*de*" appear at almost regular intervals. Therefore, the latter patterns can be more important frequent patterns in terms of the appearance intervals. On the other hand, although the respective appearance intervals of patterns "*ac*", "*cde*", "*f*" etc. are almost similar, they may not be the patterns of interest due to their relatively low frequency. The traditional frequent pattern mining techniques fail to discover such periodic-frequent patterns because they are only concerned about the occurrence frequency and disregard the pattern occurrence behavior.

Motivated by the above discussion and examples, in this paper, we address a new problem of discovering *periodic-frequent patterns* in a transactional database. We define a new *periodicity* measure for a pattern by the maximum interval at which the same pattern occurs in a database. Therefore, periodic-frequent patterns, defined such way, satisfy the downward closure property [1], i.e., if a frequent pattern is found periodic then all of its non-empty subsets will be periodic. In other words, if a frequent pattern is not periodic then none of its supersets can be periodic. In order to mine periodic-frequent patterns, we

capture the database contents in a highly compact tree structure, called a PF-tree (Periodic-frequent Pattern tree). To ensure that the tree structure is compact and informative, only periodic-frequent length-1 items will have nodes in the tree and to obtain higher prefix sharing, more frequently occurring items are located at the upper part of the tree. We also propose an efficient pattern growth-based mining approach to mine the complete set of periodic-frequent patterns from our PF-tree. The comprehensive performance study on both synthetic and real datasets demonstrates that discovering periodic-frequent patterns from the PF-tree is highly memory and time efficient.

The rest of the paper is organized as follows. In Section 2, we summarize the existing algorithms to mine interesting frequent patterns. Section 3 formally introduces the problem of periodic-frequent pattern mining. The structure and mining of PF-tree are given in Section 4. We report our experimental results in Section 5. Finally, Section 6 concludes the paper.

2 Related Work

Since its introduction by Agrawal et al. in 1993 [1], a large number of techniques [2], [4], [6] have been proposed in mining support constraint-based frequent patterns. Han et al. [2] proposed the frequent pattern tree (FP-tree) and the FP-growth algorithm to mine frequent patterns with a memory and time efficient manner. The performance gain achieved by the FP-growth is mainly based on the highly compact nature of the FP-tree, where it stores only the frequent items in a support-descending order. To reduce the size of resultant pattern set and to improve the mining efficiency closed [3] frequent pattern mining has been focused. However, none of the above frequent pattern mining techniques can successfully provide interesting frequent patterns, since their outputs are only based on the support threshold.

Mining interesting frequent patterns of different forms [9], [10], [5], [7] in transactional databases and time-series data has been well-addressed over the last decade. Minh et al. [5] proposed a top-K frequent pattern mining technique that allows the user to control the number of patterns to be discovered without any support threshold. In [9] the authors put efforts to discover the maximum length frequent patterns, rather than finding the complete set of frequent patterns. They have shown the suitability of their method in several real world scenario where long patterns play significant role. Wang et al. [10] mined frequent patterns from relational database. Using the user's query, they find the frequently occurring pattern structures defined by attributes values of items. However, the above models still fail to discover the interesting periodic occurrence characteristics of frequent patterns.

Temporal relationships among pattern occurrences were studied in [7] which focused on discovering the frequently occurring substring patterns in a dimension of multivariate time-series data. Periodic pattern mining has also been studied as a wing of sequential pattern mining [8] in recent years. Although periodic pattern mining is closely related to our work, it cannot be directly applied for

finding the periodic-frequent patterns from a transactional database because of two primary reasons. First, it considers either time-series or sequential data; second, it does not consider the support threshold which is the only constraint to be satisfied by all frequent patterns. Our proposed periodic-frequent pattern mining technique, on the other hand, introduces a new interesting measure of periodicity and provides the set of patterns that satisfy both of the periodicity and support thresholds in a transactional database.

3 Problem Definition

In this section, we describe the conceptual framework of the periodic-frequent pattern mining and introduce the basic notations and definitions in this regard.

Let $I = \{i_1, i_2, ..., i_n\}$ be a set of literals, called items that have ever been used as a unit information of an application domain. A set $X = \{i_j, ..., i_k\} \subseteq I$, where $j \leq k$ and $j, k \in [1, n]$, is called a pattern (or an itemset). A transaction $t = (tid, Y)$ is a tuple where *tid* represents a transaction-id (or timestamp) and Y is a pattern. A transactional database TDB over I is a set of transactions $T = \{t_1, ..., t_m\}$, $m = |TDB|$, where $|TDB|$ is the size of TDB in total number of transactions. If $X \subseteq Y$, it is said that t contains X or X occurs in t and such transaction-id is denoted as t_j^X, $j \in [1, m]$. Therefore, $T^X = \{t_j^X, ..., t_k^X\}$, $j, k \in [1, m]$ and $j \leq k$ is the set of all transaction-ids where X occurs in TDB.

Definition 1. (***a period of pattern*** X) *Let t_{j+1}^X and t_j^X, $j \in [1, (m-1)]$ be two consecutive transaction-ids where X appears. The number of transactions or the time difference between t_{j+1}^X and t_j^X can be defined as a period of X, say p^X (i.e., $p^X = t_{j+1}^X - t_j^X$, $j \in [1, (m-1)]$). For the simplicity of period computation, the first and the last transactions (say, t_f and t_l) in TDB are respectively identified as "null" (i.e., $t_f = 0$) and t_m (i.e., $t_l = t_m$). For instant, in the TDB of Table 1 the set of transactions where pattern "de" appears is $T^{de} = \{1, 2, 4, 7, 8\}$. Therefore, the periods for this pattern are $1(= 1 - t_f)$, $1(= 2 - 1)$, $2(= 4 - 2)$, $3(= 7 - 4)$, $1(= 8 - 7)$, and $2(= t_l - 8)$, where $t_f = 0$ and $t_l = 10$.*

The occurrence intervals, defined as above, can give the exact information of appearance behavior of a pattern. The largest occurrence period of a pattern, therefore, can provide the upper limit of its periodic occurrence characteristic. Hence, the measure of the characteristic of a pattern of being *periodic* in a TDB (we call it as the periodicity of that pattern) can be defined as follows.

Definition 2. (***periodicity of pattern*** X) *Let for a T^X, P^X be the set of all periods of X i.e., $P^X = \{p_1^X, ..., p_r^X\}$, where r is the total number of periods in P^X. Then, the periodicity of X can be denoted as $Per(X) = Max(p_1^X, ..., p_r^X)$. For example, in the TDB of Table 1, $Per(de) = 3$ i.e., $Max(1, 1, 2, 3, 1, 2)$.*

Definition 3. (***support of pattern*** X) *The number of transactions in a TDB that contain X is called the support of X in TDB and denoted as $Sup(X)$. Therefore, $Sup(X) = |T^X|$, where $|T^X|$ is the size of T^X. For example, the support of pattern "de" in the TDB of Table 1 is $Sup(de) = 5$, since $|T^{de}| = 5$.*

A pattern is called a *periodic-frequent pattern* if it satisfies both of the following two criteria: (i) its periodicity is no greater than a user-given maximum periodicity threshold say max_per, λ and (ii) its support is no less than a user-given minimum support threshold, say min_sup, α, with λ, α in percentage of $|TDB|$. Therefore, the *Periodic-frequent pattern mining problem*, given λ, α, and a TDB, is to discover the complete set of periodic-frequent patterns in TDB having periodicity no more than λ and support no less than α. Let PF_{TDB} refer to the set of all periodic-frequent patterns in a TDB for given λ and α.

4 PF-Tree: Design, Construction and Mining

In this section, we describe the construction and mining of Periodic-Frequent Pattern tree (PF-tree). Since periodic-frequent patterns follow the downward closure property, periodic-frequent length-1 items will play an important role in mining periodic-frequent patterns. Therefore, it is necessary to perform one database scan to identify the set of length-1 periodic-frequent items. The objective of this scan is to collect the support count (i.e., frequency) and the periodicity of each item in the database. Consequently, for further processing we can ignore all items that do not satisfy the periodicity and support thresholds. Let PF be the set of all items that are found periodic-frequent at this stage.

4.1 Structure of the PF-Tree

The structure of the PF-tree includes a prefix-tree and a periodic-frequent item list, called the PF-list, consisting of each distinct item with relative support, periodicity and a pointer pointing to the first node in the PF-tree carrying the item. To facilitate high degree of compactness, items in a PF-tree are arranged in support-descending item order. It has been proved in [6] that such tree can provide a highly compact tree structure (as FP-tree in [2] and CP-tree in [6]) and an efficient mining phase using FP-growth mining technique. Before discussing the tree construction process, we provide the PF-list construction technique and the node structures of a PF-tree.

Construction of the PF-list. Each entry in a PF-list consists of three fields - item name (i), total support (f), and the periodicity of i (p). The tids of the last occurring transactions of all items in the PF-list are explicitly recorded for each item in a temporary array, called id_l. Let t_{cur} and p_{cur} respectively denote the tid of current transaction and the most recent period for an item i. The PF-list is, therefore, maintained according to the process given in Fig. 1.

In Fig. 2 we show how the PF-list is populated for the TDB of Table 1. With the scan of the first transaction $\{a\ c\ d\ e\}$ (i.e., $t_{cur} = 1$), the items 'a', 'c', 'd', and 'e' in the list are initialized as shown in Fig. 2(a) (lines 1 and 2 in Fig. 1). The next transaction $\{a\ d\ e\ f\}$ with $t_{cur} = 2$ initializes PF-list entries for item 'f' and updates values $\{f; p\}$ and id_l (lines 3 - 6 in Fig. 1) respectively to $\{2; 1\}$ and $\{2\}$ for items 'a', 'd', and 'e' (Fig. 2(b)). As shown in Fig. 2(c), the periodicity (p)

1. **If** t_{cur} is i's first occurrence
2. $f = 1$, $id_l = t_{cur}$, $p = t_{cur}$;
3. **Else** $f = f + 1$;
4. $p_{cur} = t_{cur} - id_l$, $id_l = t_{cur}$;
5. **If** ($p_{cur} > p$)
6. $p = p_{cur}$;
7. At the end of *TDB*, calculate p_{cur} for each item by considering t_{cur} = the *tid* of the last transaction in *TDB*, and update the respective p value according to step 5 and 6;

Fig. 1. PF-list maintenance algorithm

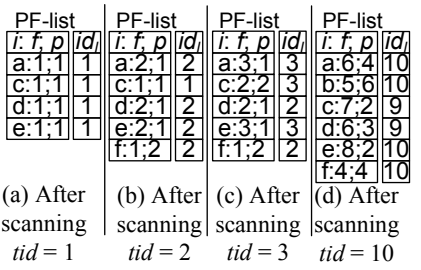


Fig. 2. PF-list population after the first scan of the *TDB* in Table 1

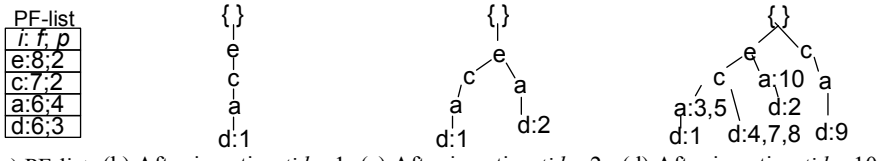
of 'c' changes from 1 to 2, since after scanning $tid = 3$ the value of p_{cur} for it is found greater than its previous periodicity (lines 5 and 6 in Fig. 1). The PF-list after scanning all ten transactions is given in Fig. 2(d). To reflect the correct periodicity for each item in the list, the whole PF-list is refreshed as mentioned in line 7 of Fig. 1 which results the final PF-list of Fig. 2(d). Therefore, once the PF-list is built, we generate the PF by removing items that do not satisfy the user-given periodicity and support thresholds from it.

PF-tree Node Structures. An important feature of a PF-tree is that, it explicitly maintains the occurrence information for each transaction in the tree structure by keeping an occurrence transaction-id list, called *tid*-list, only at the last node of every transaction. Hence, there are two types of nodes maintained in a PF-tree; ordinary node and *tail*-node. The former is the type of nodes similar to that used in FP-tree, whereas the latter is the node that represents the last item of any sorted transaction. Therefore, the structure of a *tail*-node is $N[t_1, t_2, ..., t_n]$, where N is the node's item name and t_i, $i \in [1, n]$, (n be the total number of transactions from the *root* up to the node) is a transaction-id where item N is the last item. Like the FP-tree [2], each node in a PF-tree maintains parent, children, and node traversal pointers. However, irrespective of the node type, no node in a PF-tree maintains support count value in it.

4.2 Construction of the PF-Tree

With the second database scan, the PF-tree is constructed in such a way that, it only contains nodes for items in PF. We use an example to illustrate the construction of a PF-tree.

Consider the transactional database of Table 1. In Fig. 3, we show the PF-tree construction steps for $\lambda = 4$ and $\alpha = 5$. At first, the support-descending PF-list (Fig. 3(a)) for all periodic-frequent items is constructed from the PF-list of Fig. 2(d). Next, using the FP-tree [2] construction technique, only the items in this list take part in PF-tree construction. For the simplicity of figures, we do not show the node traversal pointers in trees, however, they are maintained in a fashion like FP-tree does. The tree construction starts with inserting the



(a) PF-list (b) After inserting *tid* = 1 (c) After inserting *tid* = 2 (d) After inserting *tid* = 10

Fig. 3. Construction of a PF-tree for the *TDB* in Table 1 with $\alpha = 5$ and $\lambda = 4$

first transaction $\{a\ c\ d\ e\}$ (i.e., $tid = 1$) according to PF-list order, as shown in Fig. 3(b), since all the items in the transactions are periodic-frequent. The *tail*-node "$d : 1$" carries the *tid* of the transaction. After removing the non-periodic-frequent item 'f', the second transaction is inserted into the tree in the form and an order of $\{e\ a\ d\}$ with node "$d : 2$" as the *tail*-node for it (Fig. 3(c)). After inserting all the transactions in similar fashion we get the final PF-tree for the database as shown in Fig. 3(d).

Based on the PF-list population technique discussed in Section 4.1 and the above example, we have the following property and lemmas of a PF-tree. For each transaction t in a TDB, $PF(t)$ is the set of all periodic-frequent items in t, i.e., $PF(t) = item(t) \bigcap PF$, and is called the periodic-frequent item projection of t.

Property 1. *A PF-tree maintains a complete set of periodic-frequent item projection for each transaction in a TDB only once.*

Lemma 1. *Given a transactional database TDB, a max_per, and a min_sup, the complete set of all periodic-frequent item projections of all transactions in a TDB can be derived from the PF-tree for both of the max_per and min_sup.*

Proof: Based on Property 1, $PF(t)$ of each transaction t is mapped to only one path in the tree and any path from the *root* up to a *tail*-node maintains the complete projection for exactly n transactions (where n is the total number of entries in the *tid*-list of the *tail*-node).

Lemma 2. *The size of a PF-tree (without the root node) on a transactional database TDB for a max_per, and a min_sup is bounded by* $\sum_{t \in TDB} |PF(t)|$.

Proof: According to the PF-tree construction process and Lemma 1, each transaction t contributes at best one path of the size $|PF(t)|$ to a PF-tree. Therefore, the total size contribution of all transactions can be $\sum_{t \in TDB} |PF(t)|$ at best. However, since there are usually a lot of common prefix patterns among the transactions, the size of a PF-tree is normally much smaller than $\sum_{t \in TDB} |PF(t)|$.

One can assume that the structure of a PF-tree may not be memory efficient, since it explicitly maintains *tid*s of each transaction. But, we argue that the PF-tree achieves the memory efficiency by keeping such transaction information only at the *tail*-nodes and avoiding the support count field at each node. Moreover, keeping the *tid* information in tree can also been found in literature for efficient frequent pattern mining [3], [4].

Therefore, the highly compact PF-tree structure maintains the complete information for all periodic-frequent patterns. Once the PF-tree is constructed, we use an FP-growth-based pattern growth mining technique to discover the complete set of periodic-frequent patterns from it.

4.3 Mining Periodic-Frequent Pattern

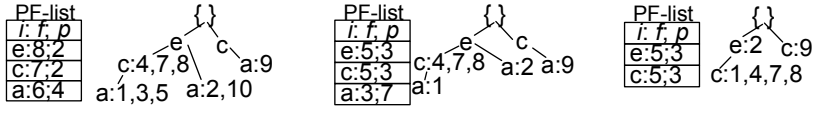
Even though both of the PF-tree and FP-tree arrange items in support-descending order, we can not directly apply the FP-growth mining on a PF-tree. The reason is that, PF-tree does not maintain the support count at each node, and it handles the *tid*-lists at *tail*-nodes. Therefore, we devise a pattern growth-based bottom-up mining technique that can handle the additional features of the PF-tree. The basic operations in mining a PF-tree for periodic-frequent patterns are (i) counting length-1 periodic-frequent items, (ii) constructing the prefix-tree for each periodic-frequent itemset, and (iii) constructing the conditional tree from each prefix-tree. The PF-list provides the length-1 periodic-frequent items. Before discussing the prefix-tree construction process we explore the following important property and lemma of a PF-tree.

Property 2. *A tail-node in a PF-tree maintains the occurrence information for all the nodes in the path (from that tail-node to the root) at least in the transactions in its tid-list.*

Lemma 3. *Let $Z = \{a_1, a_2, ..., a_n\}$ be a path in a PF-tree where node a_n is the tail-node carring the tid-list of the path. If the tid-list is pushed-up to node a_{n-1}, then a_{n-1} maintains the occurrence information of the path $Z' = \{a_1, a_2, ..., a_{n-1}\}$ for the same set of transactions in the tid-list without any loss.*

Proof: Based on Property 2, a_n maintains the occurrence information of the path Z' at least in the transactions in its *tid*-list. Therefore, the same *tid*-list at node a_{n-1} exactly maintains the same transaction information for Z' without any lose.

Using the features revealed by the above property and lemma, we proceed to construct each prefix-tree starting from the bottom-most item, say i, of the PF-list. Only the prefix sub-paths of nodes labeled i in the PF-tree are accumulated as the prefix-tree for i, say PT_i. Since i is the bottom-most item in the PF-list, each node labeled i in the PF-tree must be a *tail*-node. While constructing the PT_i, based on Property 2 we map the *tid*-list of every node of i to all items



(a) PF-tree after removing item '*d*' (b) Prefix-tree for '*d*' (c) Conditional tree for '*d*'

Fig. 4. Prefix-tree and conditional tree construction with PF-tree

in the respective path explicitly in a temporary array (one for each item). It facilitates the periodicity and support calculation for each item in the PF-list of PT_i. Moreover, to enable the construction of the prefix-tree for the next item in the PF-list, based on Lemma 3 the *tid*-lists are pushed-up to respective parent nodes in the original PF-tree and in PT_i as well. All nodes of i in the PF-tree and i's entry in the PF-list are deleted thereafter. Figure 4(a) shows the status of the PF-tree of Fig. 3(d) after removing the bottom-most items 'd'. Besides, the prefix-tree for 'd', PT_d is shown in Fig. 4(b).

The conditional tree CT_i for PT_i is constructed by removing all non-periodic-frequent nodes from the PT_i. If the deleted node is a *tail*-node, its *tid*-list is pushed-up to its parent node. Figure 4(c), for instance, shows the conditional tree for 'd', CT_d constructed from the PT_d of Fig. 4(b). The contents of the temporary array for the bottom item j in the PF-list of CT_i represent the T^{ij} (i.e., the set of all *tid*s where items i and j occur together). Therefore, it is rather simple calculation to compute $Per(ij)$ and $Sup(ij)$ from T^{ij} by generating P^{ij}. Then the pattern "ij" is generated as a periodic-frequent pattern with the periodicity and support values of $Per(ij)$ and $Sup(ij)$, respectively. The same process of creating prefix-tree and its corresponding conditional tree is repeated for further extensions of "ij". The whole process of mining for each item is repeated if PF-list $\neq \emptyset$.

The above bottom-up mining technique on support-descending PF-tree is efficient, because it shrinks the search space dramatically with the progress of mining process. In the next section, we present the experimental results of finding periodic-frequent patterns from the PF-tree.

5 Experimental Results

Since there is no existing approach to discover periodic-frequent patterns, we only investigate PF-tree's performance. All programs are written in Microsoft Visual C++ 6.0 and run with Windows XP on a 2.66 GHz machine with 1GB memory. The runtime specifies the total execution time, i.e., CPU and I/Os. The experiments are pursued on several synthetic (*T10I4D100K*) and real datasets (*chess*, *mushroom*, and *kosarak*) respectively developed at IBM Almaden Quest research group (`http://www.almaden.ibm.com/cs/quest`) and obtained from UCI Machine Learning Repository (University of California - Irvine, CA). *T10I4D100K* is a large sparse dataset with 100,000 transactions and 870 distinct items. The dense datasets *chess* and *mushroom* contain 3,196 and 8,124 transactions, and 75 and 119 distinct items respectively. In the first experiment, we study the compactness of the PF-tree on different datasets.

5.1 Compactness of the PF-Tree

The memory consumptions of PF-tree on the variations of *max_per* and *min_sup* values over several datasets are reported in Table 2. The first and second columns of the table respectively show the dataset-dependent different *max_per* and

Table 2. Memory requirements for the PF-tree

Dataset (*max_per* **values**)	α(%)	**Memory(MB)**		
		λ_1	λ_2	λ_3
mushroom	15	0.068	0.088	0.107
$\lambda_1 = 2.0\%$, $\lambda_2 = 4.0\%$, $\lambda_3 = 6.0\%$	35	0.049	0.050	0.052
chess	55	0.015	0.017	0.019
$\lambda_1 = 0.5\%$, $\lambda_2 = 0.6\%$, $\lambda_3 = 0.7\%$	85	0.014	0.016	0.016
T10I4D100K	1.5	0.288	5.090	7.349
$\lambda_1 = 0.2\%$, $\lambda_2 = 0.4\%$, $\lambda_3 = 0.6\%$	4.5	0.241	0.281	0.281

min_sup values we used in the experiment. The size of the PF-tree is, therefore, shown in the last three columns for the respective thresholds.

The data in the table demonstrate that, keeping the *min_sup* fixed the memory consumption of PF-tree increases with the increase of *max_per* for almost all of the datasets. In contrast, for fixed *max_per* the tree size becomes smaller with increasing values of *min_sup*. The reason of such threshold-dependent tree size variation is that, more and more patterns become periodic-frequent with the increase of *max_per* and the decrease of *min_sup* values. Therefore, the PF-tree size increases to represent the increasing pattern set. However, it is clear from the Table 2 that, the structure of the PF-tree can easily be handled in a memory efficient manner irrespective of the dataset type (dense or sparse) or size (large or small) and threshold values. In the next experiment, we show the execution time performance of PF-tree in mining periodic-frequent patterns.

5.2 Execution Time of the PF-Tree

The changes on the periodicity and the support thresholds show the similar effect on execution time as of the size of PF-tree structure. Because of the space limitations, we report the results, in Fig. 5, only on *T10I4D100K* and *mushroom* datasets. The execution time shown in the graphs encompasses all phases of PF-list and PF-tree constructions, and the corresponding mining operation.

We varied the values of both thresholds as we demonstrated in the previous experiment. It can be noticed from the graphs in Fig. 5 that, for both sparse

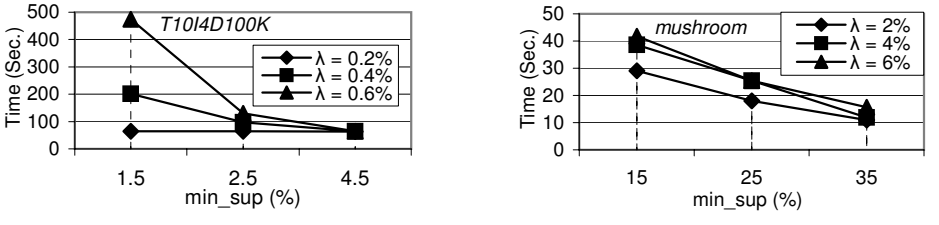


Fig. 5. Execution time on the PF-tree

and dense datasets PF-tree takes similar amount of time for relatively higher support threshold values for the variation of the periodicity thresholds. However, as the support thresholds go down, the gaps become wider. From another point of view, keeping the *max_per* fixed, the execution time increases (mainly for higher *max_per*s) with lowering the *min_sup*. The reason of such performance variation is that, for a fixed *min_sup* value the number and the lengths of periodic-frequent patterns increase for higher values of *max_per*. For a fixed *max_per* value, on the other hand, the same effect we get for lower *min_sup* values. In general, when mining for lower *min_sup* and higher *max_per* values, the PF-tree requires more execution time. However, as per as the database size and reasonably high *max_per* and low *min_sup* values are concerned, we see that mining periodic-frequent patterns from the corresponding PF-tree is rather time efficient for both sparse and dense datasets. The scalability study on PF-tree, discussed in the next subsection, also reflects this scenario.

5.3 Scalability of the PF-Tree

We study the scalability of our PF-tree on execution time and required memory by varying the number of transactions in database. We use real *kosarak* dataset for the scalability experiment, since it is a huge sparse dataset with a large number of distinct items (41,270) and transactions (990,002). We divided the dataset into five portions of 0.2 million transactions in each part. Then we investigated the performance of PF-tree after accumulating each portion with previous parts with performing periodic-frequent pattern mining each time. We fix the *max_per* to 50% and the *min_sup* to 2% of |*kosarak*| for each experiment. The experimental results are shown in Fig. 6. The time and memory in y-axes of the left and right graphs in Fig. 6 respectively specify the total execution time and required memory with the increase of database size. It is clear from the graphs that as the database size increases, overall tree construction and mining time, and memory requirement increase. However, PF-tree shows stable performance of about linear increase of runtime and memory consumption with respect to the database size. Therefore, it can be observed from the scalability test that PF-tree can mine the PF_{TDB} over large datasets and distinct items with considerable amount of runtime and memory.

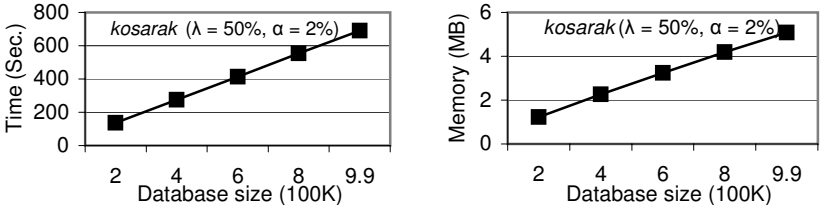


Fig. 6. Scalability of the PF-tree

6 Conclusions

In this paper, we have introduced a new interesting measure, called temporal periodicity of occurrence behavior, for frequently recurring patterns in transactional databases. We have defined such patterns as the periodic-frequent patterns under the user-given periodicity and support thresholds. This paper also shows the significance of discovering such patterns in a wide range of real-world application areas. We have provided the PF-tree, a highly compact tree structure to capture the database content, and a pattern growth-based mining technique to discover the complete set of periodic-frequent patterns on the user-given maximum periodicity and minimum support thresholds over a transactional database. The experimental results demonstrate that our PF-tree can provide the time and memory efficiency during mining the periodic-frequent pattern set. Moreover, it is highly scalable in terms of runtime and memory consumption.

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining Association Rules Between Sets of Items in Large Databases. In: ACM SIGMOD Int. Conf. on Management of Data, pp. 207–216 (1993)
2. Han, J., Pei, J., Yin, Y.: Mining Frequent Patterns without Candidate Generation. In: ACM SIGMOD Int. Conf. on Management of Data, pp. 1–12 (2000)
3. Zaki, M.J., Hsiao, C.-J.: Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure. IEEE Trans. on Knowl. and Data Eng. 17(4), 462–478 (2005)
4. Zhi-Jun, X., Hong, C., Li, C.: An Efficient Algorithm for Frequent Itemset Mining on Data Streams. In: Int. Conf. on Management of Data, pp. 474–491 (2006)
5. Minh, Q.T., Oyanagi, S., Yamazaki, K.: Mining the K-Most Interesting Frequent Patterns Sequentially. In: Corchado, E., Yin, H., Botti, V., Fyfe, C. (eds.) IDEAL 2006. LNCS, vol. 4224, pp. 620–628. Springer, Heidelberg (2006)
6. Tanbeer, S.K., Ahmed, C.F., Jeong, B.-S., Lee, Y.-K.: CP-tree: A Tree Structure for Single-Pass Frequent Pattern Mining. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS(LNAI), vol. 5012, pp. 1022–1027. Springer, Heidelberg (2008)
7. Tatavarty, G., Bhatnagar, R., Young, B.: Discovery of Temporal Dependencies between Frequent Patterns in Multivariate Time Series. In: The 2007 IEEE Symposium on Computational Intelligence and Data Mining, pp. 688–696 (2007)
8. Maqbool, F., Bashir, S., Baig, A.R.: E-MAP: Efficiently Mining Asynchronous Periodic Patterns. Int. J. of Comp. Sc. and Net. Security 6(8A), 174–179 (2006)
9. Hu, T., Sung, S.Y., Xiong, H., Fu, Q.: Discovery of Maximum Length Frequent Itemsets. Information Sciences 178, 69–87 (2008)
10. Wang, H., Perng, C.-S., Ma, S., Yu, P.S.: Demand-driven Frequent Itemset Mining Using Pattern Structures. Knowledge and Information Systems 8, 82–102 (2005)

Quantifying Asymmetric Semantic Relations from Query Logs by Resource Allocation

Zhiyuan Liu, Yabin Zheng, and Maosong Sun

Department of Computer Science and Technology,
State Key Lab on Intelligent Technology and Systems,
National Lab for Information Science and Technology,
Tsinghua University, Beijing, China, 100084
{liuliudong,yabin.zheng}@gmail.com, sms@tsinghua.edu.cn
http://nlp.csai.tsinghua.edu.cn

Abstract. In this paper we present a bipartite-network-based resource allocation(BNRA) method to extract and quantify semantic relations from large scale query logs of search engine. Firstly, we construct a query-URL bipartite network from query logs of search engine. By BNRA, we extract asymmetric semantic relations between queries from the bipartite network. Asymmetric relation indicates that two related queries could be assigned different semantic relevance strength against each other, which is more conforming to reality. We verify the validity of the method with query logs from Chinese search engine Sogou. It demonstrates BNRA could effectively quantify semantic relations from We further construct query semantic networks, and introduce several measures to analyze the networks. BNRA is not only 'language oblivious' and 'content oblivious', but could also be easily implemented in a paralleled manner, which provides commercial search engines a feasible solution to handle large scale query logs.

Keywords: Semantic relations, query log, resource allocation, asymmetric.

1 Introduction

With the development of Internet, search engine, such as Google, has become the most important tool to get information on the World Wide Web. Although it is not a perfect method to find what users want, most search engines calculate the relevance using keywords in documents and queries. As the only interface for users to access tremendous web pages, queries are one of the most important factors that affect the performance of search engines.

However, web pages returned from search engines are not always relevant to search intentions of users. An independent survey of 40,000 web users found that after a failed search, 76% of them will try to rephrase their queries on the same search engine instead of resorting to a different one [1]. Therefore, it is a non-trivial task for search engines to find better query representation of user search intentions in order to enhance search performance.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 254–265, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Search behavior of most users, including query submissions and URL selections, are meaningful. Therefore, queries convey implicit knowledge, concepts or meanings, which could be regarded as tags assigned to selected URLs by users [2]. Query logs, recording the history of click through behaviors by users from queries to selected URLs, may thereby contain tremendous user collaborative tagging information as a result of the wisdom of all users, and have attracted much work trying to extract useful information so as to improve search engine performance. Various tasks, such as query clustering [1,3], classification [4,5], recommendation [6], expansion [7] and reformulation [8], have been proposed to address the challenges from different perspectives. The common basis of these tasks is to quantify semantic relations of queries.

In a narrow sense, **semantic relations** are relations between concepts or meanings. Queries, regarded as the tags assigned by users to selected URLs, contain rich semantic relations, which imply a taxonomy of the language that people use to search for information [2]. Hence, it is essential to extract useful relations from query logs in order to improve search engine performance in various tasks mentioned above.

Most previous researches extracted semantic relations by defining a similarity function between queries based on substring matching of queries or intersection of selected URL sets. The main drawback of these methods is that the extracted relations are symmetric, which indicates the similarity function gives two queries the same relevance strength against each other. However in most instances two related queries should be assigned different relevance strength. For example, the relevance strength for query 'ipod'(a product of Apple Inc.) with 'apple' may be stronger than that for 'apple' with 'ipod': with query 'ipod', users likely want to get the information of the websites on the mp3 product ipod, therefore it is related with it manufacturer 'apple' strongly. While with query 'apple', users may have more complicated and extensive intentions, could be a fruit or an IT company, and thus not have equal strong relevance with 'ipod'. Hence, it is crucial to extract and quantify asymmetric semantic relations of queries.

In this paper we propose to apply a bipartite-network-based resource allocation(BNRA) method [9] to flexibly extract and quantify asymmetric semantic relations of queries for the first time. The method is originally applied to personal recommendations [10]. It is reported that, despite of simplicity, the method performs much better than most commonly used ranking method, such as global ranking method and collaborative filtering [9]. We also have got an initial but encouraging result using the method for query suggestions [11]. The work here follows the idea of [11]. BNRA method has three prominent features, namely asymmetric, parallelable and 'content oblivious'. In this paper, we verify the validity of the method for extracting semantic relations from query logs. We also analyze large query semantic networks constructed with the asymmetric relations.

2 Previous Work

There has been much related work on extracting query semantic relations, most of which is related to query clustering, classification, recommendation, expansion

or reformulation, and is usually carried out on bipartite networks constructed from query logs with one node set containing only queries and the other URLs. Various methods compute query relations according to the similarity between returned documents [12,13], selected documents [14,6] or snippets of returned results [15]. Most of them reported satisfactory results, but unfortunately are not applicable for large-scale documents due to unacceptably massive calculations.

Beeferman and Berger [1] proposed a 'content oblivious' method to generate a list of related query formulations for input queries by merging most related queries or URLs in query logs alternately, where the relations are measured in terms of the number of overlapped neighbors in bipartite networks. Wen et al. [3] proposed a better-designed solution for query clustering by combining content-based and link-based clustering together and using four notions of query relations, i.e. keywords or phrases in queries, substring matching of queries, common selected URLs and the similarity of selected documents. A method based on association rule was also proposed to discover related queries from a set of search transactions or sessions, where each session includes a sequence of input queries by a single user in a certain time interval [16]. One apparent disadvantage of the method is that it could merely find related queries submitted by the same user while incapable to extract most related queries submitted by different users. Query relations can also be explored by mapping queries to predefined topic categories like Broder's [17] informational/navigational/transactional taxonomy [18], geographical-locality-based categories [19] or other artificial categories [4,5]. Query classification brings great improvements to search engines, but on the other hand confines query relations into certain predefined categories. Baeza-Yates [20] described several relations between queries based on different information sources, i.e. keywords of queries, selected URL covers as well as hyperlinks or terms in selected web pages, and different semantic networks were defined based on these relations, among which the relations based on selected URL covers were qualified to be of the highest semantic strength.

A crucial common drawback of above methods is that the extracted query relations are symmetric, while asymmetric semantic relations are ubiquitous and more conforming to real world. The most relevant work to this notion was done by Baeza-Yates [2], where asymmetric query relations were extracted from Query-URL bipartite network based on selected URL covers. However, these asymmetric relations are associated from query q_i to q_j only when the URL set selected from q_i is completely covered by that from q_j, which restricts the extraction capability of asymmetric relations. BNRA, in contrast, is capable to extract and quantify asymmetric relations in a more natural and flexible manner. In subsequent sections, we will systematically investigate detailed properties of BNRA, including the recursive BNRA and its convergence, tunable parameters, etc.

3 BNRA Method

In order to implement BNRA, we need to construct a weighted Query-URL bipartite network from query logs. The click frequencies from queries to URLs

suggest the matching degree between search intentions behind queries and semantics behind URLs. Hence, it is essential to assign weight to each edge between query and URL based on click frequency. Denoting the query set as $Q = \{q_1, q_2, ..., q_n\}$ and the URL set as $U = \{u_1, u_2, ..., u_m\}$, the bipartite network could be described by an $n \times m$ adjacent matrix $A = a_{ij}$, where $a_{ij} > 0$ if u_j is clicked under submitted q_i, indicating the click frequency, and $a_{ij} = 0$ otherwise.

3.1 Method Description

BNRA is elaborated as follows. To find related queries for query q_i based on the network and quantify their relevance strength, we initially assign resource value f_i to query q_i, indicating the semantic information kept by q_i. Afterwards, the resource-allocation is processed in two steps. Firstly, the resources in query nodes (Initially only q_i keeps resource.) are proportionally distributed, in terms of corresponding edge weight, to their neighbor URL nodes. Whereafter, the resources in URL nodes, are proportionally propagated to their neighbor query nodes in reverse. The final resources located in a subset of query nodes, denoting as R_i, are regarded as the distribution of semantic information of q_i and indicate the relevance strength between q_i and the queries in R_i. The relevance strength from query q_i to $q_j \in R_i$, denoting as r_{ij}, reads

$$r_{ij} = f_i \times s_{ij} \tag{1}$$

$$s_{ij} = \frac{1}{k(q_i)} \sum_{l=1}^{m} \frac{a_{il} a_{jl}}{k(u_l)} \tag{2}$$

where $k(q_i) = \sum_{j=1}^{n} a_{ij}$ and $k(u_l) = \sum_{j=1}^{m} a_{jl}$ are weighted degrees of query q_i and URL u_l. Denoting strength matrix as $S = (s_{ij})_{n \times n}$, and initial resource distribution in query set as the row vector $\mathbf{f}^{(0)} = (f_1, f_2, ..., f_n)$, the final resource distribution is $\mathbf{f}^{(1)} = \mathbf{f}^{(0)} \cdot S$. In matrix S, the ith row indicates the resource distribution in queries originated from query q_i after resource allocation. S has the property that the sum of all values in each row is equal to unity 1, namely, $\sum_{j=1}^{n} s_{ij} = 1, \forall i = 1, ..., n$.

3.2 Computational Complexity

BNRA introduces high efficiency in both space and time. Denoting the edge number as e and the maximum degree of queries or URLs as k_{max}, BNRA for all queries requires $O(nk_{max}^2)$ operations and simply $n \times m$ memory for storing a bipartite network. In contrast, the classical agglomerative clustering method [1] requires $O((n+m)k_{max}^2 + e(4k_{max}))$ operations and $n+m+n^2+m^2$ memory for storing query similarities and URL similarities besides a bipartite network. An advanced method based on intersections of selected URL sets [2] requires $O(n^2 k_{max})$ operations. Therefore, BNRA is more efficient than methods mentioned. Moreover, compared to the agglomerative clustering method, BNRA

could be implemented in a parallel manner with ease, which serves as a defining utility for commercial search engines in mining large-scale user logs. On the other hand, it could extract and quantify query relations more flexibly than the method proposed in [2].

3.3 Recursive BNRA and Its Convergence

It is a natural conjecture that the resource allocation process in BNRA could be executed recursively as $\mathbf{f}^{(t+1)} = \mathbf{f}^{(t)} \cdot S = \mathbf{f}^{(0)} \cdot S^t$, where $\mathbf{f}^{(t)}$ indicates the resource distribution after the tth iteration. Such approach might extend the method to a diffusion-like algorithm which would converge to a stable solution of equation $\mathbf{f}^* = \mathbf{f}^* \cdot S$ mentioned by Zhou et al. [21] without further discussion. Here, we elaborate more detailed analysis.

If we regard all queries as the states transiting from one to another according to the corresponding transition probabilities, and S as the transition probability matrix, the recursive resource allocation process is in nature a Markov process among queries [22]. According to Markov Process Theory, if the Markov chain is irreducible and aperiodic, there is a unique stationary distribution $\mathbf{f}^*$, and S^t converges to a rank-one matrix in which each row is the stationary distribution, that is

$$\lim_{t \to \infty} S^t = \mathbf{1} \cdot \mathbf{f}^* \tag{3}$$

where $\mathbf{1}$ is the column vector with each value equal to 1.

In practice, a query-URL bipartite network can be composed of one large component and many small components. The set of queries within one connected component is a communicating class thus the corresponding Markov chain is irreducible. The bipartite network component is connected based on complicated behaviors of users. Therefore, the Markov chain is aperiodic. As Eq. (3) suggests, originally from any query in one component, when reach the unique stationary state, the resource distribution $\mathbf{f}^*$ is uniform, only determined by the topology of the bipartite network and has nothing to do with the initial resource distribution. The feature indicates on one hand the recursive process can expand the related queries effectively; on the other hand it may lower the relevance strength with the original query and strengthen the effect of global popularity, which is a trade off between the relevance specificity and global popularity.

3.4 Tunable Parameters

Two tunable parameters may effect the performance of BNRA. One is the iteration number t. As mentioned above, the iteration times can effect the range of related queries and the resolving power on relevance strength between queries. The other parameter is the resource allocation strategy. A naive strategy is to allocate the resource according to click frequency as shown in Eq. (1)(2). A more complicated form is

$$s_{ij} = \frac{1}{k(q_i)} \sum_{l=1}^{m} \frac{a_{il}^{\alpha} a_{jl}^{\alpha}}{k(u_l)} \tag{4}$$

$$k(q_i) = \sum_{j=1}^{m} a_{ij}^{\alpha}, k(u_l) = \sum_{j=1}^{n} a_{jl}^{\alpha} \tag{5}$$

where α is a tunable parameter controlling the force of click frequency on resource allocation, comprising the condition in Eq. (1)(2) when $\alpha = 1$.

Next, we will verify BNRA in query logs of Chinese search engine obtained from Sogou Labs and inspect the effect of the two parameters. Before coming to the details of the experiments and evaluation, we introduce the user log dataset in advance.

4 Experiment and Evaluation

4.1 Query Log Dataset

In our experiment, we use the query logs in the first week of March 2007 from Sogou Labs. Sogou Labs, founded by Chinese commercial search engine Sogou, consist of various web search resources of Sogou including query logs in one month, which can be accessed from `http://www.sogou.com/labs/`. There are 10,046,246 inquiry instances, 1,310,135 unique queries, 980,395 keywords and 4,055,171 unique URLs in this query log, where we refer *query* to a string submitted to search engines by a user which may contains one or more *keywords* delimited by white spaces, and *query instance* to one click behavior from a query to a URL. The number of keywords in queries mostly ranges from 1 to 3, and most of all keywords consist of 2 to 6 Chinese characters. Due to the shortness of keyword length, query relations extracted via keyword substring matching may be sparse and the performance will be greatly limited.

We apply BNRA and filter out the queries and URLs occurred only once in order to reduce noises. The constructed bipartite network contains 834,107 unique queries and 886,702 URLs. For each query, we assign resource $f_i = 100$, execute the resource allocation process with only one iteration and record top nine related queries. Table 1 shows some examples where related queries are listed according to the relevance strength in reverse order. For '小说网', '小说' (The English translations of these Chinese queries can be found in appendix, and hereinafter the same.) is positioned at the first place with strength 28.96. While for '小说', '小说网' is positioned at the last place with strength 1.44.

Table 1. Examples of related queries extracted by our method

Query	Related Queries
小说网	**小说**, 玄幻小说, 小说阅读网, xiaoshuo, 言情小说, 小說, 免费小说, 中文小说网, 言情
小说	玄幻小说, 起点, 小说阅读网, 言情小说, 潇湘书院, 起点中文网, **小说网**, 幻剑书盟, 起点中文

4.2 Evaluation

Most commercial search engines recommend queries they consider to be related to the original query as related search. We compare our method with recommended results by commercial search engines, i.e. Baidu (`www.baidu.com`) and Google (`www.google.cn`). As shown in Table 2, we compare the recommended queries of query '文学'. In most cases, the two search engines recommend those queries which contain the original query as substring. On the contrary, BNRA could extract related queries with no common substrings, which extends the scope widely. For example, the first query '榕树下' recommended by BNRA is the largest website of Chinese original literature, having no common substring with '文学'.

Users' perception indicates the performance of search engines to some extent. Therefore, we use editors' ratings to evaluate the performance of BNRA. We randomly select about 180 recommended queries and ask editors to rank these queries from 5 to 0, where 5 means very good and 0 means totally unrelated. All rating data can be accessed through `http://nlp.csai.tsinghua.edu.cn/~lzy/qr/rqj.zip`. In Fig. 1 we show the average scores of Baidu and BNRA with different iteration numbers. Despite of the disagreement among editors, the performance of BNRA is comparable with Baidu, which demonstrates the method is feasible and effective. Fig. 1 also suggests the loss of the specific relevance with original queries during the recursive resource allocation process. By Fig. 2, we show the average scores considering different numbers of recommended queries in the experiment with one iteration and the plot is skewed which is consistent with the decline of the relevance strength quantified by BNRA.

The agglomerative clustering method [1] was also performed on the dataset. Since the method requires huge memory, we compressed the bipartite network by filtering out the queries with unique click frequency lower than 10. The method iteratively merges the most related query pair and URL pair alternately until a termination condition applies. One reasonable termination condition proposed by Beeferman and Berger [1] is

$$\max_{q_i,q_j \in Q} \sigma(q_i, q_j) = 0 \text{ and } \max_{u_i,u_j \in U} \sigma(u_i, u_j) = 0$$

where $\sigma(*, *)$ indicates the fraction of common neighbors of two queries or URLs. However, it makes no sense for finding related queries because it leads to find the connected components where the queries distribute extremely imbalanced

Table 2. Recommended queries to '文学' from BNRA, Baidu and Google

Source	Related Queries
BNRA	**榕树下**,玄幻小说,世纪文学,成人小说,小说,原创文学,文学小说,读书,天地文学
Baidu	世纪文学,文学屋,天翼文学,吾爱文学网,起点文学,文学家,成人文学,晋江文学网,晋江文学
Google	世纪文学,吾爱文学网,星辰变世纪文学,79文学网,天地文学,文学城,艳情文学,极品家丁世纪文学,文学殿堂,文学屋

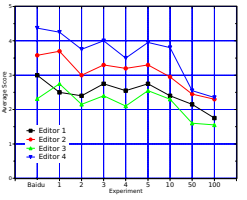


Fig. 1. Average scores of recommended queries by four editors

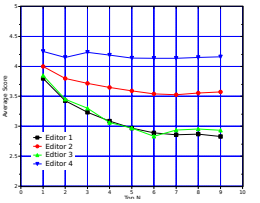


Fig. 2. Average scores of different numbers of recommended queries

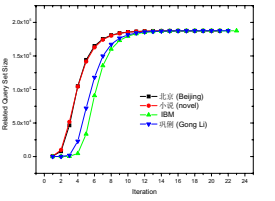


Fig. 3. Related query set size growth with iterations

and most are in several large components. Besides, the agglomerative clustering method is time-consuming. In a PC with *Intel Duo* 2.80*GHz* CPU and 1.5GB memory, it spent about 400 minutes on the compressed bipartite network constructed by the queries with unique click frequency more than 10. For BNRA, however, it spent no more than 1 minute to deal with the bipartite network constructed by the queries occurred more than once. In addition, the optimal termination condition of hierarchical agglomerative clustering algorithms is not resolved efficiently [23], so it is hard for the agglomerative clustering method to find an optimal solution.

4.3 Parameter Effects

In this subsection, we inspect the effect on BNRA of iteration numbers and α in Eq. 4. In Fig. 3 we illustrate the size changes of related queries for several queries after each iteration until finding the whole connected components. We also show the changes of top five related queries of query '小说' during the first four iterations in Table 3 with no dramatic changes found for the top related queries.

In order to track the changes during the iterations, we use Euclidean distance to measure the variation between two adjacent resource distributions. Four queries' variation dynamics along with iterations is shown in Fig. 4. Each of them terminates until the variation is less than 0.1. In in Fig. 5, we also illustrate the variation in each iteration of several semantic-free query pairs within one connected component of bipartite network, which indicates the trend towards convergence.

Table 3. Top 5 related queries of query '小说' after 1 to 4 iterations. The values in the brackets are relevance strength.

Iteration	Related Queries
1	玄幻小说(23.8), 起点(6.5), 小说阅读网(5.2), 言情小说(4.9), 潇湘书院(1.9)
2	玄幻小说(28.6), 起点(6.7), 言情小说(4.8), 小说阅读网(3.3), 起点中文网(2.2)
3	玄幻小说(30.1), 起点(6.4), 言情小说(4.6), 小说阅读网(2.4), 起点中文网(2.2)
4	玄幻小说(30.4), 起点(5.9), 言情小说(4.4), 起点中文网(2.1), 小说阅读网(2.0)

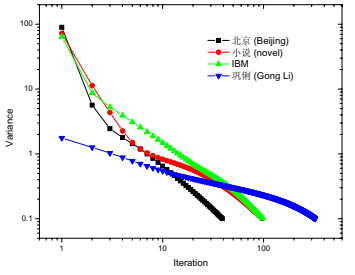


Fig. 4. Log-log plots of four queries' variation dynamics along with iterations

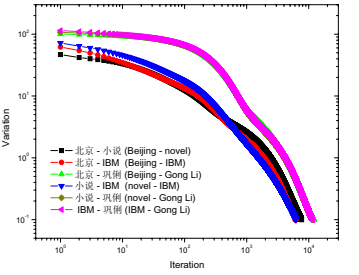


Fig. 5. Log-log plots of the variation along with iterations of several semantic-free query pairs

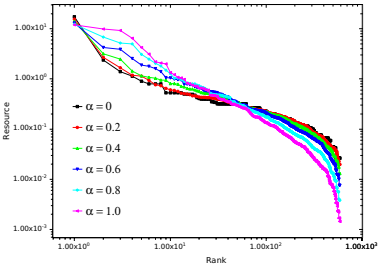


Fig. 6. Log-log plots of the resource distribution of query '北京' (Beijing)

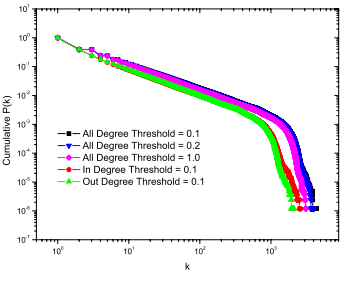


Fig. 7. Log-log plots of cumulative degree distributions of query networks

Another tunable parameter is the α in Eq. 4 which effects the resource allocation strategy. In Fig. 6, we show the resource distributions of query '北京' (Beijing) after 1 iteration with α varying from 0 to 1.0 stepped by 0.2. When $\alpha \in [0, +\infty)$, the smaller the parameter α is, the weaker the relevance between the resource allocation and the click frequency will be. If $\alpha = 0$, the resource will be allocated equally. As α grows, the variance of distributed resources increases. When $\alpha = 1.0$, the resource begins to be allocated totally according to the click frequency.

5 Semantic Networks of Queries

It is straightforward to build query semantic networks via BNRA efficiently. Through the semantic networks we can get much more information among queries. We run one iteration of BNRA for each query and construct a directed and weighted query semantic network by connecting each query to its related queries with threshold $\lambda = 0.1$ which is to discard the related queries with the allocated resource $f_i < 0.1$.

Table 4. Properties of query semantic network under threshold $\lambda = 0.1$

Property	Value
Node Number	834,107
Arc Number	4,735,880
Average All Degree	11.355
Average In/Out Degree	5.678
Average Path Length (Directed)	7.609
Average Path Length (Undirected)	7.231
Clustering Coefficient	0.527
Component Number	556,900
γ	0.915/7.867

Table 5. Examples of the paths on related queries

Paths on Related Queries
巩俐 → 张艺谋 → 章子怡 → 艺妓回忆录
雅虎 → Yahoo → www.yahoo.com.cn → 雅虎中国
百事可乐 → 可乐 → 可口可乐 → www.icoke.cn → icoke

Some properties of the network constructed under threshold $\lambda = 0.1$ are shown in Table 4. Fig. 7 shows the cumulative degree distributions of query networks constructed under different thresholds, and all of them follow power law in the rough, namely $P_c(k) \propto k^{-\gamma}$ where $P_c(k)$ is the cumulative degree distribution, and decay into two parts noticeably which indicates the lack of high degree nodes. All the degree distributions stay stable when the threshold varies from 0.1 to 1.0. The networks show definite small world phenomenon indicating shorter average path length and higher clustering coefficient than the random network of the same size, and scale free effect indicating that the degree distribution follows power law [24]. As shown in Table 5, we display some paths on related queries, which indicates the semantic shift in a sense.

6 Conclusion and Future Work

An asymmetric method was proposed for extracting and quantifying query semantic relations based on network resource allocation using user logs which is simple to implement with low computational cost. We investigated properties of BNRA and found the naive method with only one iteration and allocating resource by click frequency is good enough for relation extraction. The method is not only 'content oblivious', but also can be easily implemented in a paralleled manner. Possible future work includes: 1) the content based method, such as the common substring method used by Baidu and Google, is expected to be combined with link analysis to achieve more improvement; and 2) more rigorous evaluation will be designed by monitoring the real users choices.

Acknowledgements

This work is supported by the National Science Foundation of China under Grant No. 60621062, 60873174 and the National 863 High-Tech Project under Grant No. 2007AA01Z148. We also thank Peng Li, Qiaxia Jiang and Shaohua Teng for coding work and discussion.

References

1. Beeferman, D., Berger, A.: Agglomerative clustering of a search engine query log. In: Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining (2000)
2. Baeza-Yates, R., Tiberi, A.: Extracting semantic relations from query logs. In: Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining (2007)
3. Wen, J.R., Jian-Yun, N., Hong-Jiang, Z.: Query clustering using user logs. ACM Transactions on Information Systems 20(1) (2002)
4. Shen, D., Pan, R., Sun, J.T., Pan, J.J., Wu, K., Yin, J., Yang, Q.: Query enrichment for web-query classification. ACM Transactions on Information Systems 24(3), 320–352 (2006)
5. Beitzel, S.M., Jensen, E.C., Lewis, D.D., Chowdhury, A., Frieder, O.: Automatic classification of web queries using very large unlabeled query logs. ACM Transactions on Information Systems 25(2), 9 (2007)
6. Baeza-Yates, R., Hurtado, C., Mendoza, M.: Query recommendation using query logs in search engines. In: Workshops on current trends in database technology of 9th international conference on extending database technology (2004)
7. Chirita, P.A., Firan, C.S., Nejdl, W.: Personalized query expansion for the web. In: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval, pp. 7–14 (2007)
8. He, X.F., Yan, J., Ma, J.W., Liu, N., Chen, Z.: Query topic detection for reformulation. In: Proceedings of the 16th international conference on World Wide Web, pp. 1187–1188 (2007)
9. Zhou, T., Ren, J., Medo, M., Zhang, Y.C.: Bipartite network projection and personal recommendation. Physical Review E 76(4) (2007)
10. Deshpande, M., Karypis, G.: Item-based top-n recommendation algorithms. ACM Transactions on Information Systems 22(1) (2004)
11. Liu, Z.Y., Sun, M.S.: Asymmetrical query recommendation method based on bipartite network resource allocation. In: Proceedings of the 17th international conference on World Wide Web, Beijing (2008)
12. Raghavan, V.V., Sever, H.: On the reuse of past optimal queries. In: Proceedings of the 18th annual international ACM SIGIR conference on research and development in information retrieval, pp. 344–350 (1995)
13. Fitzpatrick, L., Dent, M.: Automatic feedback using past queries: social searching? In: Proceedings of the 20th annual international ACM SIGIR conference on research and development in information retrieval, pp. 306–313 (1997)
14. Baeza-Yates, R., Hurtado, C., Mendoza, M.: Query clustering for boosting web page ranking. In: Advances in Web Intelligence, pp. 164–175 (2004)
15. Sahami, M., Heilman, T.D.: A web-based kernel function for measuring the similarity of short text snippets. In: Proceedings of the 15th international conference on World Wide Web, pp. 377–386 (2006)

16. Fonseca, B.M., Golgher, P.B., de Moura, E.S., Ziviani, N.: Using association rules to discover search engines related queries. In: Proceedings of the first conference on Latin American Web Congress, pp. 66–71 (2003)
17. Broder, A.: A taxonomy of web search. ACM SIGIR Forum 36(2), 3–10 (2002)
18. Kang, I.H., Kim, G.C.: Query type classification for web document retrieval. In: Proceedings of the 26th annual international ACM SIGIR conference on research and development in informaion retrieval, pp. 64–71 (2003)
19. Gravano, L., Hatzivassiloglou, V., Lichtenstein, R.: Categorizing web queries according to geographical locality. In: Proceedings of the 12th international conference on information and knowledge management, pp. 325–333 (2003)
20. Baeza-Yates, R.: Graphs from search engine queries. In: van Leeuwen, J., Italiano, G.F., van der Hoek, W., Meinel, C., Sack, H., Plášil, F. (eds.) SOFSEM 2007. LNCS, vol. 4362, pp. 1–8. Springer, Heidelberg (2007)
21. Zhou, T., Jiang, L.L., Su, R.Q., Zhang, Y.C.: Effect of initial configuration on network-based recommendation. Europhysics Letters 81(5), 58004 (2008)
22. Ross, S.M.: Introduction to Probability Models, 9th edn. Academic Press, Inc., Orlando (2006)
23. Kapp, A.V., Tibshirani, R.: Are clusters found in one dataset present in another dataset? Biostatistics 8(1), 9–31 (2007)
24. Newman, M.E.J.: The structure and function of complex networks. SIAM Review 45(2), 167–256 (2003)

Appendix: Translations of Chinese Queries

For Table 1 and 3: 小说(novel), 玄幻小说(fantasy novel), xiaoshuo(Chinese Pinyin of 'novel'), 言情小说(romantic fiction), 小說(traditional Chinese of 'novel'), 免费小说(novels for free), 言情(romance), 小说阅读网/中文小说网/起点/小说阅读网/潇湘书院/起点中文网/小说网/幻剑书盟/起点中文(names of some Chinese novel websites).

For Table 2: 玄幻小说(fantasy novel), 成人小说(adult fiction), 成人文学(adult literature), 小说(novel), 原创文学(original literature), 文学小说(literature and novel), 读书(reading), 文学家(writers), 艳情文学(erotic literature), 榕树下/世纪文学/天地文学/文学屋/天翼文学/吾爱文学网/起点文学/晋江文学网/晋江文学/星辰变世纪文学/79文学网/天地文学/ 文学城/极品家丁世纪文学/文学殿堂/文学屋(names of some Chinese literature websites).

For Table 5: 巩俐(Gong Li, a Chinese famous actress cooperated with Zhang Yimou), 张艺谋(Zhang Yimou, a Chinese famous director), 章子怡(Zhang Ziyi, a Chinese famous actress cooperated with Zhang Yimou), 艺妓回忆录(Memoirs of a Geisha, a movie starring Zhang Ziyi), 雅虎(Yahoo!), 雅虎中国(Yahoo! China), 百事可乐(Pepsi), 可乐(Cola), 可口可乐(Coca Cola).

Acquiring Semantic Relations Using the Web for Constructing Lightweight Ontologies

Wilson Wong, Wei Liu, and Mohammed Bennamoun

School of Computer Science and Software Engineering
University of Western Australia
Crawley WA 6009
{wilson,wei,bennamou}@csse.uwa.edu.au

Abstract. Common techniques for acquiring semantic relations rely on static domain and linguistic resources, predefined patterns, and the presence of syntactic cues. We propose a hybrid approach which brings together established and novel techniques in lexical simplification, word disambiguation and association inference for acquiring coarse-grained relations between potentially ambiguous and composite terms using only dynamic Web resources. Our experiments using terms from two different domains demonstrate potential preliminary results.

1 Introduction

Relation acquisition is an important aspect of ontology learning. Traditionally, semantic relations are either extracted as verbs based on grammatical structures [1], induced through term co-occurrence using large text corpora [2], or discovered in the form of unnamed associations through cluster analysis [3]. Challenges faced by conventional techniques include 1) the reliance on static patterns and text corpora together with rare knowledge resources, 2) the need for named entities to guide relation acquisition, 3) the difficulty in classifying composite or ambiguous names into the required categories, and 4) the dependence on grammatical structures and the presence of verbs can result in the overlooking of indirect, implicit relations. In recent years, there is a growing trend in relation acquisition using Web resources such as Wikipedia [4] and online ontologies (e.g. Swoogle) [5] to partially address the shortcomings of conventional techniques.

In this paper, we propose a hybrid approach which integrates lexical simplification, word disambiguation and association inference for acquiring semantic relations using only the Web (i.e. Wikipedia and web search engines) for constructing lightweight domain ontologies. Our approach performs an iterative process of *term mapping* and *term resolution* to identify coarse-grained relations between domain terms. The main contribution of this paper is the resolution phase which allows our relation acquisition approach to handle complex and ambiguous terms, and terms not covered by our background knowledge on the Web. Our approach can be used to complement conventional techniques for acquiring fine-grained relations and to automatically extend knowledge resources

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 266–277, 2009.
© Springer-Verlag Berlin Heidelberg 2009

such as Wikipedia. The rest of the paper is structured as follows. Section 2 and 3 present existing work related to relation acquisition, and the detail of our approach, respectively. The outcome of the initial experiment is summarised in Section 4. We conclude this paper in Section 5.

2 Related Work

Techniques for relation acquisition can be classified as symbolic-based, statistics-based or a hybrid of both. The use of linguistic patterns enables the discovery of fine-grained semantic relations. For instance, Poesio & Almuhareb [6] developed specific lexico-syntactic patterns to discover named relations such as *part-of* and *causation*. However, linguistic-based techniques using static rules tend to face difficulties in coping with the structural diversity of a language. The technique by Sanchez & Moreno [1] for extracting verbs as potential named relations is restricted to handling verbs in simple tense and verb phrases which do not contain modifiers such as adverbs. In order to identify indirect relations, statistics-based techniques such as co-occurrence analysis and cluster analysis are necessary. Co-occurrence analysis makes use of redundancy in large domain corpora to detect the presence of statistically significant associations between terms. However, the textual resources required by such techniques are difficult to obtain, and remain static over a period of time. For example, Schutz & Buitelaar [2] manually constructed a domain corpus in the *"football"* domain containing only $1,219$ documents from an online football site for relation acquisition. Cluster analysis [3], on the other hand, requires tremendous computational effort in preparing features from texts for similarity measurements. The lack of emphasis on indirect relations is also evident in existing techniques. Many relation acquisition techniques in information extraction acquire semantic relations with the guidance of named entities [7]. Relation acquisition techniques which require named entities have restricted applicability since many domain terms with important relations cannot be easily categorised. In addition, the common practice of extracting triples using only patterns and grammatical structures tends to disregard relations between syntactically unrelated terms.

In view of the shortcomings of conventional techniques, there is a growing trend in relation acquisition which favours the exploration of rich, heterogeneous resources on the Web over the use of static, rare background knowledge. *SCARLET* [5], which stemmed from a work in ontology matching, follows this paradigm by harvesting online ontologies on the Semantic Web to discover relations between concepts. Sumida et al. [4] developed a technique for extracting a large set of hyponymy relations in Japanese using the hierarchical structures of Wikipedia. There is also a group of researchers who employ Web documents as input for relation acquisition [8]. Similar to the conventional techniques, this group of work still relies on the ubiquitous WordNet and other domain lexicons for determining the proper level of abstraction and labeling of relations between the terms extracted from Web documents. Pei et al. [9] employed predefined local

(i.e. WordNet) and online ontologies to name the unlabeled associations between concepts in Wikipedia. The labels are acquired through a mapping process which attempts to find lexical matches for Wikipedia concepts in the predefined ontologies. The obvious shortcomings include the inability to handle complex and new terms which do not have lexical matches in the predefined ontologies.

3 A Hybrid Approach for Relation Acquisition

Our relation acquisition approach is composed of two phases, namely, *term mapping* and *term resolution*. The input is a set of domain terms T produced using a separate *term recognition* technique. The inclusion of a resolution phase sets our approach apart from existing techniques which employ resources on the Web for relation acquisition. This resolution phase allows our approach to handle complex and ambiguous terms, and terms which are not covered by the background knowledge on the Web. Figure 1 provides an overview of our approach.

In this approach, Wikipedia is seen as a directed acyclic graph W where vertices V are topics covered by Wikipedia, and edges E are three types of coarse-grained relations between the topics, namely, *hierarchical* H, *associative* A, and *polysemous* P, or $E = H \cup A \cup P$. It is worth noting that H, A and P are disjoint sets. These coarse-grained links are obtained from Wikipedia's classification scheme, *"See Also"* section, and disambiguation pages, respectively. The term mapping phase creates a subgraph of W for each set T, denoted as W_T by recursively querying W for relations that belong to the terms $t \in T$. The querying aspect is defined as the function $map(t)$, which finds an equivalent topic $u \in V$ in W for term t, and returns the closed neighbourhood N_t:

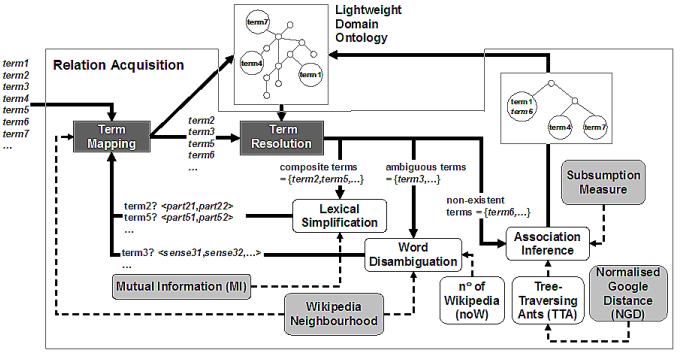


Fig. 1. An overview of our hybrid approach. The main phases are *term mapping* and *term resolution*, represented by black rectangles. The three steps involved in resolution are *simplification*, *disambiguation* and *inference*. The rounded rectangles represent techniques and resources used in this approach. The techniques represented by the white rounded rectangles were developed by the authors, while existing techniques and resources are shown using grey rounded rectangles.

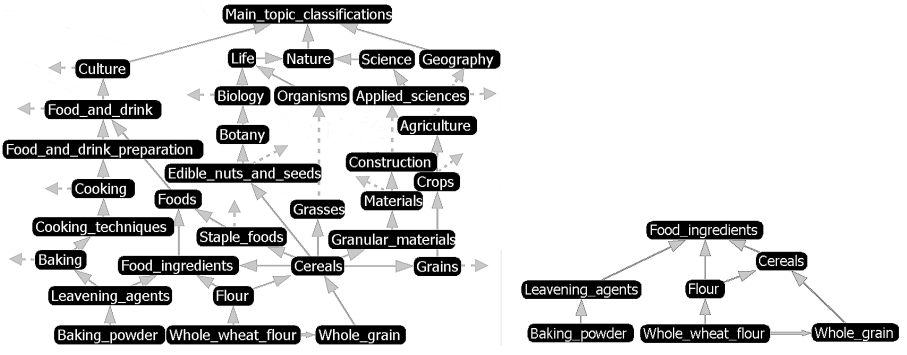
Algorithm 1. termMap($t, W_T, M, root, iteration$)

```
1: rslt := map(t)
2: if iteration equals to 1 then
3:    if rslt equals to undef then
4:       if t is multi-word then return composite
5:       else return non-existent
6:    else if rslt equals to N_t = (V_t, E_t) ∧ P_t ≠ φ then
7:       return ambiguous
8:    else if rslt equals to N_t = (V_t, E_t) ∧ P_t = φ then
9:       add neighbourhood N_t to the subgraph W_T and iteration ← iteration + 1
10:      for each u ∈ V_t where (t, u) ∈ H_t ∪ A_t do
11:         termMap(u, W_T, M, root, iteration)
12:      M ← M ∪ {t}
13:      return mapped
14: else if iteration more than 1 then
15:   if rslt equals to N_t = (V_t, E_t) ∧ P_t = φ then
16:      add neighbourhood N_t to the subgraph W_T and iteration ← iteration + 1
17:      for each u ∈ V_t where (t, u) ∈ H_t do
18:         if u not equal to root then termMap(u, W_T, M, root, iteration)
19:         else return // all paths from the origin t will arrive at the root
```

$$map(t) = \begin{cases} N_t = (V_t, E_t) & \text{if } (\exists u \in V, u \equiv t) \\ \texttt{undef} & \text{otherwise} \end{cases} \quad (1)$$

The neighbourhood for term t is denoted as (V_t, E_t) where $E_t = \{(t, y) : (t, y) \in H_t \cup A_t \cup P_t, y \in V_t\}$ and V_t is the set of vertices in the neighbourhood. The sets H_t, A_t and P_t contain hierarchical, associative and polysemous links which connect term t to its adjacent terms $y \in V_t$. The process of term mapping is summarised in Algorithm 1. The term mapper in Algorithm 1 is invoked once for every $t \in T$. The term mapper ceases the recursion upon encountering the base case, which consists of the root vertices of Wikipedia (e.g. *"Main topic classifications"*). An input term $t \in T$ which is traced to the root vertex is considered as successfully mapped, and is moved from set T to set M. Figure 2(a) shows the subgraph W_T created for the input set T={*'baking powder','whole wheat flour'*}. In reality, many terms cannot be straightforwardly mapped because they do not have lexically equivalent topics in W due to the non-exhaustive coverage of Wikipedia, the tendency to modify terms for domain-specific uses, and the polysemous nature of certain terms. The term mapper in Algorithm 1 returns different values, namely, `composite`, `non-existent` and `ambiguous` to indicate the causes of mapping failures. The term resolution phase resolves mapping failures through the iterative process of lexical simplification, word disambiguation and association inference. Upon the completion of mapping and resolution of all input terms, any direct or indirect relations between the mapped terms $t \in M$ can be identified by finding paths which connect them in the subgraph W_T.

Finally, we devise a 2-step technique to transform the subgraph W_T into a lightweight domain ontology. Firstly, we identify the *nearest common ancestor*



(a) The dotted arrows represent additional hierarchical links from each vertex. The only associative link is between *"whole wheat flour"* and *"whole grain"*.

(b) *"Food ingredients"* is the NCA.

Fig. 2. Figure 2(a) shows the subgraph W_T constructed for T={*'baking powder'*,*'whole wheat flour'*} using Algorithm 1, which is later pruned to produce a lightweight ontology in Figure 2(b)

(NCA) for the mapped terms. The discussion on our simple algorithm for finding NCA is omitted due to space constraints. Secondly, we identify all directed paths in W_T which connect the mapped terms to the new root NCA and use those paths to form the final lightweight domain ontology. The lightweight domain ontology for the subgraph W_T in Figure 2(a) is shown in Figure 2(b). We discuss the details of the three parts of term resolution in the following three subsections.

3.1 Lexical Simplification

The term mapper in Algorithm 1 returns the `composite` value to indicate the inability to map a *composite term* (i.e. multi-word term). Composite terms which have many modifiers tend to face difficulty during term mapping due to the absence of lexically equivalent topics in W. To address this, we designed a lexical simplification step to reduce the lexical complexity of composite terms in a bid to increase their chances of re-mapping. A composite term is comprised of a head noun altered by some pre- (e.g. adjectives and nouns) or post-modifiers (e.g. prepositional phrases). These modifiers are important in clarifying or limiting the extent of the semantics of the terms in a particular context. For instance, the modifier *"one cup"* as in *"one cup whole wheat flour"* is crucial for specifying the amount of *"whole wheat flour"* required for a particular pastry. However, the semantic diversity of terms created by certain modifiers is often unnecessary in a larger context. Our lexical simplifier make use of this fact to reduce the complexity of a composite term for re-mapping.

The lexical simplification step breaks down a composite term into two structurally coherent parts, namely, an *optional constituent* and a *mandatory constituent*. A mandatory constituent is comprised of but not limited to the head

x	y	n_x	n_y	$MI(x,y)$
one	cup whole wheat flour	4,710,000,000	125,000	0.0044
one cup	whole wheat flour	3,670,000	870,000	0.0008
one cup whole	wheat flour	4,270	2,690,000	0.1282
one cup whole wheat	flour	2,320	33,400,000	0.2355
one cup whole wheat flour, n_t		897		
x	y	n_x	n_y	$MI(x,y)$
salt	to taste	144,000,000	14,300,000	0.3699
salt to	taste	3,210,000	204,000,000	1.5266
salt to taste, n_t		1,790,000		

Fig. 3. The computation of mutual information for all pairs of contiguous constituents of the composite terms *"one cup whole wheat flour"* and *"salt to taste"*

noun of a composite term, and has to be in common use in the language independent of the optional constituent. The lexical simplifier then finds the least dependent pair as the ideally decomposed constituents. The dependencies are measured by estimating the mutual information of all contiguous constituents of a term. A term with n-words has $n-1$ possible pairs denoted as $< x_1, y_1 >, ..., < x_{n-1}, y_{n-1} >$. The mutual information for each pair $< x, y >$ of term t is computed as $MI(x,y) = f(t)/f(x)f(y)$ where f is a frequency measure. In a previous work [10], we utilise the page count returned by Google to compute the relative frequency required for mutual information. Given that $Z = \{t, x, y\}$, the relative frequency for each $z \in Z$ is computed as $f(z) = \frac{n_z}{n_Z} e^{(-\frac{n_z}{n_Z})}$ where n_z is the page count returned by web search engines, and $n_Z = \sum_{u \in Z} n_u$. Upon identifying the two least dependent constituents, we re-map the mandatory portion. To retain the possibly significant semantics delivered by the modifiers, we also attempt to re-map the optional constituents. If the decomposed constituents are in turn not mapped, another iteration of term resolution is performed. Unrelated constituents will be discarded. For this purpose, we define the distance of a constituent with respect to the set of mapped terms M as:

$$\delta(\{x, y\}, M) = \frac{\sum_{m \in M} noW(\{x, y\}, m)}{|M|} \tag{2}$$

where $noW(a, b)$ is a measure of geodesic distance between topic a and b based on Wikipedia developed by the authors [11] known as n° *of Wikipedia (noW)*. A constituent is discarded if $\delta(\{x, y\}, M) > \tau$ and the current set of mapped terms is not empty, $|M| \neq 0$. The threshold $\tau = \delta(M) + \sigma(M)$, where $\delta(M)$ and $\sigma(M)$ are the average and the standard deviation of the intra-group distance of M.

3.2 Word Disambiguation

The term mapping phase in Algorithm 1 returns the `ambiguous` value if a term t has a non-empty set of polysemous links P_t in its neighbourhood. In such cases, the terms are considered as ambiguous and cannot be directly mapped. To address this, we include a word disambiguation step which automatically resolves *ambiguous terms* using *noW* [11]. Since all input terms in T belong to the same

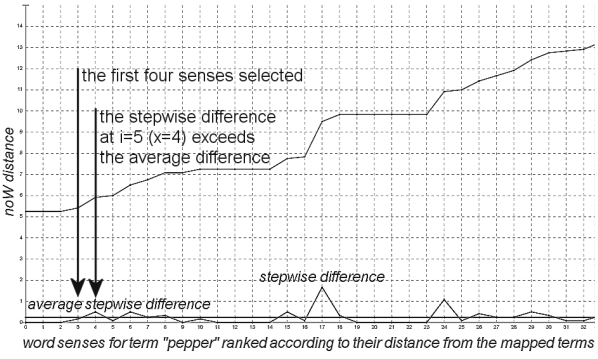


Fig. 4. A graph showing the distribution of *noW* distance and the stepwise difference for the sequence of word senses for the term *"pepper"*. The set of mapped terms is M={ *"fettuccine", "fusilli", "tortellini", "vinegar", "garlic", "red onion", "coriander", "maple syrup", "whole wheat flour", "egg white", "baking powder", "buttermilk"*}. The line *"stepwise difference"* shows the $\Delta_{i-1,i}$ values. The line *"average stepwise difference"* is the constant value μ_Δ. Note that the first sense s_1 is located at $x = 0$.

domain of interest, the word disambiguator finds the proper senses to replace the ambiguous terms by the virtue of the senses' relatedness to the already mapped terms. Senses which are highly related to the mapped terms have lower *noW* value. For example, the term *"pepper"* is considered as ambiguous since its neighbourhood contains a non-empty set P_{pepper} with numerous polysemous links pointing to various senses in the *"food ingredients"*, *"music"* and *"sports"* domains. If the term *"pepper"* is provided as input together with terms such as *"vinegar"* and *"garlic"*, we can eliminate all semantic categories except *"food ingredients"*. Each ambiguous term t has a set of senses $S_t = \{s : s \in V_t \wedge (t, s) \in P_t\}$. Equation 2, denoted as $\delta(s, M)$, is used to measure the distance between a sense $s \in S_t$ with the set of mapped terms M.

The senses are then sorted into a list $(s_1, ..., s_n)$ in ascending order according to their distance with the mapped terms. The smaller the subscript, the smaller the distance, and therefore, the closer to the domain in consideration. An interesting observation is that many senses for an ambiguous term are in fact minor variations belonging to the same semantic category (i.e. paradigm). Referring back to our example term *"pepper"*, within the *"food ingredients"* domain alone, multiple possible senses exist (e.g. *"sichuan pepper"*, *"bell pepper"*, *"black pepper"*). While these senses have their intrinsic differences, they are paradigmatically substitutable for one another. Using this property, we devise a *senses selection mechanism* to identify suitable paradigms covering highly related senses as substitutes for the ambiguous terms. The mechanism computes the difference in *noW* value as $\Delta_{i-1,i} = \delta(s_i, M) - \delta(s_{i-1}, M)$ for $2 \leq i \leq n$ between every two consecutive senses. We currently employ the average stepwise difference of the sequence as the cutoff point. The average stepwise difference for a list of n senses is $\mu_\Delta = \frac{\sum_{i=2}^{n} \Delta_{i-1,i}}{n-1}$. Finally, the first k senses in the sequence

with $\Delta_{i-1,i} < \mu_\Delta$ are accepted as belonging to a single paradigm for replacing the ambiguous term. Using this mechanism, we have reduced the scope of the term *"pepper"* to only the *"food ingredient"* domain out of the many senses across genres such as *"musician"* (e.g. *"pepper (band)"*) and *"beverage"* (e.g. *"dr pepper"*). In our example in Figure 4, the ambiguous term *"pepper"* is replaced by {*"black pepper"*, *"allspice"*, *"melegueta pepper"*, *"cubeb"*}. These $k = 4$ word senses are selected as replacements since the stepwise difference at point $i = 5$, $\Delta_{4,5} = 0.5$ exceeds $\mu_\Delta = 0.2417$.

3.3 Association Inference

Terms that are labelled as `non-existent` by Algorithm 1 simply do not have any lexical matches on Wikipedia. We propose to use cluster analysis to infer potential associations for such *non-existent terms*. We employ our term clustering algorithm with featureless similarity measures known as *Tree-Traversing Ant (TTA)* [11]. *TTA* is a hybrid algorithm inspired by ant-based methods and hierarchical clustering which utilises two featureless similarity measures, namely, *Normalised Google Distance (NGD)* [12] and *noW*. A detailed discussion on *TTA* is beyond the scope of this paper. Unlike conventional clustering algorithms which involve feature extraction and selection, terms are automatically clustered using *TTA* based on their usage prevalence and co-occurrence on the Web.

In this step, we perform term clustering on the non-existent terms together with the already mapped terms in M to infer hidden associations. The association inference step is based on the premise that terms grouped into similar clusters are bound by some common dominant properties. By inference, any non-existent terms which appear in the same clusters as the mapped terms should have similar properties. The *TTA* returns a set of term clusters $C = \{C_1, ..., C_n\}$ upon the completion of term clustering for each set of input terms. Each $C_i \in C$ is a set of related terms as determined by *TTA*. Figure 5 shows the results of clustering the non-existent term *"conchiglioni"* with 14 mapped terms. The output is a set of three clusters $\{C_{1'}, C_2, C_3\}$. Next, we acquire the parent topics of all mapped terms located in the same cluster as the non-existent

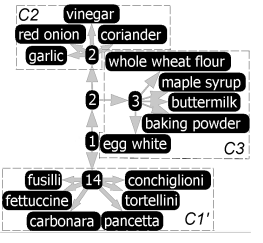


Fig. 5. The result of clustering the non-existent term *"conchiglioni"* and the mapped terms M={*"fettuccine"*, *"fusilli"*, *"tortellini"*, *"vinegar"*, *"garlic"*, *"red onion"*, *"coriander"*, *"maple syrup"*, *"whole wheat flour"*, *"egg white"*, *"baking powder"*, *"buttermilk"*, *"carbonara"*, *"pancetta"*} using *TTA*.

term by calling the mapping function in Equation 1. We refer to such cluster as *target cluster*. These parent topics, represented as the set R, constitute the potential topics which may be associated with the non-existent term. In our example in Figure 5, the target cluster is $C_{1'}$, and the elements of set R are { *"pasta", "pasta", "pasta", "italian cuisine", "sauces", "cuts of pork", "dried meat", "italian cuisine", "pork", "salumi"*}. We devise a *prevailing parent selection mechanism* to identify the most suitable parent in R to which we attach the non-existent term. The prevailing parent is determined by assigning a weight to each parent $r \in R$, and ranking the parents according to their weights. Given the non-existent term t and a set of parents R, the *prevailing parent weight* (ρ_r) where $0 \leq \rho_r < 1$ for each unique $r \in R$ is defined as $\rho_r = common(r)sim(r,t)subsume(r,t)\delta_r$ where $sim(a,b)$ is given by $1 - NGD(a,b)\theta$, and $NGD(a,b)$ is the *Normalised Google Distance* [12] between a and b. θ is a constant within the range $(0,1]$ for adjusting the NGD distance. The function $common(r) = \frac{\sum_{q \in R, q=r} 1}{|R|}$ determines the number of occurrence of r in set R. $\delta_r = 1$ if $subsume(r,t) > subsume(t,r)$ and $\delta_r = 0$ otherwise. The subsumption measure $subsume(x,y)$ is the probability of x given y computed as $n(x,y)/n(y)$, where $n(x,y)$ and $n(y)$ are page counts obtained from web search engines. This measure is used to quantify the extent of term x being more general than term y. The higher the subsumption value, the more general term x is with respect to y. Upon ranking the unique parents in R based on their weights, we select the prevailing parent r as the one with the largest ρ. A link is then created for the non-existent term t to hierarchically relate it to r.

4 Initial Experiments and Discussions

We experimented with a prototype of our approach shown in Figure 1 using two manually-constructed data sets, namely, a set of 11 terms in the *"genetics"* domain, and a set of 31 terms in the *"foods"* domain. The system performed the initial mappings of the input terms at level 0. This results in 6 successfully mapped terms and 5 unmapped composite terms in the *"genetics"* domain. As for the terms in the *"foods"* domain, 14 were mapped, 16 were composite and 1 was non-existent. At level 1, the 5 composite terms in the *"genetics"* domain were decomposed into 10 constituents where 8 were remapped and 2 required further level 2 resolution. The non-existent term in the *"foods"* domain was successfully clustered and remapped at level 1. The 16 composite terms were decomposed into 32 constituents where 14, 10, 5 and 3 were remapped, composite, non-existent and discarded, respectively. Figure 6 summarises the experiment results.

Overall, the system has a 100% precision in the aspect of term mapping, lexical simplication and word disambiguation in all two levels using the small set of 11 terms in the *"genetics"* domain as shown in Figure 6. As for the set of foods-related terms, there was one false positive (i.e. incorrectly mapped) involving the composite term *"100g baby spinach"* which results in an 80% precision in level 2. In level 1, this composite term was decomposed into the appropriate constituents *"100g"* and *"baby spinach"*. In level 2, the term *"baby spinach"* was

Level	Results		Genetics		Foods	
L0	mapped terms	correctly mapped	6	100%	14	100%
		incorrectly mapped	0		0	
	unmapped terms	composite terms	5		16	
		non-existent terms	0		1	
		ambiguous terms	0		0	
L1	composite terms from L0	correctly decomposed	5	100%	16	100%
		incorrectly decomposed	0		0	
	non-existent terms from L0	correctly clustered	0	-	1	100%
		incorrectly clustered	0		0	
	ambiguous terms from L0	correctly disambiguated	0	-	0	-
		incorrectly disambiguated	0		0	
	remapping at L1	correctly mapped	8	100%	15	100%
		incorrectly mapped	0		0	
	unsuccessful remapping at L1	composite terms	0		10	
		ambiguous terms	1		0	
		non-existent terms	1		5	
		discarded	0		3	
L2	composite terms from L1	correctly decomposed	0	-	10	100%
		incorrectly decomposed	0		0	
	non-existent terms from L1	no clustering for non-existent terms after L1	1	-	5	-
	ambiguous terms from L1	correctly disambiguated	1	100%	0	-
		incorrectly disambiguated	0		0	
	remapping at L2	correctly mapped	1	100%	4	80%
		incorrectly mapped	0		1	
	unsuccessful remapping at L2	composite terms	0		1	
		ambiguous terms	0		1	
		non-existent terms	0		2	
		discarded	1		16	

Level	Results		Genetics		Foods	
L3	composite terms from L2	correctly decomposed	0	-	1	100%
		incorrectly decomposed	0		0	
	non-existent terms from L2	no clustering for non-existent terms after L1	0	-	2	-
	ambiguous terms from L2	correctly disambiguated	0	-	1	100%
		incorrectly disambiguated	0		0	
	remapping at L3	correctly mapped	0	-	1	100%
		incorrectly mapped	0		0	
	unsuccessful remapping at L3	composite terms	0		0	
		ambiguous terms	0		1	
		non-existent terms	0		0	
		discarded	0		3	
L4	composite terms from L3	correctly decomposed	0	-	0	-
		incorrectly decomposed	0		0	
	non-existent terms from L3	no clustering for non-existent terms after L1	0	-	0	-
	ambiguous terms from L3	correctly disambiguated	0	-	1	100%
		incorrectly disambiguated	0		0	
	remapping at L4	correctly mapped	0	-	1	100%
		incorrectly mapped	0		0	
	unsuccessful remapping at L4	composite terms	0		0	
		ambiguous terms	0		0	
		non-existent terms	0		0	
		discarded	0		0	

Fig. 6. The results of relation acquisition using our approach for the *"genetics"* and *"foods"* domains. The labels *"correctly xxx"* and *"incorrectly xxx"* represent the true positives (TP) and false positives (FP). Precision is computed as $TP/(TP + FP)$.

further decomposed and its constituent *"spinach"* was successfully remapped. The constituent *"baby"* in this case refers to the adjectival sense of *"comparatively little"*. However, the modifier *"baby"* was inappropriately remapped and attached to the concept of *"infant"*. The lack of information on polysemes and synonyms for basic English words is the main cause to this problem. In this regard, we are planning to incorporate dynamic linguistic resources such as *Wiktionary* to complement the encyclopaedic nature of Wikipedia. Other established, static resources such as WordNet can also be used as a source of basic English vocabulary. Moreover, the incorporation of such complementary resources can assist in retaining and capturing additional semantics of complex terms by improving the mapping of constituents such as *"dried"* and *"sliced"*. General words which act as modifiers in composite terms often do not have corresponding topics in Wikipedia, and are usually unable to satisfy the relatedness requirement outlined in Section 3.1. Such constituents are currently ignored as shown through the high number of discarded constituents in level 2 in Figure 6. Moreover, the clustering of terms to discover new associations is only performed at level 1, and non-existent terms at level 2 and beyond are currently discarded.

Upon obtaining the subgraphs W_T for the two input sets, the system finds the corresponding nearest common ancestors. The NCA for the genetics-related and the foods-related terms is *"genetics"* and *"foods"*, respectively. Using these NCAs, our system constructed the corresponding lightweight domain ontologies as shown in Figure 7. The lightweight domain ontology for the *"foods"* domain is not included due to space constraint. However, a detailed account of this experiment is available to the public[1].

[1] http://explorer.csse.uwa.edu.au/research/_sandbox_evaluation.pl

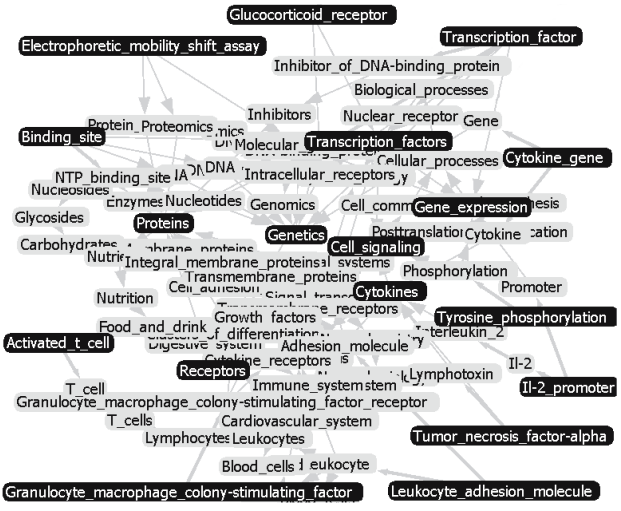


Fig. 7. The lightweight domain ontology generated using the subgraph W_T for the *"genetics"* domain. The important vertices (i.e. NCA, input terms, vertices with degree more than 3) have darker shades. The concept *"genetics"* in the center of the graph is the NCA. All 11 input terms are located along the side of the graph.

5 Conclusion and Future Work

Acquiring semantic relations is an important part of ontology learning. Many existing techniques face difficulty in extending to different domains, disregard implicit and indirect relations, and unable to handle relations between composite, ambiguous and non-existent terms. We presented a hybrid approach which combines lexical simplification, word disambiguation and association inference for acquiring semantic relations between potentially composite and ambiguous terms using only dynamic resources on the Web (i.e. Wikipedia and web search engines). During our initial experiment, the approach demonstrated the ability to handle terms from different domains, to accurately acquire relations between composite and ambiguous terms, and to infer relations for terms which do not exist in Wikipedia. The lightweight ontologies discovered using this approach is a valuable resource to complement other techniques for constructing full-fledged ontologies. Our future work includes the diversification of domain and linguistic knowledge by incorporating online dictionaries to support general words not available on Wikipedia. Evaluation using larger data sets, and the study on the effect of clustering words beyond level 1 is also required.

Acknowledgement

This research is supported by the Australian Endeavour International Postgraduate Research Scholarship, the DEST (Australia-China) Grant, and the Inter-university Grant from the Department of Chemical Engineering, Curtin University of Technology.

References

1. Sanchez, D., Moreno, A.: Learning non-taxonomic relationships from web documents for domain ontology construction. Data & Knowledge Engineering 64(3), 600–623 (2008)
2. Schutz, A., Buitelaar, P.: Relext: A tool for relation extraction from text in ontology extension. In: Proceedings of the 4th International Semantic Web Conference (ISWC), Ireland (2005)
3. Rozenfeld, B., Feldman, R.: Clustering for unsupervised relation identification. In: Proceedings of the 16th ACM Conference on Information and Knowledge Management (2007)
4. Sumida, A., Yoshinaga, N., Torisawa, K.: Boosting precision and recall of hyponymy relation acquisition from hierarchical layouts in wikipedia. In: Proceedings of the 6th International Language Resources and Evaluation (LREC), Marrakech, Morocco (2008)
5. Sabou, M., d'Aquin, M., Motta, E.: Scarlet: Semantic relation discovery by harvesting online ontologies. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 854–858. Springer, Heidelberg (2008)
6. Poesio, M., Almuhareb, A.: Identifying concept attributes using a classifier. In: Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition, Ann Arbor, USA (2005)
7. Shinyama, Y., Sekine, S.: Preemptive information extraction using unrestricted relation discovery. In: Proceedings of the NAACL Conference on Human Language Technology (HLT), New York (2006)
8. Jiang, T., Tan, A., Wang, K.: Mining generalized associations of semantic relations from textual web content. IEEE Transactions on Knowledge and Data Engineering 19(2), 164–179 (2007)
9. Pei, M., Nakayama, K., Hara, T., Nishio, S.: Constructing a global ontology by concept mapping using wikipedia thesaurus. In: Proceedings of the 22nd International Conference on Advanced Information Networking and Applications, Okinawa, Japan (2008)
10. Wong, W., Liu, W., Bennamoun, M.: Determination of unithood and termhood for term recognition. In: Song, M., Wu, Y. (eds.) Handbook of Research on Text and Web Mining Technologies, IGI Global (2008)
11. Wong, W., Liu, W., Bennamoun, M.: Tree-traversing ant algorithm for term clustering based on featureless similarities. Data Mining and Knowledge Discovery 15(3), 349–381 (2007)
12. Cilibrasi, R., Vitanyi, P.: The google similarity distance. IEEE Transactions on Knowledge and Data Engineering 19(3), 370–383 (2007)

Detecting Abnormal Events via Hierarchical Dirichlet Processes

Xian-Xing Zhang[1], Hua Liu[2], Yang Gao[1], and Derek Hao Hu[3]

[1] State Key Laboratory for Novel Software Technology, Nanjing University
flyaway2009@gmail.com, gaoy@nju.edu.cn
[2] Software Engineering School, Xi'an Jiaotong University
doudoulh@gmail.com
[3] Department of Computer Science and Engineering, Hong Kong University of Science and Technology
derekhh@cse.ust.hk

Abstract. Detecting abnormal event from video sequences is an important problem in computer vision and pattern recognition and a large number of algorithms have been devised to tackle this problem. Previous state-based approaches all suffer from the problem of deciding the appropriate number of states and it is often difficult to do so except using a trial-and-error approach, which may be infeasible in real-world applications. Yet in this paper, we have proposed a more accurate and flexible algorithm for abnormal event detection from video sequences. Our three-phase approach first builds a set of weak classifiers using Hierarchical Dirichlet Process Hidden Markov Model (HDP-HMM), and then proposes an ensemble learning algorithm to filter out abnormal events. In the final phase, we will derive abnormal activity models from the normal activity model to reduce the FP (False Positive) rate in an unsupervised manner. The main advantage of our algorithm over previous ones is to naturally capture the underlying feature in abnormal event detection via HDP-HMM. Experimental results on a real-world video sequence dataset have shown the effectiveness of our algorithm.

1 Introduction

Abnormal event detection is one of the most important problems in analyzing video sequences. Consider the situation that you are facing with a long video sequence, possibly thousands of hours in duration, but all you want is to find critical events among those video sequences. Anomaly detection, in the context of activity recognition, is the process whereby a baseline of normal behavior is established, with deviation from this norm triggering an alert.

Detecting unusual activities is difficult since unusual events by definition rarely occur, they may be unexpected as Zhang et al.[3] suggests, but at the same time they are relevant for the task. This difficulty becomes more significant during training due to the fact that we are in short of such labeled sequences of unusual activities.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 278–289, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Traditional framework for abnormal event detection is mainly addressed by state-space models and differential equation models, such as works in [2][6][7][29].

To address the complexity of time-series data, parametric approaches such as state-space models have to address the problem of model selection. In this context, a methodology for defining an appropriate set of states constitutes a key issue, among those available methods, state decomposition has typically been performed in an *adhoc* manner [30], this method essentially is a trial-and-error process to choose the number of states. In some particular settings, such as tracking-based abnormal event detection problem, we always can calculate many latent useful features after the tracking procedure, and always wish to select a number of critical features from those features [27], if we choose the Bayesian classifier, we have to decide the number of states for every state-space models[14]; this is impossible if the total number of features are large. Thus, in a word, previous state-space models all suffer the problem of deciding an appropriate number of features, and a non-optimal number of features decided would affect the overall performance a lot.

Here we apply the Hierarchical Dirichlet Process Hidden Markov model (HDP-HMM) [17], which by construction has an infinite number of states and can decide the suitable, or optimal, number of states automatically. Thanks to the effectiveness of HDP-HMM, we can propose a new feature selection algorithm which calculates the confidence vector by an ensemble of HDP-HMM-based weak classifiers. This ensemble of weak classifiers forms a general model, whose threshold is trained mainly to reduce the false positive rate. After we detected abnormal events from this general model, we filter out the misclassified normal activities later on, in order to reduce the false negative rate. Our experimental results show the effectiveness of this framework.

In the remaining of this paper, we will first review relevant background in abnormal event detection in Section 2. In Section 3 we describe our proposed three-phrase framework to tackle the problem of abnormal event detection in detail. In Section 4 we present the experimental results to show the effectiveness of our algorithm. Finally we conclude the paper and provide possible directions for future work.

2 Related Work

There are many important previous work done in the problem of abnormal activity recognition. However, due to a lack of space, we can only briefly review some related works using state-based models. Another important point in tackling in this problem is the feature selection step, which we will also review some important works.

Most of the previous work proposed in the topic of event detection [2][5][7][14] were centered on the recognition of predefined events in particular situations[13]. Now we consider the problem of abnormal event detection, and according to [7], the approaches for abnormal event detection can be divided into: similarity-based approaches and state model-based approaches. In model-based approaches, HMMs[2][6][7], and other graphical models[5] are most widely adopted.

In this paper, we only care about the trajectories of the objects, which is the results of object movement tracking [12]. Based on the tracking result we can, almost always, get a better result than by just using the original image sequences, because more useful and meaningful features can be computed from these trajectories, for example, semantic primitives can be included [29]. To address the problem of feature selection, which by definition is calculated from the object moving trajectories, [6] proposed a hybrid discriminative / generative approach, where the useful features were extracted by a variation of the AdaBoost algorithm proposed in [27]. At the same time [14] suggested several useful methods for feature selection, including Relief algorithm[15][16].

3 Our Proposed Algorithm

In this section, we present our three-phase framework for abnormal event detection from video surveillance. Intuitively, tracking-based abnormal detection techniques typically compare a profile of all normal activity sequences to objects' moving coordinates, we should calculate candidate features from those coordinates first, then use feature selection techniques to choose critical features to train the model. Any deviation from the model is flagged as a potential alarm.

Here we provide an overview of our proposed framework. First we will use every single calculated features to train an HDP-HMM based classifier using Beam Sampler in a sequential manner, by deciding a suitable model for every feature automatically.

In the second phase, we apply a new AdaBoost-like algorithm to train an ensemble of these weak classifier and get a confidence vector for later classification use, we propose a new algorithm because classical algorithms such as Relief[15] and its variants[16], and ensemble learning framework such as AdaBoost[27] cannot be applied directly to our setting, as explained later in this section. These weak classifiers can identify normal activities with a higher likelihood and assume that everything else is abnormal with a lower likelihood, and when choosing the threshold for the general model, we always intend to reduce the false positive rate, but this somehow involved with increasing the false negative rate. Therefore, abnormal event detection has the potential to detect even the unknown events that have rarely happened[7].

Based on this intuition, our proposed algorithm goes to a third phase process. We adapt the already well-estimated normal event model to a particular unusual event model using detected outlier if low likelihood observed[3], i.e, we start from the usual event model, and move towards an unusual event model in some constrained way, this process is proved to be able to effectively reduce false negative rate. In Section 3.1, we quickly review the Hierarchical Dirichlet Process and Gibbs Sampling methods, and introduce the sequential beam sampling methods. In Section 3.2, we briefly introduce how we calculate features from input coordinate data and how we build an ensemble of weak classifiers. Finally in Section 3.3, we describe how we build suitable model adaptation techniques.

3.1 HDP-HMM and Beam Sampling Method

Hierarchical Dirichlet Process Hidden Markov Model. Consider groups of data, denoted as $\{\{y_{ij}\}_{i=1}^{n_j}\}_{j=1}^{J}$, where n_j denotes the number of data in group j, J denotes the total number of groups thought to be produced by related, yet unique, generative processes. Each group of data is modeled via a mixture model[17]. A Dirichlet Process(DP) representation may be used separately for each of the data group. In an HDP manner, the base distribution of each of the DPs are drawn from a DP, which is discrete with probability 1, so each of the DPs can share the statistical strength, for instance, this encourages appropriate sharing of information between the data sets. An HDP formulation can decide the right number of states for the hidden Markov model(HMM); from its posterior density function on the appropriate number of mixture components, to some extent, the number of states in HMM can go to infinite if necessary. Besides, it learns the appropriate degree of sharing of data across data sets through the sharing of mixture components.

The HDP could be built as follows (Due to space constraint, we will omit the detailed explanation of HDP in this paper, interested readers could refer to [17] for details):

$$G_0(\theta) = \sum_{k=1}^{\infty} \beta_k \delta(\theta - \theta_k) \qquad \beta \sim GEM(\gamma) \qquad \theta_k \sim H(\lambda) \qquad k = 1, 2, \ldots \quad (1)$$

$$G_j(\theta) = \sum_{t=1}^{\infty} \bar{\pi}_{jt} \delta(\theta - \tilde{\theta}_{jt}) \qquad \tilde{\pi}_j \sim GEM(\alpha) \quad j = 1, \ldots, J \tilde{\theta}_{jt} \sim G_0 \quad t = 1, 2, \ldots \quad (2)$$

$$\bar{\theta}_{ji} \sim G_j \qquad y_{ji} \sim F(\bar{\theta}_{ji}) \qquad j = 1, \ldots, J, i = 1, \ldots, N_j. \quad (3)$$

Where $GEM(\cdot)$ stands for the stick-breaking process [17] as follows:

$$\beta'_k \sim Beta(1, \gamma)$$

$$\beta_k = \beta'_k \prod_{l=1}^{k-1} (1 - \beta'_l), \qquad k = 1, 2, \ldots \quad (4)$$

To better illustrate the construction of HDP-HMM, we introduce another equivalent representation of the generative model using indicator random variables

$$\beta \sim GEM(\gamma) \quad \pi_j \sim DP(\alpha, \beta) \quad z_{ji} \sim Mult(\pi_j) \quad \theta_k \sim H(\lambda) \quad y_{ji} \sim F(\theta z_{ji})$$

Identifying each $G(k)$ as describing both the transition probabilities $\pi_{kk'}$ from state k to k' and the emission distributions parametrized by $\phi_{k'}$, we can now formally define the HDP-HMM as follows:

$$\beta \sim GEM(\gamma), \qquad \pi_k \sim DP(\alpha, \beta), \qquad \phi_k \sim H, \quad (5)$$

$$s_t \sim Mult(\pi_{s_{t-1}}), \qquad y_t \sim F(\phi_{s_t}) \quad (6)$$

The Beam Sampler. The beam sampler[18] is a successful combination of slice sampling and dynamic programming, it does not marginalize out neither π nor ϕ. Specifically, the beam sampler iteratively samples the auxiliary variables $\boldsymbol{u}$, the trajectory $\boldsymbol{s}$, the transition probabilities $\boldsymbol{\pi}$, the shared DP parameters $\boldsymbol{\beta}$ and the hyperparameters conditioned on all other variables. We describe the Beam Sampler as follows:

Sampling $\boldsymbol{u}$: $u_t \sim \text{Uniform}(0, \pi_{s_{t-1}s_t})$

Sampling $\boldsymbol{s}$: Due to the fact that only the trajectories $\boldsymbol{s}$ with $\pi_{s_{t-1}s_t} \geq u_t$ for all t will have non-zero probability given $\boldsymbol{u}$, and there are only finite number of such trajectories due to the nature of stick-breaking process, as a consequence, we can apply dynamic programming to do the sampling job as classical HMM.[18]

$$p(s_t \mid y_{1:t}, u_{1:t}) \propto p(s_t, u_t, y_t \mid y_{1:t-1}, u_{1:t-1}) \tag{7}$$

$$= \sum_{s_{t-1}} p(y_t \mid s_t) p(u_t \mid s_t, s_{t-1}) p(s_t \mid s_{t-1}) p(s_{t-1} \mid y_{1:t-1}, u_{1:t-1}) \tag{8}$$

$$= p(y_t \mid s_t) \sum_{s_{t-1}: u_t < \pi_{s_{t-1}, s_t}} ps_{t-1} \mid y_{1:t-1}, u_{1:t-1}. \tag{9}$$

From (9), we can apply the classic dynamic programming[25] to sample the whole trajectory $\boldsymbol{s}$.

Sampling $\boldsymbol{\beta}$, m_{jk} is as follows[17]:

$$p((\beta_1, \ldots, \beta_K, \beta_{\bar{k}}) \boldsymbol{t}, \boldsymbol{k}, y_{1:T}, \gamma) \propto Dir(m_{\cdot 1}, ..., m_{\cdot K}, \gamma). \tag{10}$$

Sampling m_{jk}:

$$p(m_{jk} = m \mid n_{jk}, \beta, \alpha) = \frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k + n_{jk})} s(n_{jk}, m)(\alpha\beta_k)^m \tag{11}$$

where $s(n, m)$ are unsigned Stirling numbers of the first kind.

The conditional distribution of the observation y_t given an assignment $s_t = k$ and given all other observations y_τ, having marginalized out θ_k, is derived as follows:

$$p(y_t \mid y_{\setminus t}, s_t = k, s_{\setminus t}, \lambda) \propto \int_{\theta_k} p(y_t \mid \theta_k) p(\theta_k \mid \{y_\tau \mid s_\tau = k, \tau \neq t\}, \lambda) d\theta_k . f \tag{12}$$

Towards each ϕ_k we notice that[17] they are independent of others conditional on $\boldsymbol{s}$, $\boldsymbol{y}$ and their prior distribution H. One of the greatest merits of Beam Sampler is that even if the base distribution H is not conjugate to the data distribution F, an effective and simple Metropolis-Hastings could be developed[18].

3.2 Feature Calculation and Selection

In this section, we mainly introduce the feature calculation and selection method we adopted in our abnormal event detection framework.

Feature Calculation. Our abnormal event detection framework is based on the results of object tracking, so the first step of our method is to calculate useful features from the input coordinate sequences. There are many methods to calculate useful features, we adopt the method proposed in [14]. In [14], features are derived from the target global position provided by a tracker, and they can be divided into three groups:(1) instantaneous measurements, (2) average speed/velocity based features and (3) second order moments/energy related indicators.

Feature Selection. A variation of AdaBoost algorithm for classifier learning is provided by [27] for feature selection, AdaBoost is widely adopted in practice[6], but it's mainly designed for the discriminant classifiers. Therefore it's not suitable for our generative HDP-HMM based approach. At the same time, Relief algorithm [15] and its variations [16] are widely accepted in feature selection application, However, due to the fact that the majority of our training data always come from only one class, so neither can we apply Relief algorithm directly. For our HDP-HMM based generative model, we proposed a new AdaBoost-like algorithm as below, which combines merits from those two famous algorithms listed above, namely, our algorithm favors the feature that distinguishes the data best, and this algorithm belongs to an ensemble learning framework. Here we describe our algorithm in the following steps:

1. Given examples of feature sequences $(x_1, y_1) \ldots (x_n, y_n)$ where x_i is a matrix of features corresponding to a activity sequence, and $y_i = \{0, 1\}$ for normal and abnormal examples respectively.
2. Initialize weights $u_{i,j} = \frac{1}{2m}, \frac{1}{2l}$, votes $v_{i,j} = 1.1$, and initialize the total number of votes V_i to m, n for every i that $y_i = 0, 1$, corresponds to the normal and abnormal sequences, here m and l are the number of negative and positive sequences respectively, and j stands for one of the J features, where J is the total number of features.
3. Normalize the votes into weights: $u_{i,j} \leftarrow \frac{v_{i,j}}{V_i}$
4. for t=1:T, T is the number of iterations
   - for j=1:J, J is the total number of features
     - Compute the likelihood $L_{i,j}$ for every normal sequence i by the weak classifier trained by j feature, then multiply the sum of likelihood by its corresponding weight $u_{1,j}$, $L'_j \leftarrow u_(1, j) * \sum_{i=1}^{m} L_j$
   - Transform the weights for every feature back into votes by multiply total positive votes V_1. $v_{1,j} \leftarrow u_{1,j} * V_1$
   - Choose feature α with the greatest likelihood, increase the $v_{1,\alpha}$ by *voteinc*, and choose the feature β with the lowest likelihood, decrease $v_{1,\beta}$ by *votedec* if $v_{1,\beta} > 1$, else set it to 0.
5. Do the same procedure for the abnormal sequences
6. then we get two vectors for both the abnormal and normal training sequences(W1 and W2), now we can combine the two vectors into a single weight vector by a linear combination $C = \alpha * W1 + (1 - \alpha) * W2$. We can change the parameter α in order to get a lower false negative rate or a higher false positive rate, here we call the matrix as confidence vector.

7. The final classifier is: $Result(i) = \sum_{j=1}^{J} H(j) * Likelihood(J)$.
8. Sequence(i) is identified as abnormal if $Result(i) < Threshold$.

In our framework, we always choose a threshold with lower false positive rate, to make sure most of the outliers / abnormal activities can be detected and sent to the third phase.

3.3 Model Adaptation

The abnormal event models are derived from a general normal model in an unsupervised manner, and the benefit of such an unsupervised manner is that this framework can address the unbalanced label problem due to the scarcity of training data and the difficulty in pre-defining abnormal events, more specifically, after the second step we may get a high false negative rate, so it's necessary for us to apply a third phase, that is, to adapt models for the abnormal events, and use these abnormal classifiers to reduce the false negative rate. Besides, due to the lack of negative training data, we can not directly build models for abnormal events. However, we can use adaptation techniques to get them during the test time or even in the future use, that is, we can dynamically build the model for the abnormal event after the training phrase. Here we briefly introduce the algorithm's framework[3][7] first. The steps are listed as below:

Prerequisites: A well defined general HDP-HMM with Gaussian observation density trained by all normal training sequences.

Step 0 : Using the first outlier detected from the former phase - which is considered to be able to represent a particular type of abnormal activities - to train an abnormal event model by adaptation using beam sampler.
Step 1 : Slice the test sequence into fixed length segments, calculate the likelihood of these segments by the existing models, if the maximum likelihood is given by the general model, we predict this trace to be of a normal activity, then goto Step 4. Else goto Step 2;
Step 2 : If the maximum likelihood is larger than the threshold, we consider this trace to belong to an existing abnormal model; then we predict this trace to be that kind of abnormal events, goto Step 4, else goto Step 3;
Step 3 : Using adaptation method to adapt the general model to a new abnormal activity model, then add this adapted abnormal model to the set of models, go to Step 4, here this outlier is regarded to represent one kind of the certain events.
Step 4 : Goto Step 11 if new outlier comes.

In this procedure, we provide the outlier a second chance to be recognized as a normal events, in this way, we try to identify those normal events that are misclassified due to their unexpectedness or scarcity in training data. Thanks to the effectiveness of beam sampler again, we can do the adaptation effectively without other special design. Suppose now we have the new parameters for the HDP-HMM λ, here the parameters that are adapted are the HDP parameter $\beta, \alpha_0, \gamma, K$ and the π, μ for the HMM.

Here a latent problem for the adaptation is that, we can no longer use traditional adaptation algorithms, such as Maximum Likelihood Linear Regression(MLLR) and Maximum a posterior(MAP), or Kernel Nonlinear Regression(KNLR) because the number of states K will change due to the adaptation, leading to the different dimensions between old and new variables, we adopted a heuristic method to deal with the problem, as will be discussed in the experimental part, and we will discuss it later in the final section.

4 Experiments and Results

In this section, we first describe our dataset and performance measures, then we describe the baseline systems we use to evaluate our results. At last we illustrate the effectiveness of the proposed algorithm framework using the dataset.

4.1 Dataset Description

We have concentrated on low-level, short term events, especially human activities, as those studied in the context of the CAVIAR project [28]. We will explore the availability of a large dataset of video sequences and manually classified activities, in a total of about 16,000 images with ground truth data. The activities (classes) considered can be detected from a relatively short video sequence (a few seconds) and are described in Table 1, we train the general model from a portion of the set of normal sequences, and test the model for the rest of normal sequences and the other sets of abnormal sequences, all of the data is available at [28].

Table 1. Examples of Normal and Abnormal Events

Normal events	Abnormal events
Walker: Walking	...
Browser: Browsing	Fighter: Running
Fighter: Fighting	Fighter: Walking
...	Walker: Running
...	...

4.2 Performance Measures

One of the most challenging problems involved with abnormal detection is the different misclassification costs corresponding to different types of error[19]. According to [7], cost-sensitive learning are useful for learning problems where the classes are extremely unbalanced. And in most of the time it's more important to recognize the negative class. We usually apply cost-sensitive learning under such situation, for example, Receiver Operating Characteristic (ROC) curves are recommended instead of accuracy [20]. In [21], the authors show that class-imbalance often affects the performance of cost-sensitive classifiers: when the

misclassification costs are not seriously unequal, cost-sensitive classifiers generally favor natural class distribution although it might be imbalanced; when misclassification costs are seriously unequal, a balanced class distribution is more favorable.

As we mentioned above, the problem of abnormal event detection can be recognized as a two-class classification problem, where the positive class(normal events) takes dominating majority in the whole training data, and so the traditional accuracy-based performance measure is no longer suitable for this setting. Here we evaluate the performance of the abnormal event detection algorithm framework in terms of two error rates: detection rate and false alarm rate. The detection rate is the same as true-positive rate while the false alarm rate is the same as false positive rate.

4.3 Results on CAVIAR Sequences

In order to evaluate the performance of our proposed algorithm, experiments were carried out on a real data set from [28]. For comparison, two other approaches were used as baselines. The first one is composed by HDP-HMM + AL, the second one is composed by HDP-HMM LITE + Adaptation, and our proposed algorithm is referred to as HDP-HMM + AL + Adaptation, here AL is referred to our proposed AdaBoost-like feature selection algorithm, and LITE is referred the feature selection algorithm proposed in[14], which is proven to be effective to the problem of activity recognition also based on data set[28].

We randomly select a 50% subset of normal event sequences(50 sequences) for training, these sequences are of $13KB$ size in average. The other 50 normal traces and all the 20 abnormal traces were randomly mixed together for testing. Fig. 1 shows the ROC curve with respect to the detection rate and the false alarm rate.

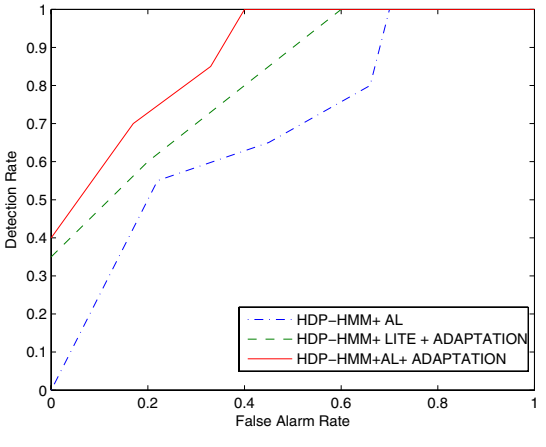


Fig. 1. Comparison of the detection rate and the false alarm rate

We can see from Figure 1 that, HDP-HMM + AL gives the poorest detection result because it achieves a high detection rate at the cost of incurring a high false alarm rate. From Figure 1, it is also noteworthy to mention that our AL method for feature selection performs much better than the LITE algorithm, which will achieve a perfect detection rate of 1 at the cost of 0.4 false alarm rate, while LITE algorithm requires a 0.6 False Alarm Rate when the detection rate is 1.

Furthermore, after we adopted the Adaptation technique, both HDP-HMM + LITE + Adaptation and HDP-HMM + AL + Adaptation received better results. This is because the adapted abnormal activity models contribute to distinguishing normal and abnormal activities, which can significantly reduce the false alarm rate, in this sense the formerly misclassified normal events get a second chance to be differentiated from the abnormal events. The essential advantage of AL, compared to LITE is that it not only gains the information of how the feature contributes to classification.

5 Conclusions

In this paper, we propose a novel approach for abnormal event detection. To deal with the scarcity of training data for abnormal events, we proposed a three-phrase abnormality detection algorithm. In the first phrase, we train a set of weak classifiers based on HDP-HMM, in particular, we have chosen beam sampling for inference, which combines slice sampling with dynamic programming. In the second phase, we developed an ensemble learning algorithm to filter out abnormal events, and the suspicious traces are then passed on to a collection of abnormal events models adapted from the general model for further detection. The major advantage of our approach is that HDP-HMM can naturally capture the underlying features and the hierarchical detection phase can achieve a better tradeoff between detection rate and false alarm rate. We demonstrate the effectiveness of our approach using real data collected by CAVIAR PROJECT.

One limitation of our approach is the adaptation technique we adopted in this framework. We have chosen HDP-HMM because it does not select a fixed number of states, and a full posterior density function is inferred for all model parameters, this introduces one problem: since the number of parameters may change after the adaptation, as a result, we can not employ classic adaptation techniques such as MLLR[6]. Our solution is just based on the sequential nature of beam sampler but not consider the mismatch between an initial model and the adaptation data. Therefore we need to be able to develop an adaptation technique to address this problem. Another latent problem involved with the adaptation step is that there is a risk of generating a large number of abnormal models when abnormal activities suddenly become the norm, as suggested in [7], we need to detect when abnormal activities turns normal and vice versa.

Acknowledgement

We would like to acknowledge support for this project from the National Science Foundation of China(NSFC grant No.60775046 and No.60721002) and the

National Grand Fundamental Research 973 Program of China (grant No.2009CB320700).

References

1. Yao, Y., Wang, F., Wang, J., Zeng, D.D.: Rule+Exception Strategies for Security Information Analysis. IEEE Intelligent Systems 20(5), 52–57 (2005)
2. Wang, J., Xu, C., Chng, E.S., Tian, Q.: Sports highlight detection from keyword sequences using HMM. In: Proc. IEEE ICME, Taiwan (June 2004)
3. Zhang, D., Gatica-Perez, D., Bengio, S., McCowan, I.: Semi- Supervised Adapted HMMs for Unusual Event Detection. In: CVPR, pp. 611–618 (2005)
4. MacKay, D.J.C.: Ensemble learning for hidden Markov models. Tech. Rep., Cavendish Lab, Univ. Cambridge, Cambridge, U.K (1997)
5. Gong, S., Xiang, T.: Recognition of group activities using a dynamic probabilistic network. In: Proc. IEEE ICCV, Nice (October 2003)
6. Lester, J., Choudhury, T., Kern, N., Borriello, G., Hannaford, B.: A Hybrid Discriminative/Generative Approach for Modeling Human Activities. In: Proc. 19th Int'l Joint Conf. Articial Intelligence (IJCAI 2005), July-August 2005, pp. 766–772 (2005)
7. Yin, J., Yang, Q., Pan, J.J.: Sensor-based Abnormal Human-Activity Detection IEEE Transactions on Knowledge and Data Engineering (IEEE TKDE). IEEE Computer Society Digital Library. IEEE Computer Society, Los Alamitos (2007)
8. Pruteanu-Malinici, I., Carin, L.: Infinite Hidden Markov Models for Unusual-Event Detection in Video. IEEE Transactions On Image Processing 17(5) (May 2008)
9. Bradley, A.P.: Shift-invariance in the discrete wavelet transform. Digit. Image Comput. Tech. Appl., pp. 29–38 (December 2003)
10. Hurri, J., Hyvarinen, A., Karhunen, J., Oja, E.: Image feature extraction using independent component analysis. In: The IEEE Nordic Conf. Signal Processing (1996)
11. Kohonen, T.: Emergence of invariant-feature detectors in the adaptive- subspace self-organizing map. Biol. Cybern. 75(4), 281–291 (1996)
12. Stauffer, C., Grimson, W.E.L.: Learning patterns of activity using real-time tracking. IEEE Trans. Pattern Anal. Mach. Intell. 22(8), 747–757 (2000)
13. Viola, P., Jones, M., Snow, D.: Detecting pedestrians using patterns of motion and appearance. IJCV 63(2), 153–161 (2005)
14. Ribeiro, P.C., Santos-victor, J.: Human activity recognition from video: modeling, feature selection and classification architecture. In: International Workshop on Human Activity Recognition and Modeling (HAREM) (2005)
15. Kira, K., Rendell, L.: A practical approach to feature selection. In: Proc. 9th Int. Workshop on Machine Learning, pp. 249–256 (1992)
16. Robnik-Sikonja, M., Kononenko, I.: Theoretical and Empirical Analysis of ReliefF and RReliefF. Machine Learning Journal 53, 23–69 (2003)
17. Teh, Y.W., Jordan, M.I., Beal, M.J., Blei, D.M.: Hierarchical dirichlet processes. J. Amer. Statist. Assoc. 101, 1566–1581 (2006)
18. Gael, J.V., Saatci, Y., Teh, Y.W., Ghahramani, Z.: Beam Sampling for the Infinite Hidden Markov Model. In: ICML (2008)
19. Elkan, C.: The Foundations of Cost-Sensitive Learning. In: Proc. 17th Int'l Joint Conf. Articial Intelligence (IJCAI 2001), pp. 973–978 (August 2001)

20. Ling, C.X., Huang, J., Zhang, H.: AUC: A Statistically Consistent and More Discriminating Measure than Accuracy. In: Proc. 18th Int'l Joint Conf. Artificial Intelligence (IJCAI 2003), pp. 519–526 (August 2003)
21. Liu, X.-Y., Zhou, Z.-H.: The influence of class imbalance on cost-sensitive learning: An empirical study. In: Perner, P. (ed.) ICDM 2006. LNCS, vol. 4065, pp. 970–974. Springer, Heidelberg (2006)
22. Yilmaz, A.: Object Tracking by Asymmetric Kernel Mean Shift with Automatic Scale and Orientation Selection. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007) (2007)
23. MacKay, D.J.C.: Ensemble learning for hidden Markov models. Tech. Rep., Cavendish Lab, Univ. Cambridge, Cambridge, U.K (1997)
24. Hongeng, S., Nevatia, R., Bremond, F.: Video-based event recognition: activity representation and probabilistic recognition methods. Computer Vision and Image Understanding 96, 129–162 (2004)
25. Rabiner, L.R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proc. IEEE 77(2), 257–286 (1989)
26. Beal, M.J., Ghahramani, Z., Rasmussen, C.E.: The infinite hidden Markov model. In: Advances in Neural Information Processing Systems, pp. 577–584. MIT Press, Cambridge (2002)
27. Viola, P., Jones, M.: Rapid Object Detection using a Boosted Cascade of Simple Features. Computer Vision and Pattern Recognition (2001)
28. CAVIAR PROJECT, `http://homepages.inf.ed.ac.uk/rbf/CAVIAR/`
29. Chan, M.T., Hoogs, A., Schmiederer, J., Perterson, M.: Detecting rare events in video using semantic primitives with HMM. In: Proc. ICPR (August 2004)
30. Runkle, P., Bharadwaj, P., Carin, L.: Hidden Markov model for multi-aspect target classification. IEEE Trans. Signal Process. 47(7), 2035–2040 (1999)

Active Learning for Causal Bayesian Network Structure with Non-symmetrical Entropy

Guoliang Li and Tze-Yun Leong

School of Computing, National University of Singapore
Computing 1, Law Link, Singapore 117590
{ligl,leongty}@comp.nus.edu.sg

Abstract. Causal knowledge is crucial for facilitating comprehension, diagnosis, prediction, and control in automated reasoning. Active learning in causal Bayesian networks involves interventions by manipulating specific variables, and observing the patterns of change over other variables to derive causal knowledge. In this paper, we propose a new active learning approach that supports interventions with node selection. Our method admits a node selection criterion based on non-symmetrical entropy from the current data and a stop criterion based on structure entropy of the resulting networks. We examine the technical challenges and practical issues involved. Experimental results on a set of benchmark Bayesian networks are promising. The proposed method is potentially useful in many real-life applications where multiple instances are collected as a data set in each active learning step.

Keywords: Bayesian networks, active learning, intervention, non-symmetrical entropy, node selection, stop criterion.

1 Introduction

Causal knowledge is important for facilitating comprehension, diagnosis, prediction and control in automated reasoning. Causal Bayesian networks are extensions to Bayesian networks that explicitly and concisely represent causal knowledge as variables and their directed graphical relationships in uncertain domains [10]. This research focuses on learning causal knowledge from data that corresponds to learning the structure of causal Bayesian networks for knowledge discovery. A major research challenge is to learn causal knowledge from both observational and interventional data. Observational data are derived from passive observations when the underlying system evolves autonomously. Interventional data are observed when some variables are actively manipulated to specific values, while the other variables evolve autonomously according to the underlying system mechanisms; such data directly reflect the effects of the manipulated variables on the other variables of the system.

Most of the existing Bayesian network structure learning methods deal with observational data [5]. Recently, some new methods have been proposed to combine observational data with interventional data [2, 14]. Cooper and Yoo [2] identified the possible assumptions for probability updates with both observational and interventional

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 290–301, 2009.
© Springer-Verlag Berlin Heidelberg 2009

data and extended the Bayesian method in Cooper and Herskovits [1] for probability update with a closed form. Tong and Koller [14] applied active learning strategy to Bayesian networks and collected new interventional data for further structure probability updates. Eberhardt et al. [4] proved that, under ideal conditions with causal Markov assumption and faithfulness assumption (and ideal distributions), the number of interventions required to identify the causal relationships among N variables is $N-1$ when only one variable can be manipulated each time, and the number of interventions is $\log_2 N$ when multiple variables can be manipulated simultaneously.

Active learning in Bayesian networks involves interventions by manipulating specific variables, and observing the patterns of change over the other variables to derive causal knowledge. In previous work [14], the interventional data are assumed to be one instance at each active learning step. In this work, we consider a new scenario: *multiple instances are collected when one variable is manipulated at each active learning step*. Such experiments arise in many real-life applications. For instance, in measuring protein expression levels with flow cytometry in biology, the expression levels of some proteins (as variables) can be manipulated to certain levels; their effects on the expression levels of the other proteins are observed from many cells at a time. Such observations of protein expression levels from one cell constitute the values in one instance [3, 11].

With an interventional data set, we can determine the causal influences of the manipulated variables on the other variables based on *the theory of causality with agency*: manipulating causes can change the effects but not vice versa. In practice, marginal distributions of the variables are used to detect causal influences. If the marginal distribution of variable B changes when variable A is manipulated to different values, we say that variable A *precedes* variable B in causal ordering.

There are different definitions of intervention: perfect intervention, imperfect intervention [8, 13], and uncertain intervention [3]. Different types of intervention have different effects on the Bayesian network structures learned from data. We will focus on perfect intervention in this work. When we manipulate a variable under perfect intervention, the manipulated variable takes the value we specify in the intervention. This is the meaning of manipulation in the general sense.

Our objective is to learn the causal Bayesian network structure that achieves the specified structure accuracy with a minimal number of interventions, when the interventional data comprise a data set at each active learning step: 1) What is a good criterion for selecting the nodes for new interventions, with respect to "correctness" in terms of entropy of the learned structure? 2) What is the effect of the stop criterion on the learned structure in the learning process?

We introduce a new active learning algorithm for causal Bayesian networks with a non-symmetrical-entropy-based node selection criterion and an entropy-based stop criterion. The non-symmetrical entropy is motivated by the non-symmetrical nature of the interventions. We examine the effectiveness and efficiency of the proposed method on identifying causal relationships based on three benchmark Bayesian networks and two Bayesian networks we created; we compare our method with some other major methods including node selection with symmetrical entropy, random node selection, and observational data only, and we observe that the results are promising.

2 Method

2.1 Causal Bayesian Networks

A causal Bayesian network [10] is a directed acyclic graph (DAG), in which each node corresponds to a distinct variable X_i in the domain, and each edge corresponds to a causal influence from the parent variable to the child variable. The parent variable of an edge is the variable at the tail of the edge, and the child variable is the variable at the head of the edge. The meaning of "causality" in causal Bayesian networks is as follows: when we manipulate the parent variable by fixing its state to different values, we can observe the change in the probability distribution of the child variable. If there is no causal influence from variable A to variable B, there will be no edge from variable A to variable B in the causal Bayesian network. Moreover, when one variable is manipulated, the causal influence relationship between other variables will not change, and the conditional probability of the child variable given its parents will be the same. Under the causal Markov assumption, each variable is independent of its ancestors given the values of its parents. The joint probabilities in the domain can be represented as

$$p(X_1,...,X_n) = \prod_i p(X_i \mid Pa(X_i))$$

where $Pa(X_i)$ denotes the parents of X_i in the causal Bayesian network. In this paper, we will use "node" and "variable" interchangeably. A good definition of causal Bayesian networks and its properties can be found in Pearl's book [10].

2.2 Active Learning

Generally, there are two categories of approaches to learn Bayesian network structures from data: score-and-search-based approaches [1, 6] and constraint-based approaches [12]. These methods are considered to be passive learning and the data set does not change in the learning process.

Active learning is different from passive learning. In active learning, new data will be collected in the learning process with the guidance of the available data to maximally reduce the variance in the model. In previous work [14], active learning starts with an available data set, and node selection for intervention is based on the expected posterior loss of the structure entropy. The edge probabilities need to be estimated for expected posterior loss calculation under all the possible interventions and the possible outcomes of each intervention. The intervention with the maximal expected posterior loss will be selected and one new interventional instance will be collected. The new collected instance will be combined with the available data for edge probability estimation. The process can be repeated until the goal is reached.

Estimating the edge probabilities is an important part of the active learning process. For every pair of variables, three possible situations between them are usually considered: an edge from A to B ($A \rightarrow B$), an edge from B to A ($A \leftarrow B$), or no edge between A and B ($A \perp B$). The probabilities of the edges given the available data D and domain knowledge K are defined as

$$\Pr(A \to B \mid D, K) = \sum_{A \to B \in E(G)} \Pr(G \mid D, K)$$

where $\Pr(G \mid D, K)$ is the probability of Bayesian network G given the data D and domain knowledge K, and $E(G)$ is the set of edges in Bayesian network G. In the following discussions, D and K will be omitted for brevity. The probabilities of $A \leftarrow B$ and $A \perp B$ are similarly defined as the probability of $A \to B$. The edge entropy is defined as in Tong and Koller's paper [14]:

$$\begin{aligned} H(A,B) = & -p(A \to B)\log p(A \to B) \\ & - p(A \leftarrow B)\log p(A \leftarrow B) \\ & - p(A \perp B))\log p(A \perp B) \end{aligned}$$

The structure entropy of Bayesian network G is defined as

$$H(G) = \sum_{A,B} H(A,B)$$

In Tong and Koller's work [14], the edge probabilities are estimated approximately with Markov Chain Monte Carlo (MCMC). In contrast, we estimate the edge probabilities with an exact method proposed by Koivisto [7], since the exact edge probabilities can provide more information for node selection. Koivisto utilized the intuition that the order of the parents of a variable is irrelevant to the variable's probability estimation, and applied forward and backward dynamic programming and fast truncated Mobius transform to estimate all the edge probabilities in $O(n2^n)$ time, where n is the number of variables in the domain. For the interventional data, the instances with the variable manipulated will not be used in calculating the probability of the family with the manipulated variable as the child (Refer to Cooper and Yoo's work [2] for this method). Koivisto's exact method can be applied to domains with a moderate number of variables (around 25).

2.3 Selecting Nodes for New Interventions

We consider the situation where a data set will be collected when one variable is under one intervention. It is likely that an interventional data set will show whether the manipulated variable will affect the probability distributions of other variables. The method mentioned above could not be effectively applied to this situation due to computational complexity. Suppose that m instances are collected in each active learning step when one variable is manipulated in a domain with n binary variables, the computational complexity to estimate the expected posterior loss is $O(n^2 * 2^{mn+n-m})$, which is infeasible even for small m and n.

We propose to select the node with the maximum node uncertainty from the current data for intervention and will not consider the possible interventions and the possible outcomes from each intervention. This will reduce the computational cost significantly. The node uncertainty between a variable and all the other variables can be estimated under two different conditions:

$$U_{NS}(A) = \sum_{B} \left(-\Pr(A \to B) * \log(\Pr(A \to B)) - (1 - \Pr(A \to B)) * \log(1 - \Pr(A \to B)) \right) \quad (1)$$

$$U_{S}(A) = \sum_{B} H_{S}(A, B) \quad (2)$$

The first case U_{NS} considers two conditions between variable A and the other variables: the probabilities of whether there is an edge from A to other variables. The second case U_S considers the three possible conditions between variable A and the other variables: $A \to B$, $A \leftarrow B$, and $A \perp B$. The second case is generally used in Bayesian network structure leaning.

We refer to U_{NS} as *non-symmetrical entropy* and U_S as *symmetrical entropy*. The definition of the non-symmetrical entropy is motivated by the non-symmetrical nature of the intervention. In an intervention, we can only manipulate one variable in a pair of variables to derive the causal information between the pair: whether the manipulated variable affects the non-manipulated variable. We cannot derive causal information from the non-manipulated variable to the manipulated variable. If both variables are manipulated, we cannot derive useful causal information between this pair of variables from the interventional data.

Besides examining the effects on node selection with these two measures, we also consider random node selection for intervention and observational data.

2.4 Stop Criteria for Causal Structure Learning

Another main problem in applying Bayesian network learning for causal knowledge discovery in practice is to decide when to stop the learning process – when do we think that the learned causal Bayesian network is good enough? The intuitive way is to choose a fixed number of interventions as the stop criterion. The disadvantage of this approach is that there is no guarantee on the quality of the learned Bayesian network structure. We propose to use certain "acceptable" entropy of the learned structure as the stop criterion. The ideal entropy of the learned structure is 0; in practice, however, it is difficult to reach the ideal condition. We consider the effects of the different values of entropy of the learned structure as the stop criteria on the accuracy of the learned structures.

3 Experiments

The proposed method has been tested in experiments with the same benchmark Bayesian networks as those reported in Tong and Koller's work [14]: Cancer network (as shown in Figure 1), Asia network, and Car network, and two Bayesian networks created by ourselves: Study network and Cold network [9]. There are 2 variables in Study network, 3 variables in Cold network, 5 variables in Cancer network, 8 variables in Asia network and 12 variables in Car network, respectively. We conducted the simulations under MATLAB[1] (version 7) with the support of the BDAGL

[1] http://www.mathworks.com/products/matlab/

package[2]. The machine used is a Dell OptiPlex GX280 desktop with 1 Gigabyte memory and 3GigaHz Intel processor.

The experiment setup is as follows:

1) *Choose one Bayesian network from the five Bayesian networks as the ground truth Bayesian network;*
2) *Sample an observational data set with N_obs instances from the ground truth Bayesian network;*
3) *Estimate the edge probabilities and structure entropy with the available data (and domain knowledge, if any);*
4) *Check the stop criterion. If the stop criterion is satisfied, stop the learning process; otherwise, continue;*
5) *Select one node for intervention based on the node uncertainty measures from non-symmetrical entropy, symmetrical entropy, random node selection for intervention, or without interventional node;*
6) *Generate a set of new interventional data with N_int instances from the ground truth with the selected manipulated variable; combine the new data with the existing data as the new available data; return to step 3).*

In the experiments, the edge probabilities are estimated with the exact method from Koivisto [7]. The uniform prior of Bayesian network structures is used. The size of the observational data *N_obs* is set to 20, and the size of the interventional data *N_int* in each intervention changes from 1 to 200 instances. Such data size for each intervention is more realistic than an ideal distribution as discussed in Eberhardt et al. [4]. We tested two stop criteria in our experiments – the number of interventions and the structure entropy of the learned Bayesian networks. In the latter, the maximum number of interventions is set to 6 in the Cancer network when the size of the interventional data is 200. This is because we observed that the structure entropy of the learned Bayesian network would not reach certain small values when the manipulated variables are selected with symmetrical entropy, even if a very large data set is collected. Experiments showed that, the results from the different tested Bayesian networks and different sizes of the interventional data are similar. In the later sections, the results will be demonstrated with the Cancer network and the size of the interventional data as 200. More detailed results are available in Li [9].

In the experiments, when one variable is selected for intervention, the links pointing to this variable will be removed from the graph and this variable will be set to a fixed value. The values of other variables are sampled based on the Bayesian network structure and the original conditional probabilities. In addition, one variable can be selected for more than one round of intervention in the active learning process.

We used the original conditional probabilities in the Bayesian networks first. To test whether the conclusions depend on specific values of the conditional probabilities in the original Bayesian networks, we also conducted experiments with the same Bayesian network structures but with randomized conditional probabilities. The conclusions from the experiments with the randomized conditional probabilities are similar to the results with the original conditional probabilities, and are consistent over all the Bayesian networks tested.

[2] http://www.cs.ubc.ca/~murphyk/Software/BDAGL/

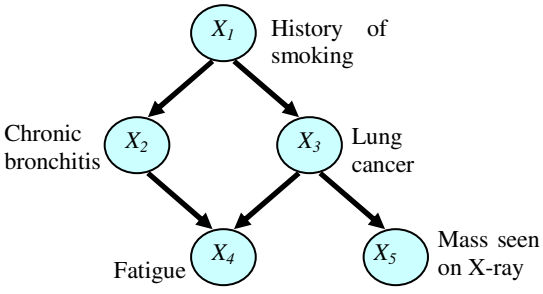


Fig. 1. Cancer Bayesian network

3.1 Number of Interventions vs. Structure Entropy

In the first experiment, we tested the relationship between the number of interventions and the entropy of the learned structures. The objective is to show how the entropy of the learned structures varies with the different node selection methods, when the number of the interventions is the same. The total instances to be collected are set to 2000 for Cancer network, 1000 for Study network and Cold network and 5000 for Asia network and Car network. The maximum number of interventions is set to 6 when the size of the interventional data in each active learning step is 200. This is because the structure entropy of the learned Bayesian networks with more than 6 interventions was observed to be very low. When the size of the interventional data in each active learning step is different, the maximum number of possible interventions would change. For the Cancer network, when the number of the total instances from all the active learning steps approaches 1000 to 1200, the entropy of the learned structure with non-symmetrical entropy would converge.

The programs ran 8 hours and finished 608 repeated experiments[3] on the Cancer network (about 48 seconds for one experiment), and the results are shown in Figure 2. In Figure 2, the lines represent the change of the average structure entropy with the number of interventions. Figure 2 shows that, with the same number of interventions, node selection with non-symmetrical entropy can derive a Bayesian network with the lowest entropy (also with the smallest variance on average), which means the structure of the learned Bayesian network is more certain.

The structure learned with observational data has the highest entropy. The entropy of Bayesian network structure learned with the random node selection and node selection with the symmetrical entropy fall between those of the node selection with non-symmetrical entropy and the observational data. This is consistent with our expectation, since the intervention is non-symmetrical in nature and the interventional data can provide more causal information about the probabilities between the manipulated variable and other variables. If there is a real edge from the manipulated variable to another variable, the probability of this edge should increase with the interventional

[3] We distinguish between the terms "intervention" and "experiment" here. "Intervention" means to manipulate the variables and observe their effects on other variables. "Experiment" means to run the method for testing.

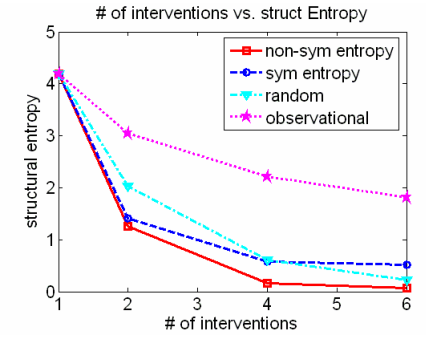


Fig. 2. Relationship between the number of interventions and the structure entropy of the learned Bayesian network from Cancer network. The non-sym entropy and the sym entropy refer to the node uncertainty measures with non-symmetrical entropy and symmetrical entropy defined in formulas (1) and (2), which are the same for other figures.

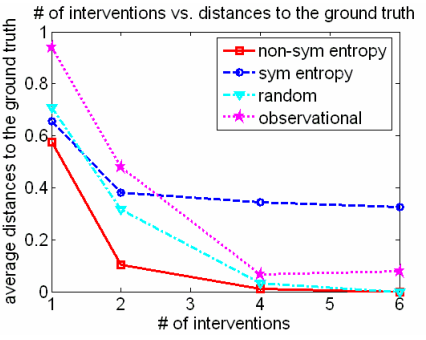


Fig. 3. Relationship between the number of interventions and the average Hamming distance from the learned Bayesian network structure to the ground truth from Cancer network

data, and the non-symmetrical entropy will decrease. However, the symmetrical entropy may not decrease since we do not know the probability change in other two conditions between these two variables.

The significance of the entropy difference from different node selection measures was evaluated by one-sided t-test. The p-values between the entropy of the learned Bayesian network structure from non-symmetrical entropy and other methods are all smaller than 10^{-10}. This means that the entropy from non-symmetrical entropy is significantly smaller than that from other methods.

From Figure 2, we have a surprising observation. When the number of interventions is smaller than 6 in the Cancer network, the entropy of the learned structure with nodes selected from the symmetrical entropy is lower than that from random node selection. When the number of interventions is equal to or greater than 6, the entropy of the learned structure by node selection with symmetrical entropy is higher than that from random node selection. It means that, in the first several interventions, symmetrical entropy selects the nodes to reduce the structure uncertainty significantly. However, when the number of interventions is greater than 6, the leaf nodes (nodes X_4 and X_5 in Figure 1) are always selected by symmetrical entropy. The data with leaf nodes manipulated can reduce the estimated probabilities of the edges from the leaf nodes to other nodes. But, the data cannot provide information about the influence relationships from other nodes to the leaf nodes. The uncertainty of the leaf nodes calculated from symmetrical entropy can still be quite large. However, the random method may select other nodes for intervention, which could generate subsequent interventional data with more causal information about the edges from other nodes to leaf nodes and leaf nodes to other nodes. Such information will reduce the total structure entropy.

Figure 2 also shows that, with more interventions (or more data), the entropy of the learned structure decreases with all the node selection criteria. The entropy of the learned Bayesian network structure generally decreases more in the first few

interventions. Then, the entropy of the learned structure seems to converge to certain values. These results are similar across all the Bayesian networks tested.

3.2 Number of Interventions vs. Distance to the Ground Truth

In this experiment, we compared the learned structure with the ground truth Bayesian networks. The difference between the learned structure and the ground truth is measured with Hamming distance. Figure 3 shows that node selection with non-symmetrical entropy leads to Bayesian networks with the smallest average Hamming distance to the ground truth, as compared with other methods for node selection. With 6 or more interventions, the average distance is 0 and the variance is 0 with the Cancer network when the nodes are selected based on non-symmetrical entropy. The variance of the Hamming distance from non-symmetrical entropy is the lowest, while the variances of the Hamming distances from the symmetrical entropy and observational data are quite high (about 0.55 and 0.33 respectively). In addition, Figure 3 shows the changes of the average Hamming distance with the number of interventions. With more interventional data, the average distance from the learned structure to the ground truth will be smaller.

From Figures 2 and 3, we can observe that, when the number of the interventions increases, the structure entropy converges to a certain low value with either node selection with non-symmetrical entropy or random node selection. The reason is that the true causal Bayesian network structure can be identified with sufficient interventional data from any node selection method. We note that, however, when the number of interventions is small, non-symmetrical entropy could outperform all other methods for node selection in active learning. The difference in performance could be significant in applications where the resources are scarce or only a small number of interventions are feasible.

3.3 Structure Entropy vs. Distance of the Learned Structure to the Ground Truth

In practice, we do not know the ground truth structure, and cannot use the Hamming distance from the learned structure to the ground truth structure as the stop criteria to

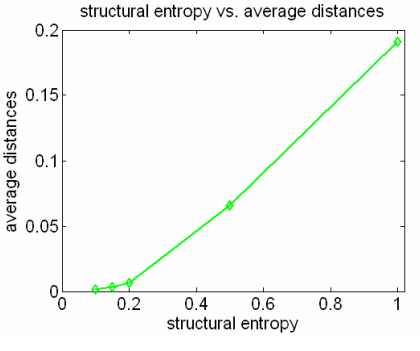


Fig. 4. Relationship between structure entropy of the learned Bayesian network and the Hamming distance to the ground truth

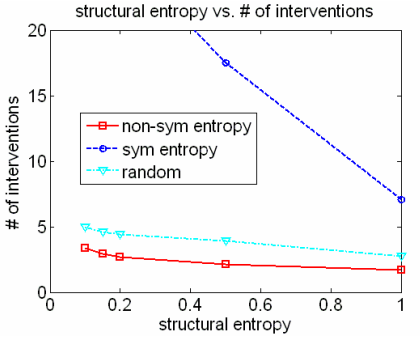


Fig. 5. Relationship of structure entropy and the number of interventions required for Cancer network

learn causal Bayesian networks. This experiment will examine the relationship between the structure entropy and the Hamming distance from the learned structure to the ground truth Bayesian network structure. Figure 4 shows how the entropy of the learned structure approximates the average Hamming distance from the learned structure to the ground truth. The relationship between the average entropy of the learned structure and the average distance from the learned structure to the ground truth is approximately linear, which means that the entropy of the learned structure is a good approximation of the distance of the learned structure to the ground truth Bayesian network and can be used as a stop criterion for the structure learning.

3.4 Structure Entropy as Stop Criterion

In the next experiment, we tested the effect of the structure entropy as the stop criterion. Figure 5 shows that, with non-symmetrical entropy as the node selection criterion, the program can reach the required structure entropy with a smaller number of interventions. When the interventional node is selected with symmetrical entropy, a large number of interventions are needed. The results from observational data do not show in Figure 5, as the program cannot reach the required structure entropy in the maximum steps allowed (50 steps) in that set of experiments.

3.5 Comparison with Expected-Posterior-Loss-Based Method

For comparison, we have implemented the method based on the expected posterior loss [14]. In our implementation, we sample the orderings of variables from the current data and estimate the probabilities of the possible observations. The edge probabilities are estimated with both the exact method by Koivisto [7] and the Markov Chain Monte Carlo (MCMC) method. Experiments show that the MCMC methods take more time to converge to the probabilities estimated with the exact method and will not be discussed.

We tested our method with the benchmark Study network and the Cold network. In the experiment, the number of instances collected from each intervention is set to 1 when the selected node is manipulated to a distinct value. Due to the computational complexity, the multiple instances from each intervention and other Bayesian networks with more variables are not tested with expected posterior loss.

Fig. 6 shows the results from the Study network. Fig. 6 (a) shows that all the methods with interventional data can reach the required structure entropy with smaller than 50 interventional instances, while the observational data alone cannot reach the requires structure entropy with the maximal instances allowed. In this example, node selection with the expected posterior loss requires the minimal number of instances to reach the structure entropy on average. The next best-performing method is node selection with non-symmetrical entropy. Node selection with symmetrical entropy and random node selection requires a larger number of instances to reach the required structure entropy. Fig. 6 (b) shows the average running time the different methods spent. We can see that the expected posterior loss requires much more time than other methods for node selection. The time for observational data converges when the maximal number of instances is reached. In summary, active learning based on expected posterior loss can reach the minimal structure entropy with the same number of interventions on average, while the computational cost makes it infeasible for

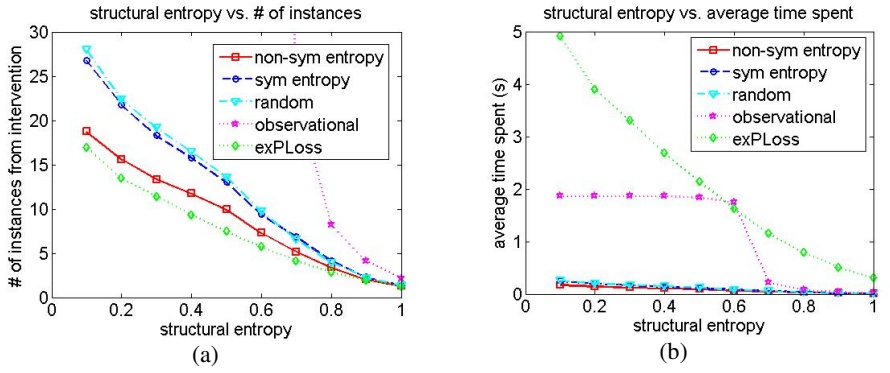


Fig. 6. Comparison of node selection methods for intervention on Study network

domains with more variables or multiple instances collected in each active learning step. The learned structure with non-symmetrical entropy is similar to that with expected posterior loss, but with much reduced computational time.

4 Discussion and Conclusion

In this work, we investigate active learning of Bayesian network structure when the interventional data is a data set at each active learning step, and propose using non-symmetrical entropy from the current data to select nodes for intervention. Experiments show that non-symmetrical entropy can reach the required structure entropy with smaller number of interventions than symmetrical entropy and random node selection for intervention, and much better than merely estimating the structure with observational data in all the Bayesian networks tested. A possible reason for the better performance of the non-symmetrical entropy is that interventions are non-symmetrical in nature. When compared with expected posterior loss, our method can reach the similar structure entropy with much lower computational complexity.

Experimental results also show that the learned structure entropy has an approximately linear relationship with the average Hamming distance from the learned structure to the ground truth Bayesian network. This implies that structure entropy is an effective measure for the goodness of the learned causal Bayesian network structure, and can be used as an effective stop criterion.

We have tested significance of the difference of the learned structure entropy from node selection based on the non-symmetrical entropy and other methods. The statistical test shows that the structure entropy from node selection with the non-symmetrical entropy is significantly smaller than that from other methods.

A surprising observation in the experiments is that the random node selection for intervention can outperform the node selection with symmetrical entropy when the number of interventions is large. When the number of interventions is large, the symmetrical entropy will often select leaf nodes for intervention, which cannot provide sufficient information to reduce the uncertainty of the edge probabilities.

Our method is not designed to replace other related work, and does not apply to domains where repeated interventions are not feasible, such as economics or social science. We have based our investigations on a set of different, complementary, or

integrated situations with respect to the previous efforts [2, 3, 7, 14]: these efforts have also inspired some technical and presentation ideas reported in this paper. There are some general directions to extend our work, such as considering missing values or hidden variables in the causal Bayesian networks. In future, we will try to extend our results to more situations and apply the method to some real-life applications in different domains.

Acknowledgement

We would like to thank Daniel Eaton and Kevin Murphy for sharing their BDAGL code publicly. This work is partially supported by an Academic Research Grant No. R-252-000-309-112 from the National University of Singapore.

References

1. Cooper, G.F., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning 9, 309–347
2. Cooper, G.F., Yoo, C.: Causal discovery from a mixture of experimental and observational data. In: Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence, pp. 116–125. Morgan Kaufmann Publishers, San Francisco (1999)
3. Eaton, D., Murphy, K.: Exact Bayesian structure learning from uncertain interventions. In: AI & Statistics, pp. 107–114 (2007)
4. Eberhardt, F., Glymour, C., Scheines, R.: On the Number of Experiments Sufficient and in the Worst Case Necessary to Identify All Causal Relations Among N Variables. In: UAI 2005, pp. 178–184. AUAI Press (2005)
5. Heckerman, D.: A Tutorial on Learning with Bayesian Networks. In: Jordan, M. (ed.) Learning in Graphical Models, pp. 301–354. MIT Press, Cambridge (1998)
6. Heckerman, D., Geiger, D., Chickering, D.M.: Learning Bayesian networks: The combination of knowledge and statistical data. Machine Learning 20, 197–243
7. Koivisto, M.: Advances in exact Bayesian structure discovery in Bayesian networks. In: UAI 2006, pp. 241–248. AUAI Press (2006)
8. Korb, K.B., Hope, L., Nicholson, A.E., Axnick, K.: Varieties of causal intervention. In: Zhang, C., Guesgen, H.W., Yeap, W.-K. (eds.) PRICAI 2004. LNCS (LNAI), vol. 3157, pp. 322–331. Springer, Heidelberg (2004)
9. Li, G.: Knowledge Discovery with Bayesian Networks, PhD Thesis Department of Computer Science, National University of Singapore, 1–210 (submitted, 2009)
10. Pearl, J.: Causality: models, reasoning, and inference. Cambridge University Press, New York (2000)
11. Sachs, K., Perez, O., Pe'er, D., Lauffenburger, D.A., Nolan, G.P.: Causal Protein-Signaling Networks Derived from Multiparameter Single-Cell Data. Science 308(5721), 523–529
12. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search, 2nd edn. MIT Press, Cambridge (2000)
13. Tian, J., Pearl, J.: Causal Discovery from Changes. In: Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI), pp. 512–521. Morgan Kaufmann, San Francisco (2001)
14. Tong, S., Koller, D.: Active Learning for Structure in Bayesian Networks. In: IJCAI 2001, pp. 863–869. Morgan Kaufmann, Washington (2001)

A Comparative Study of Bandwidth Choice in Kernel Density Estimation for Naive Bayesian Classification

Bin Liu, Ying Yang, Geoffrey I. Webb, and Janice Boughton

Clayton School of Information Technology, Monash University, Australia
{bin.liu,ying.yang,geoff.webb,janice.boughton}@infotech.monash.edu.au

Abstract. Kernel density estimation (KDE) is an important method in nonparametric learning. While KDE has been studied extensively in the context of accuracy of distribution estimation, it has not been studied extensively in the context of classification. This paper studies nine bandwidth selection schemes for kernel density estimation in Naive Bayesian classification context, using 52 machine learning benchmark datasets. The contributions of this paper are threefold. First, it shows that some commonly used and very sophisticated bandwidth selection schemes do not give good performance in Naive Bayes. Surprisingly, some very simple bandwidth selection schemes give statistically significantly better performance. Second, it shows that kernel density estimation can achieve statistically significantly better classification performance than a commonly used discretization method in Naive Bayes, but only when appropriate bandwidth selection schemes are applied. Third, this study gives bandwidth distribution patterns for the investigated bandwidth selection schemes.

1 Introduction

A critical task in Bayesian learning is estimation of the probability distributions of attributes in datasets, especially when the attributes are numeric. Traditionally, the numeric attributes are handled by discretization [1]. These methods are usually simple and computationally efficient. However, they suffer from some basic limitations [2, 3]. An alternative to calculating probability estimates for numeric attributes using discretized intervals is to estimate the probabilities directly, using an estimate of the point-wise density distribution. Both parametric and nonparametric density estimation methods have been developed.

Parametric density estimation imposes a parametric model on the observations. For example, the parameters for a Gaussian model are its sufficient statistics, the mean and variance. Normally simple parametric models do not work very well with Bayesian classification [4], as the real distributions do not exactly fit specific parametric models.

Some estimation methods, including Gaussian mixture models, use subsets of the data to obtain local models, then mix these models to obtain the density

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 302–313, 2009.
© Springer-Verlag Berlin Heidelberg 2009

estimate for all observations. In contrast, Kernel Density Estimation estimates the probability density function by imposing a model function on every data point and then adding them together. The function applied to each data point is called a kernel function. For example, a Gaussian function can be imposed on every single data point, making the center of each Gaussian kernel function the data point that it is based on. The standard deviation of the Gaussian kernel function adjusts the dispersion of the function and is called a *bandwidth* of the function.

Given sufficiently large sample data, KDE can converge to a reasonable estimate of the probability density. As there are no specific finite parameters imposed on the observations, KDE is a nonparametric method.

The univariate KDE [5, 6] can be expressed as:

$$f(x) = \frac{1}{nh} \sum_{i=1}^{n} \mathbf{K}\left(\frac{x - X_i}{h}\right), \tag{1}$$

where $\mathbf{K}(.)$ is the density kernel; x is a test instance point; X_i is a training instance point, which controls the position of the kernel function; h is the bandwidth of the kernel, which controls the dispersion of each kernel; and n is the number of data points in the data. For a univariate Gaussian kernel $\mathbf{K}(\xi) = \frac{1}{\sqrt{2\pi}} e^{-\frac{\xi^2}{2}}$.

Naive Bayes is a widely employed effective and efficient approach for classification learning, in which the predicted class label $y(x)$ of a test instance x is evaluated by $y(x) = \mathbf{argmax}_{\ c} \left[P(c) \times \prod_{i=1}^{d} P(x_i \mid c)\right]$, where $P(c)$ is the estimated class probability, d is the number of attributes, x_i is the i'th attribute of instance x, and $P(x_i \mid c)$ is the estimated probability (or probability density) of x_i given the class. KDE (Equation (1)) can be used to estimate the class conditional probabilities for numeric attributes. Because the Naive Bayesian classifier considers each attribute independently, we use only univariate kernel density estimation in this paper.

It is known that the specific choice of kernel function $\mathbf{K}$ is not critical [7]. The key challenge is the choice of the bandwidth. A bandwidth value which is too small will give a too detailed curve and hence leads to an estimation with small bias and large variance. Large bandwidth leads to low variance at the expense of increased bias.

Many bandwidth selection schemes in kernel density estimation have been studied mainly for optimizing the mean squared error loss of the estimation which supports good density curve fitting. However, bandwidth selection schemes are still not extensively studied in the classification context applying 0-1 loss criteria.

We look at the seven most commonly used bandwidth selection schemes in the statistical community plus two very simple schemes, using 52 datasets. It is shown that the choice of bandwidth dramatically affects the accuracy results of classification. An appropriate bandwidth selection scheme can achieve statistically significantly better classification performance than a commonly used discretization

method. Surprisingly, the two simple bandwidth selection schemes both achieved good performance, whereas the more sophisticated and computationally expensive schemes delivered no improvement in classification performance.

2 Bandwidth Selection Schemes

Background. Intuitively, it is assumed that there is a positive correlation between the accuracy of the probability estimates and the accuracy of classification. Friedman [8] challenged this assumption and states that more accurate probability estimates do not necessarily lead to better classification performance and can often make it worse.

Unfortunately, most bandwidth selection research considers the assumption to be true and attempts to achieve the highest possible probability estimation accuracy. These schemes are often based on a mean squared error (MSE) criteria, instead of a 0-1 loss criteria.

To the best of our knowledge, there is no practical bandwidth selection scheme that focuses on improving the classification accuracy, rather than the accuracy of the probability estimates. A recent paper [9] explores the theory of bandwidth choice in classification under limited conditions. It states that the optimal size of the bandwidth for 0-1 loss based estimation is generally the same as that which is appropriate for squared error based estimation.

Generally speaking, KDE bandwidth choice in the context of classification under 0-1 loss is more difficult than bandwidth choice under MSE loss. For example, we consider using Cross-Validation to chose optimal bandwidths in Naive Bayes, using class labels as the supervised information. Every evaluation under 0-1 loss (according to the class label) should use all attributes in the dataset. This is a global optimization problem in which the optimal bandwidth for one attribute may interact with those for other attributes. It is different to the MSE criteria which only uses the attribute under consideration.

In this section we give some theoretical descriptions of the mean squared error criteria and describe 7 bandwidth selection schemes that are based on this criteria. We also discuss two schemes which are not theoretically related to MSE.

Mean Squared Error Criteria. In probability density estimation, the Mean Squared Error (MSE) or Mean Integrated Squared Error (MISE) are the most used density estimation error criteria,

$$MISE(\hat{f}) = E \int [\hat{f}(x) - f(x)]^2 dx \,. \tag{2}$$

The integral is in the range of x, to measure how well the entire estimated curve $\hat{f}$ approximates the real curve f. The expectation operation averages over all possible samplings. From this equation, we can get $MISE(\hat{f}) = \int Bias^2[\hat{f}(x)]dx + \int Var[\hat{f}(x)]dx$, where $Bias[\hat{f}(x)] = E[\hat{f}(x)] - f(x)$ and $Var[\hat{f}(x)] = E[f^2(x)] - E^2[\hat{f}(x)]$. This equation is the starting point of the bandwidth selection scheme UCV we discuss below.

We process $E[\hat{f}(x)]$ first by using Equation (1). This leads to $E[\hat{f}(x)] = E\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}\mathbf{K}(\frac{x-X_i}{h})\right] = E\left[\frac{1}{h}\mathbf{K}(\frac{x-X}{h})\right] = \int \frac{1}{h}\mathbf{K}(\frac{x-y}{h})f(y)dy$, where for each test point x, we regard each X_i as an independent and identically distributed random variable with distribution f. Making a simple variable substitution $y = x - ht$, we obtain: $Bias[\hat{f}(x)] = \int \mathbf{K}(t)[f(x-ht) - f(x)]dt$. A Taylor series expansion $f(x-ht) \approx f(x) - htf'(x) + \frac{1}{2}h^2t^2f''(x)$ can be substituted into this equation. The first term of $f(x-ht)$ is canceled out by the negative $f(x)$. The second term is also canceled out because the $K(t)$ in the integral is a symmetric function. So, $\int Bias^2[\hat{f}(x)]dx \approx \frac{1}{4}h^4(\int t^2K(t)dt)^2\int(f''(x))^2dx = \frac{1}{4}h^4\mu_2^2(K)R(f'')$, where $R(g) = \int g^2(x)dx$ and $\mu_2(g) = \int x^2g(x)dx$.

In a similar way, we can get, $Var[\hat{f}(x)] = \frac{1}{nh}R(K)$. The elementary Equation (2) becomes an asymptotic form, as the error term in Taylor expansion is the higher-order term of h, which monotonously decreases when samples grow. The asymptotic mean integrated squared error is,

$$AMISE = \frac{1}{nh}R(K) + \frac{1}{4}h^4\mu_2^2(K)R(f'')\,. \tag{3}$$

This equation is the starting point for the bandwidth selection schemes BCV, STE and DPI, which are discussed below.

Unbiased Cross-Validation (UCV) Scheme. The method of Unbiased Cross-Validation [10] is based on the elementary Equation (2). It is also called least squares cross-validation. UCV obtains a score function to estimate the performance of candidate bandwidth. In practice, UCV minimizes the integrated square error, the Equation (4), which uses one realization of samples from the underlaying distribution f.

$$ISE = \int[\hat{f}(x) - f(x)]^2dx = R(\hat{f}) - 2\int\hat{f}(x)f(x)dx + R(f)\,, \tag{4}$$

where $R(g)$ is similar to Equation (3).

Notice the first term in Equation (4) is only related to the estimated $\hat{f}(x)$, so it is easy to process given a specific bandwidth $\hat{h}$. The third term is independent of the estimated $\hat{h}$ and remains constant for all estimations, so it can be ignored. The second term can be written as $\int\hat{f}(x)f(x)dx = E[\hat{f}(x)]$, i.e., it is the statistic mean of $\hat{f}(x)$ with respect to x.

If we get n samples of x, for the sake of obtaining a stable estimation of $E[\hat{f}(x)]$, we can use a Leave-One-Out method to get an n-points estimation value of the $\hat{f}(x)$. The Leave-One-Out method estimates the value of $\hat{f}(x_i)$ by leaving the x_i out and using the other n-1 points of x. This is why this method is called a Cross-Validation. We use $\hat{f}_{-i}(x_i)$ to express this Leave-One-Out estimation, which is evaluated from Equation (1). Then, $E[\hat{f}(x)] = \frac{1}{n}\sum_{i=1}^{n}\hat{f}_{-i}(x_i)$. Substituting this to Equation (4) we construct a score function in the sense of

ISE. Now for some specific candidate bandwidth $\hat{h}$, we can give a unbiased cross validation score for the candidate bandwidth $\hat{h}$ as,

$$UCV(\hat{h}) = R(\hat{f}) - \frac{2}{n}\sum_{i=1}^{n}\hat{f}_{-i}(x_i)\,.$$

We can use a start bandwidth as a reference estimation, and make a brute-force search near this reference bandwidth with respect to the minima of UCV score function.

Normal Reference Density (NRD-I, NRD and NRD0) Schemes. Normal Reference Density [5] scheme is also called the Rule of Thumb scheme. It is based on Equation (3). To minimize AMISE, a simple first order differential can be used on Equation (3) towards the bandwidth h and setting the differential to zero. The optimal bandwidth is:

$$\hat{h}_{AMISE} = \left[\frac{R(K)}{\mu_2^2(K)R(f'')}\right]^{1/5} n^{-1/5}\,. \tag{5}$$

This result still depends on the unknown density derivative function $f''(x)$, which will depend on h recursively again. Normal Reference Density scheme simplifies this problem by using a parametric model, say, a Gaussian to estimate $f''(x)$. Compared with the Cross-validation selection, this is a straightforward method and can lead to an analytical expression of bandwidth $\hat{h} = 1.06\,\hat{\sigma}n^{-1/5}$, where n is the number of samples and $\hat{\sigma}$ is the estimated normal distribution standard deviation of the samples.

This bandwidth selection scheme is a classic one. We use this bandwidth as a standard bandwidth in our experiments. We call this scheme *NRD-I*.

A more robust approach [5] can be applied by considering the interquartile range (IQR). The bandwidth is calculated from the minimum of standard deviation and standard IQR: $\hat{h} = 1.06\,\mathbf{min}\,(\hat{\sigma}, IQR/1.34)\,n^{-1/5}$. This procedure [5] helps to lessen the risk of oversmoothing. We call the bandwidth the NRD bandwidth in this paper. A smaller version of NRD suggested in R [11] is $\hat{h} = 0.9\,\mathbf{min}\,(\hat{\sigma}, IQR/1.34)\,n^{-1/5}$. We call this bandwidth the NRD0.

Biased Cross-Validation (BCV) Scheme. Biased cross-validation uses Equation (3) as the basis of the score function. Scott and Terrel [12] develop an estimation of $R(f'')$ in Equation (3), using $\hat{R}(f'') = R(\hat{f}'') - \frac{1}{nh^5}R(K'')$, where f'', $\hat{f}''$ and K'' are second-order derivatives of distribution and kernel respectively. The right hand side of this estimation can be evaluated given a specific bandwidth $\hat{h}$. Substituting the $\hat{R}(f'')$ to Equation (3), we can get a new score function,

$$BCV(\hat{h}) = \frac{1}{nh}R(K) + \frac{1}{4}h^4\mu_2^2(K)[R(\hat{f}'') - \frac{1}{nh^5}R(K'')]\,.$$

A exhaustive search procedure similar to UCV scheme can be applied to find the optimal bandwidth.

Direct-Plug-In (DPI) Scheme and Solve-The-Equation (STE) Scheme. The Direct-Plug-In scheme [13] is a more complicated version of the Normal Reference Density scheme. It seeks $R(f'')$ by estimation of $R(f'''')$. This problem continues because the $R(f^{(s)})$ will depend on $R(f^{(s+2)})$. Normally, for a specific s, $R(f^{(s+2)})$ is estimated by a simple parametric method, to obtain $R(f^{(s)})$ and so on. We call Direct-Plug-In Scheme the DPI in our experiments.

Notice that Equation (5) is a fixed point equation $h = F(h)$, where $F(h) = \left[\frac{R(K)}{\mu_2^2(K)R(f'')}\right]^{1/5} n^{-1/5}$. and $R(f'')$ is a function of h. Solve-The-Equation Scheme [6, 13] is applied by solving the fixed point of $F(h)$. We call Solve-The-Equation scheme the STE in our experiments.

Two Very Simple (WEKA and SP) Schemes. We use two very simple bandwidth selection schemes. These two schemes are both based on the range of data divided by a measure of the size of the samples. There is less theoretical consideration [4, 14, 15] of these methods compared with the other methods discussed above. They merely conform to the basic requirement in KDE that when the number of samples approaches infinity, the bandwidth approaches zero.

One scheme uses $\sqrt{n}$ as a division factor [4], so the bandwidth approaches zero as n increases,

$$\hat{h} = \frac{range(x)}{\sqrt{n}},$$

where n is the number of samples, $range(x)$ is the range of values of x in training data. This scheme is used in WEKA [14], with some calibration that $\hat{h}$ should be no less than $\frac{1}{6}$ of the average data interval, which avoids $\hat{h}$ becoming too small compared with the average data interval. We call this scheme WEKA.

The other scheme is a very old scheme[16].

$$\hat{h} = \frac{range(x)}{2(1 + log_2 n)}.$$

The basic principle of this equation does not have very strong theoretical basis [15]. However it was widely used in the old version of S-PLUS statistic package (up to version 5.0) [17, page 135]. We call it the SP scheme.

3 Experiments

3.1 Data and Design

In addition to the nine bandwidth selection schemes described in Section 2, the widely used MDL discretization method [1] was also used as a performance reference. The Naive Bayesian classifier was the classifier used for all schemes being evaluated. Every statistic sample (every dataset, every experiment trial and fold) and every piece of classifier code is the same for all schemes. The only difference between each scheme in the classifier algorithm is the bandwidth of the kernel.

Table 1. The 52 experimental datasets, with the numbers of instances, classes, attributes and numeric attributes

Data	Ins.	Cls.	Att.	NAtt.	Data	Ins.	Cls.	Att.	NAtt.
Abalone	4177	3	8	8	Letter	20000	26	16	16
Adult	48842	2	14	6	Liver-disorders	345	2	6	6
Anneal	898	6	38	6	Lymph	148	4	18	3
Arrhythmia	452	16	279	206	Mfeat-factors	2000	10	216	216
Autos	205	7	25	15	Mfeat-fourier	2000	10	76	76
Backache	180	2	32	6	Mfeat-karhunen	2000	10	64	64
Balance-scale	625	3	4	4	Mfeat-morphological	2000	10	6	6
Biomed	209	2	8	7	Mfeat-zernike	2000	10	47	47
Cars	406	3	7	6	New-thyroid	215	3	5	5
Cmc	1473	3	9	2	Optdigits	5620	10	64	64
Collins	500	15	23	20	Page-blocks	5473	5	10	10
German	1000	2	20	7	Pendigits	10992	10	16	16
Crx(credit-a)	690	2	15	6	Prnn-synth	250	2	2	2
Cylinder-bands	540	2	39	18	Satellite	6435	6	36	36
Diabetes	768	2	8	8	Schizo	340	2	14	12
Echocardiogram	131	2	6	5	Segment	2310	7	19	19
Ecoli	336	8	7	7	Sign	12546	3	8	8
Glass	214	7	9	9	Sonar	208	2	60	60
Haberman	306	2	3	2	Spambase	4601	2	57	57
Heart-statlog	270	2	13	13	Syncon	600	6	61	60
Hepatitis	155	2	20	6	Tae	151	3	5	3
Horse-colic	368	2	21	8	Vehicle	846	4	18	18
Hungarian	294	2	13	6	Vowel	990	11	13	10
Hypothyroid	3772	4	29	7	Waveform-5000	5000	3	40	40
Ionosphere	351	2	34	34	Wine	178	3	13	13
Iris	150	3	4	4	Zoo	101	7	17	1

The fifty-two datasets used in the experiments were drawn from the UCI machine learning repository [18] and the web site of WEKA [14]. We use all the datasets that we could identify from these places, given the dataset has at least one numeric attribute and has at least 100 instances. Table 1 describes these datasets. Any missing values occurring in the data for numeric attributes were replaced with the mean average value for that attribute.

Each scheme was tested on each dataset using a 30-trial 2-fold cross validation bias-variance decomposition. A large number of trials was chosen because bias-variance decomposition has greater accuracy when a sufficiently large number of trials are conducted [19]. Selecting two folds for the cross-validation maximizes the variation in the training data from trial to trial.

Thirty trials and two folds yields sixty Naive Bayesian classification evaluations for each dataset. For these evaluations we recorded the mean training time, mean error rate, mean bias and mean variance. Kohavi and Wolpert's method [20] of bias and variance decomposition was employed to determine the bias and variance based on the obtained error rate.

Since there are nine alternative KDE classifiers and one discretization classifier, we get ten comparators of the performance measure for each dataset.

After the classification performance comparison, we also produce a statistic for the bandwidth distribution for alternative bandwidth selection schemes. The fifty-two datasets contain 1294 numeric attributes collectively. Every numeric attribute has at least two and at most 26 class labels. Since we evaluate the KDE for every class conditional probability, there are 10967 class conditional

probability evaluation objects. Each of these evaluation objects produces 60 different realization samples by the 30 trails 2 fold cross-validation. Every bandwidth selection scheme is applied to each realization of the conditional probability evaluation objects, and produces an estimated bandwidth for that realization.

These bandwidths are transformed to a ratio to a standard bandwidth. We use the NRD-I bandwidth as the standard. By using these bandwidth ratios, we get a statistical distribution of the bandwidth size for each scheme.

3.2 Observations and Analysis

Classification Error, Bias and Variance. We use Friedman's method [21] to rank classification error, bias and variance. The scheme that performs the best is ranked 1, the second best is ranked 2 and so forth. The mean rank of classification accuracy and time measure (real time) are summarized in Figure 1 as the shaded bars. Since the bandwidth calculations are carried out during training, the computational time for the test stage is essentially the same for all schemes and therefore is not reported.

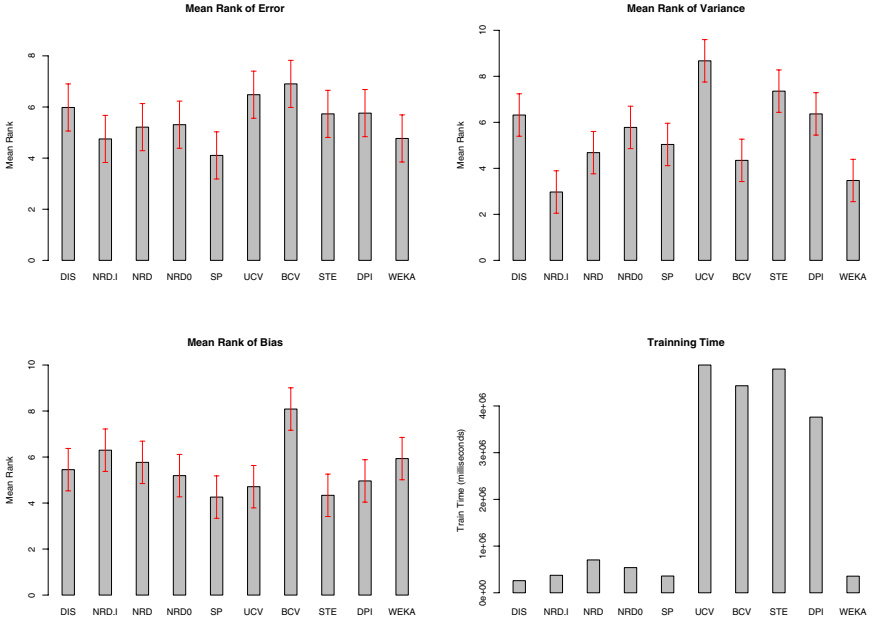


Fig. 1. Comparison of alternative methods' mean ranks of classification accuracy. Classification error can be decomposed into bias and variance. The shaded bars illustrate the mean rank and the smaller rank has the better performance. The line segments accompanying each bar indicate the Nemenyi test results. The performance of two methods are statistically significantly different if their vertical line segments are not overlapping. The mean training time is real time of computation.

Table 2. Comparison of rival schemes' win/tie/lose records with regard to classification error, bias and variance. Each three-number entry indicates the number of times the scheme named in the row wins, ties and loses against the scheme named in the column. A statistically significant record (at the 0.05 critical level) is indicated in a bold face.

(a) ERROR

w/t/l	DIS	NRD-I	NRD	NRD0	SP	UCV	BCV	STE	DPI
NRD-I	32/0/20								
NRD	30/0/22	22/1/29							
NRD0	28/0/24	22/0/30	25/1/26						
SP	**33/0/19**	32/0/20	34/2/16	34/0/18					
UCV	24/0/28	21/0/31	17/0/35	17/0/35	**14/0/38**				
BCV	26/0/26	9/0/43	15/0/37	16/1/35	**15/1/36**	23/0/29			
STE	25/1/26	19/0/33	23/0/29	25/0/27	18/0/34	33/1/18	29/0/23		
DPI	28/1/23	23/1/28	21/0/31	22/1/29	16/1/35	31/1/20	30/0/22	24/1/27	
WEKA	32/0/20	26/0/26	31/0/21	28/1/23	23/1/28	30/0/22	41/0/11	30/1/21	29/1/22

(b) BIAS

w/t/l	DIS	NRD-I	NRD	NRD0	SP	UCV	BCV	STE	DPI
NRD-I	22/0/30								
NRD	25/1/26	31/1/20							
NRD0	26/0/26	34/1/17	33/0/19						
SP	28/0/24	39/0/13	37/0/15	33/0/19					
UCV	27/0/25	32/0/20	29/0/23	31/1/20	27/0/25				
BCV	**19/0/33**	12/0/40	**12/0/40**	**9/0/43**	**8/0/44**	**13/0/39**			
STE	28/0/24	35/0/17	32/0/20	30/0/22	28/1/23	26/0/26	**45/1/6**		
DPI	29/0/23	35/1/16	33/0/19	31/0/21	20/0/32	20/1/31	**44/0/8**	18/0/34	
WEKA	27/0/25	26/0/26	25/0/27	21/0/31	15/0/37	20/0/32	**40/0/12**	16/1/35	21/0/31

(c) VARIANCE

w/t/l	DIS	NRD-I	NRD	NRD0	SP	UCV	BCV	STE	DPI
NRD-I	**37/1/14**								
NRD	34/0/18	8/1/43							
NRD0	34/0/18	**8/0/44**	11/0/41						
SP	32/0/20	**14/0/38**	26/0/26	30/0/22					
UCV	**20/0/32**	**3/0/49**	**5/0/47**	**5/0/47**	**7/0/45**				
BCV	32/0/20	16/3/33	31/0/21	36/0/16	29/1/22	**46/0/6**			
STE	24/0/28	**6/0/46**	**10/0/42**	13/0/39	**13/0/39**	42/0/10	**9/0/43**		
DPI	27/0/25	**8/0/44**	11/0/41	22/1/29	18/1/33	**43/0/9**	**11/1/40**	37/1/14	
WEKA	36/0/16	23/0/29	36/0/16	**39/0/13**	35/2/15	**48/0/4**	33/1/18	**46/0/6**	**42/0/10**

A win/tie/lose record (w/t/l) is calculated for each pair of competitors A and B with regard to a performance measure M. The record represents the number of datasets in which A wins loses or ties with B on M. The win/tie/loss records are summarized in Table 2.

We also apply statistical comparison methods of multiple classifiers over multiple data sets recommended by Demsar [22].

The null hypothesis was rejected for all Friedman tests (using the 0.05 critical level) conducted on error, bias and variance, so we can infer that there exists significant difference among all ten schemes.

Having determined that a significant difference exists, the post-hoc Nemenyi test was used to identify which pairs of schemes differ significantly. The results of this test(using the 0.05 critical level) are illustrated by the line segments accompanying each bar in the graph in Figure 1. The length of these lines indicate the critical difference, and the performance of two methods are considered to be significantly different if the difference between their mean rank is greater than the critical difference (i.e. their vertical line segments do not overlap).

Figure 1 and Table 2 show that the more sophisticated bandwidth selection schemes investigated do not yield improved performance over simpler schemes, although they are far more computationally expensive. The poorest performer was BCV, which was statistically significantly worse than the more simplistic SP scheme (with w/t/l record 15/1/36) and WEKA scheme (with w/t/l record 11/0/41). UCV was also determined to be statistically significantly worse than the SP scheme (with /w/t/l record 18/0/36). The computational time costs of the four sophisticated schemes are far more than the others.

The UCV scheme achieved low bias, but high variance, as stated by its name. Conversely, BCV achieved low variance, but high bias. Neither the SP scheme's bias nor its variance was particularly high or low, and it was found to be statistically significantly better than the discretization method and the two worst sophisticated bandwidth selection schemes, UCV and BCV. This analysis shows that the choice of bandwidth dramatically affects the accuracy results of classification. The more sophisticated schemes can not guarantee good classification performance. Trade-off between bias and variance performance is essential to improve upon classification accuracy.

This analysis also shows that only one bandwidth selection scheme (the SP scheme) gives statistically better performance than a classical discretization method. It suggests that KDE can achieve statistically significantly better

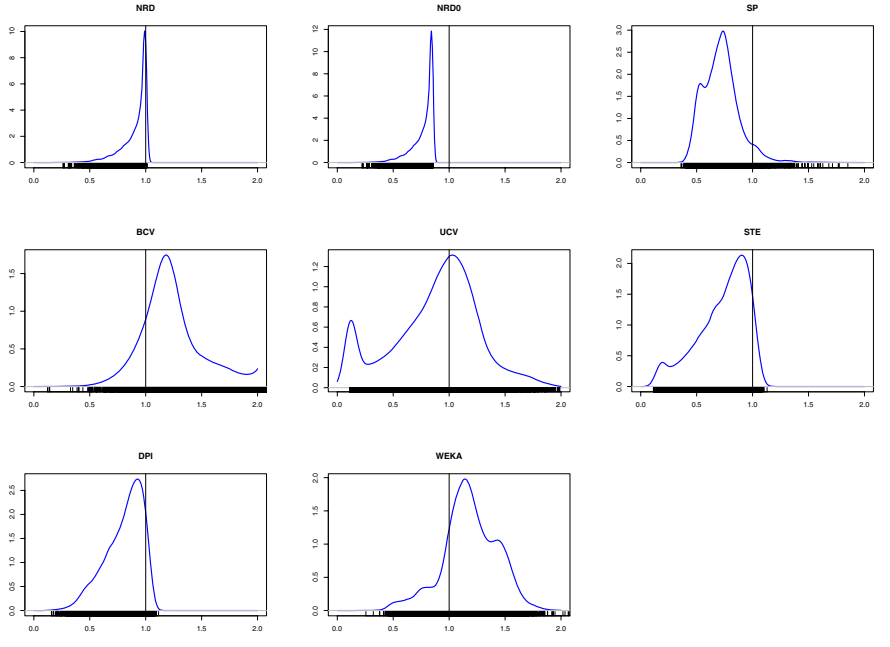


Fig. 2. Distribution of the size of bandwidth. X-axis is the ratio of alternative bandwidth to a standard bandwidth. Y-axis is the density of the ratio distribution. Standard bandwidth is NRD-I.

performance in classification, but the bandwidth selection schemes in classification behave different with traditional sophisticated bandwidth selection schemes. More theoretical researches are needed for kernel density estimation in classification.

Distribution of the Bandwidth. The distribution of the bandwidth size for each scheme is illustrated in Figure 2. By comparing Figure 1 and Figure 2 we can see that the bandwidth of BCV and WEKA is statistically larger than others. This gives them a small variance and large bias in classification. By contrast, NRD0, SP, STE and DPI tend to have smaller bandwidths. This gives them a relatively small bias and large variance in classification. We can see that there is a transition range (from approximately 0.5 to 1.5 times of NRD-I bandwidth) that indicates a change in tendency of bias-variance trade off, from a low-bias high-variance to a high-bias low-variance profile. This transition range is narrow. This relatively narrow distribution range shows that classification performance is more sensitive to the size of the bandwidth than was first thought.

4 Conclusions

The more simplistic and less computationally intensive bandwidth selection schemes performed significantly better compared to some of the more sophisticated schemes in Naive Bayesian Classification. A kernel density estimation method can significantly outperform a classical discretization method, but only when appropriate bandwidth selection schemes are applied.

Our experiments and analysis also show that an unsuitable bandwidth value can easily give poor classification performance. In a relatively narrow distribution range, we find that the bias-variance trade off changes, from low-bias and high-variance to high-bias and low-variance. Comparison of the bandwidth distribution patterns with error performance suggests that bandwidths within the range of 0.5 to 1.5 times NRD-I standard bandwidth are preferable.

Acknowledgements

The authors thank Dr. Eibe Frank and Dr.Leonard Trigg for the helpful discussion and Mr. Colin Enticott for the support of cluster computing. This work is supported by Australian Research Council grant DP0770741.

References

[1] Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence, vol. 2, pp. 1022–1027 (1993)

[2] Yang, Y., Webb, G.: Discretization for naive-bayes learning: managing discretization bias and variance. Machine Learning (2008) Online First

[3] Bay, S.D.: Multivariate discretization for set mining. Knowledge and Information Systems 3(4), 491–512 (2001)
[4] John, G.H., Langley, P.: Estimating continuous distributions in bayesian classifiers. In: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, pp. 338–345 (1995)
[5] Silverman, B.W.: Density Estimation for Statistics and Data Analysis, 1st edn. Chapman & Hall/CRC (1986)
[6] Wand, M.P., Jones, M.C.: Kernel Smoothing. Chapman & Hall/CRC (1994)
[7] Epanechnikov, V.A.: Non-parametric estimation of a multivariate probability density. Theory of Probability and its Applications 14(1), 153–158 (1969)
[8] Friedman, J.H.: On bias, variance, 0/1-loss, and the curse-of-dimensionality. Data Mining and Knowledge Discovery 1(1), 55–77 (1997)
[9] Hall, P., Kang, K.H.: Bandwidth choice for nonparametric classification. Annals of Statistics 33(1), 284–306 (2005)
[10] Bowman, A.W.: An alternative method of cross-validation for the smoothing of density estimates. Biometrika 71(2), 353–360 (1984)
[11] R Development Core Team: R: A Language and Environment for Statistical Computing, Austria, Vienna (2008), http://www.R-project.org
[12] Scott, D.W., Terrell, G.R.: Biased and unbiased cross-validation in density estimation. Journal of the American Statistical Association 82(400), 1131–1146 (1987)
[13] Sheather, S.J., Jones, M.C.: A reliable data-based bandwidth selection method for kernel density estimation. Journal of the Royal Statistical Society. Series B 53(3), 683–690 (1991)
[14] Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
[15] Hyndman, R.J.: The problem with sturge's rule for constructing histograms (1995), http://www--personal.buseco.monash.edu.au/~hyndman/papers
[16] Sturges, H.A.: The choice of a class interval. Journal of the American Statistical Association 21(153), 65–66 (1926)
[17] Venables, W.N., Ripley, B.D.: Modern Applied Statistics with S-PLUS, 3rd edn. Springer, Heidelberg (1999)
[18] Asuncion, A., Newman, D.J.: UCI Machine Learning Repository (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html
[19] Webb, G.I.: Multiboosting: A technique for combining boosting and wagging. Machine Learning 40(2), 159–196 (2000)
[20] Kohavi, R., Wolpert, D.H.: Bias plus variance decomposition for zero-one loss functions. In: Machine Learning: Proceedings of the Thirteenth International Conference, vol. 275, p. 283 (1996)
[21] Friedman, M.: The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association 32(200), 675–701 (1937)
[22] Demsar, J.: Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research 7, 1–30 (2006)

Analysis of Variational Bayesian Matrix Factorization

Shinichi Nakajima[1] and Masashi Sugiyama[2]

[1] Nikon Corporation, 1-6-3 Nishi-Ohi, Shinagawa-ku, Tokyo 140-8601, Japan
[2] Tokyo Institute of Technology, 2-12-1 O-okayama, Meguro-ku, Tokyo 152-8552, Japan
nakajima.s@nikon.co.jp, sugi@cs.titech.ac.jp
http://watanabe-www.pi.titech.ac.jp/~nkj23/index.html

Abstract. Recently, the variational Bayesian approximation was applied to probabilistic matrix factorization and shown to perform very well in experiments. However, its good performance was not completely understood beyond its experimental success. The purpose of this paper is to theoretically elucidate properties of a variational Bayesian matrix factorization method. In particular, its mechanism of avoiding overfitting is analyzed. Our analysis relies on the key fact that the matrix factorization model induces non-identifiability, i.e., the mapping between factorized matrices and the original matrix is not one-to-one. The positive-part James-Stein shrinkage operator and the Marcenko-Pastur law—the limiting distribution of eigenvalues of the central Wishart distribution—play important roles in our analysis.

1 Introduction

The problem of estimating a matrix that describes a linear relation between two vectors has been extensively studied by the name of multivariate linear regression with multiple responses, canonical correlation analysis, or reduced rank regression [1]. On the other hand, a recent focus of matrix estimation includes imputation of missing entries of a single matrix, e.g., in the context of microarray data analysis [2] and recommender systems[1] [3,4]. In this paper, we consider the problem of interpolating missing entries of a matrix.

The paper [5] proposed the weighted low-rank approximation (WLRA) method, which based on the expectation-maximization (EM) algorithm: a matrix is fitted to the data without a rank constraint in the E-step and it is projected back to the set of low-rank matrices by the singular value decomposition (SVD) in the M-step. The web article [4] proposed the regularized SVD method, which minimizes the loss function combined with the Frobenius-norm penalty by gradient descent. If the *trace-norm* penalty is used instead of the Frobenius-norm penalty, a low-rank solution can be obtained without having an explicit low-rank constraint; when the trace-norm penalty is combined with the hinge-loss, a semi-definite programming formulation is obtained [6] (see also [7]

[1] A recommender system is aimed at predicting a preference score of users based on the preference scores of other users. If we consider a matrix where each row corresponds to each user and each column corresponds to each item, the task can be regarded as completing missing entries. This formulation is often referred to as *collaborative filtering*.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 314–326, 2009.
© Springer-Verlag Berlin Heidelberg 2009

for a gradient method with smooth approximation). When the trace-norm penalty and the squared-loss are used, a computationally efficient algorithm is obtained [8].

The above methods, minimizing a loss function combined with a regularizer, could be viewed as Bayesian MAP estimation. On the other hand, it is said that full-Bayesian estimation (considering the entire posterior distribution) is often more accurate than only taking into account the mode of the posterior distribution [9]. However, working with the entire posterior distribution is often computationally intractable, and the variational approximation [10] is known to be a useful approach to coping with this problem. Following this idea, the papers [11,12] proposed variational Bayesian (VB) approaches to matrix factorization and experimentally showed their superior performance.

In this paper, we try to give a theoretical insight into the experimental facts that the VB approach often has better performance; more specifically, we investigate how the VB method avoids overfitting and why a low-rank solution tends to be produced. We first show that the VB solution can be regarded as a type of the *positive-part James-Stein shrinkage estimator* [13], which is known to *dominate* the least-squares estimator under some conditions. Our analysis strongly relies on the fact that the matrix factorization model induces *non-identifiability*, i.e., the decomposition is redundant and therefore the mapping between factorized matrices and the original matrix is not one-to-one [14,15]. We then analyze the generalization performance of the VB solution based on the *Marcenko-Pastur law* [16,17], which elucidates the limiting distribution of eigenvalues of the *central Wishart distribution*.

2 Probabilistic Approach to Matrix Factorization

In this section, we first give a probabilistic formulation of the matrix factorization problem and then review existing approaches.

2.1 Formulation

Let us consider the problem of estimating a target matrix X ($\in \mathbb{R}^{L\times M}$) from its observation Y ($\in \mathbb{R}^{L\times M}$). In the case of recommender systems, the entry $X_{l,m}$ represents the preference score (e.g., one to five stars) given by the l-th user for the m-th item. Assume that the observed matrix Y is subject to the following additive-noise model:

$$Y = X + \mathcal{E}, \tag{1}$$

where $\mathcal{E}$ ($\in \mathbb{R}^{L\times M}$) is a noise matrix. Assuming that each entry of $\mathcal{E}$ is independently subject to the normal distribution with mean zero and variance σ^2, we have the following likelihood:

$$p(Y|X) \propto \exp\left(-\tfrac{1}{2\sigma^2}\|Y - X\|_{\mathrm{Fro}}^2\right), \tag{2}$$

where $\|\cdot\|_{\mathrm{Fro}}^2$ denotes the Frobenius norm of a matrix.

If there are missing entries in the observation Y, the likelihood is expressed as

$$p(Y|X) \propto \exp\left(-\tfrac{1}{2\sigma^2}\|W \bullet (Y - X)\|_{\mathrm{Fro}}^2\right), \tag{3}$$

where $\bullet$ denotes the Hadamard product (or the element-wise product) and W is the $L \times M$ matrix with $W_{l,m} = 0$ if $Y_{l,m}$ is a missing entry; otherwise $W_{l,m} = 1$.

Let[2] $H = \min(L, M)$, and let us decompose the matrix X into the product of $A \in \mathbb{R}^{M \times H}$ and $B \in \mathbb{R}^{L \times H}$: $X = BA^\top$, where $\top$ denotes the transpose of a matrix/vector. Then, the likelihood (3) is written as

$$p(Y|A, B) \propto \exp\left(-\frac{1}{2\sigma^2}\|W \bullet (Y - BA^\top)\|_{\mathrm{Fro}}^2\right). \quad (4)$$

2.2 Maximum A Posteriori (MAP) Estimation

An estimate of X can be obtained by a procedure similar to the expectation-maximization (EM) algorithm in latent variable models[3]. Let Z ($\in \mathbb{R}^{L \times M}$) be a (latent) complete observation matrix, and let $\phi(X)$ be the prior distribution of X. Then the maximum a posteriori (MAP) solution can be obtained by the following EM algorithm:

E-step: $Z^{(t)} = W \bullet Y + (1 - W) \bullet X^{(t)}$,
M-step: $X^{(t+1)} = \mathrm{argmax}_X\, p(Z^{(t)}|X)\phi(X)$,

where $t = 0, 1, \ldots$ is the iteration number. The M-step corresponds to MAP estimation given a fully observed matrix $Z^{(t)}$, while the E-step updates the latent variable $Z^{(t)}$.

Weighted Low-Rank Approximation (WLRA): To avoid overfitting, the paper [5] proposed the WLRA method, which approximates the matrix X with a given rank H' ($\leq H$). This can be regarded as the ℓ_0-norm constraint on the singular values. The WLRA method can be obtained from the following prior distribution on X: $\phi(X) \propto \exp\left(-\frac{1}{c^2}\sum_{h=1}^{H}\theta(\widehat{\gamma}_h > 0)\right)$, where $\theta(\cdot)$ denotes the indicator function, $\widehat{\gamma}_h$ is the h-th largest singular value of X, and c^2 is a constant determined by H'. Then the M-step yields

M-step: $X^{(t+1)} = \sum_{h=1}^{H'} \gamma_h \boldsymbol{\omega}_{b_h} \boldsymbol{\omega}_{a_h}^\top$,

where γ_h is the h-th largest singular value of Z, and $\boldsymbol{\omega}_{a_h}$ and $\boldsymbol{\omega}_{b_h}$ are the corresponding right and the left singular vectors, respectively. Thus the WLRA algorithm sharply cuts off irrelevant singular values for avoiding overfitting.

Matrix Estimation with Trace-Norm Regularization (METR): Another possibility of avoiding overfitting would be regularization—the METR method employs the *trace-norm regularization*, which imposes the ℓ_1-norm constraint on the singular values [8]. METR can be obtained from the following prior distribution on X: $\phi(X) \propto \exp\left(-\frac{1}{c^2}\sum_{h=1}^{H}\widehat{\gamma}_h\right)$, where c^2 is a hyperparameter. The M-step yields

[2] Although we can analyze the case that $H < \min(L, M)$ in a similar way, we assume $H = \min(L, M)$ for the sake of simplicity.

[3] Note that there are other computationally efficient approaches to obtaining a MAP solution. However, the purpose of the review here is not to discuss computational issues, but to compare regularization schemes. For this reason, we focus on the EM algorithm.

M-step: $X^{(t+1)} = \sum_{h=1}^{H} \max\left(0, \gamma_h - \frac{\sigma^2}{c^2}\right) \boldsymbol{\omega}_{b_h} \boldsymbol{\omega}_{a_h}^\top$.

Note that the METR method can also be obtained as MAP estimation when Gaussian priors are assumed on A and B as Eq.(10) [6].

Matrix Estimation with Frobenius Regularization (MEFR): Another regularization approach is to use the *Frobenius regularization*. The MEFR method imposes the ℓ_2-norm constraint on the singular values. MEFR is obtained from the following prior distribution on X: $\phi(X) \propto \exp\left(-\frac{1}{2c^2}\sum_{h=1}^{H} \widehat{\gamma}_h^2\right)$. The M-step yields

M-step: $X^{(t+1)} = \sum_{h=1}^{H} \left(1 - \frac{\sigma^2}{\sigma^2 + c^2}\right) \gamma_h \boldsymbol{\omega}_{b_h} \boldsymbol{\omega}_{a_h}^\top$.

However, the MEFR method is not useful in missing entry completion as it is since it only proportionally shrinks the original matrix and therefore missing values are always zero. Thus the MEFR method should be combined with a low-rank constraint [4].

Maximum-Margin Matrix Factorization (MMMF): The paper [6] proposed a matrix factorization method called MMMF, which involves the trace-norm regularization similar to the METR method, but employs the *hinge-loss* inspired by the large-margin principle of support vector machines. For the binary observation Y ($\in \{\pm 1\}^{L \times M}$), the MMMF optimization problem is expressed as

$$\min_X \left[\sum_{h=1}^{H} \widehat{\gamma}_h + \lambda \sum_{l,m} \max(0, 1 - Y_{l,m} X_{l,m})\right],$$

where λ is a regularization parameter and $\sum_{l,m}$ goes over all non-missing entries of Y.

The MMMF method could also be regarded as MAP estimation with the same prior as METR; but the noise model is different from Eq.(2).

2.3 Variational Bayes (VB) Estimation

The papers [11,12] proposed matrix factorization algorithms based on the VB approach [10] to approximating the posterior $p(A, B|Y)$.

Let $\phi(A)$ and $\phi(B)$ be priors on the factors A and B. Then the posterior distribution of A and B is written as follows:

$$p(A, B|Y) = \frac{\phi(A)\phi(B)p(Y|A,B)}{\int \phi(A)\phi(B)p(Y|A,B)dAdB}. \tag{5}$$

This is the minimizer of the following functional, called the free energy, with respect to $r(A, B)$:

$$F(r|Y) = \int r(A, B) \log \frac{r(A,B)}{p(Y|A,B)\phi(A)\phi(B)} dAdB. \tag{6}$$

The VB approach approximates the posterior $p(A, B|Y)$ within a function class where A and B are independent of each other:

$$r(A, B) = r(A)r(B). \tag{7}$$

Then, using the variational method to minimize Eq.(6), we obtain the following conditions:

$$r(A) \propto \phi(A) \exp\left(\langle \log p(Y|A,B) \rangle_{r(B)}\right), \tag{8}$$

$$r(B) \propto \phi(B) \exp\left(\langle \log p(Y|A,B) \rangle_{r(A)}\right), \tag{9}$$

where $\langle \cdot \rangle_p$ is the expectation over a distribution p. Since $p(Y|A,B)$ is bilinear with respect to A and B (see Eq.(4)), the expectations in Eqs.(8) and (9) can be calculated simply by using the Gaussian integration.

Let us assume the Gaussian priors on the factors A and B:

$$\phi(A) \propto \exp\left(-\tfrac{1}{2c_a^2}\|A\|_{\mathrm{Fro}}^2\right) \quad \text{and} \quad \phi(B) \propto \exp\left(-\tfrac{1}{2c_b^2}\|B\|_{\mathrm{Fro}}^2\right), \tag{10}$$

where c_a^2 and c_b^2 are hyperparameters corresponding to the prior variance of the elements of A and B, respectively. Then the conditions (8) and (9) show that the posterior is also Gaussian. Based on this property, the papers [11,12] proposed algorithms that iteratively update the mean and the covariance of A and B by Eqs.(8) and (9), respectively. Then the posterior *mean* of $BA^\top$, i.e., $\langle BA^\top \rangle_{r(A,B)}$, is outputted[4] as an estimate of X.

3 Analysis of the VB Approach

VB estimation in general is shown to be a useful alternative to MAP estimation [10,9], and its good performance has been theoretically investigated in the light of model *non-identifiability*—a statistical model is said to be non-identifiable if the mapping between a parameter value and a probability distribution is not one-to-one [14,15].

The VB-based matrix factorization methods reviewed in Section 2.3 are shown to work well in experiments [11,12]. However, their good performance was not completely understood beyond their experimental success. In this section, we theoretically investigate properties of a VB-based matrix factorization method. Note that the factorized matrix model (4) is also non-identifiable since the mapping between (A,B) and X is not one-to-one.

In order to make the analysis feasible, let us consider a variant of VB-based matrix factorization which consists of the following VBEM iterations:

VBE-step: $Z^{(t)} = W \bullet Y + (1-W) \bullet X^{(t)}$,
VBM-step: $X^{(t+1)} = \langle BA^\top \rangle_{r(A,B|Z^{(t)})}$

3.1 Regularization Properties of VBEM

Here, we investigate the regularization properties of the above VBEM algorithm. Unlike other MAP estimation methods, the VBM-step is not explicitly given. We first show an analytic form of the VBM-step, and then elucidate the regularization mechanism of VBEM.

[4] A method to estimate the hyperparameters c_a^2 and c_b^2 has also been proposed.

Note that our analysis below can be regarded as an extension of the paper [15], which analyzes properties of reduced rank regression in *asymptotic* settings. In the current setting of matrix factorization, on the other hand, we need *non-asymptotic* analysis since only one observation matrix is available.

Analytic Solution of VBM-Step: Let

$$A = (\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_H) \quad \text{and} \quad B = (\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_H).$$

Below, we assume as in [12] that $\{\boldsymbol{a}_h\}_{h=1}^H$ and $\{\boldsymbol{b}_h\}_{h=1}^H$ are independent of each other, i.e., we restrict our function class used for approximating the posterior $p(A, B|Z)$ to

$$r(A, B) = \textstyle\prod_{h=1}^H r(\boldsymbol{a}_h) r(\boldsymbol{b}_h). \tag{11}$$

Then, we have a simpler update rule than Eqs.(8) and (9) as follows:

$$r(\boldsymbol{a}_h) \propto \phi(\boldsymbol{a}_h) \exp\left(\langle \log p(Z|A, B)\rangle_{r(A,B)/r(\boldsymbol{a}_h)}\right), \tag{12}$$

$$r(\boldsymbol{b}_h) \propto \phi(\boldsymbol{b}_h) \exp\left(\langle \log p(Z|A, B)\rangle_{r(A,B)/r(\boldsymbol{b}_h)}\right). \tag{13}$$

Substituting Eqs.(2) (with $Y = Z$ and $X = BA^\top$) and (10) into Eqs.(12) and (13), we can express the VB posterior as

$$r(A, B) = \textstyle\prod_{h=1}^H \mathcal{N}_M(\boldsymbol{a}_h; \boldsymbol{\mu}_{a_h}, \Sigma_{a_h}) \cdot \mathcal{N}_L(\boldsymbol{b}_h; \boldsymbol{\mu}_{b_h}, \Sigma_{b_h}), \tag{14}$$

where $\mathcal{N}_d(\cdot; \mu, \Sigma)$ denotes the density of the d-dimensional normal distribution with mean μ and covariance matrix Σ. Note that $\boldsymbol{\mu}_{a_h}$, $\boldsymbol{\mu}_{b_h}$, Σ_{a_h}, and Σ_{b_h} satisfy

$$\boldsymbol{\mu}_{a_h} = \sigma^{-2} \Sigma_{a_h} Z \boldsymbol{\mu}_{b_h}, \qquad \boldsymbol{\mu}_{b_h} = \sigma^{-2} \Sigma_{b_h} Z \boldsymbol{\mu}_{a_h}, \tag{15}$$

$$\Sigma_{a_h} = \sigma^2 \left[(\|\boldsymbol{\mu}_{b_h}\|^2 + \mathrm{tr}(\Sigma_{b_h})) + \sigma^2 c_a^{-2} \right]^{-1} I_M, \tag{16}$$

$$\Sigma_{b_h} = \sigma^2 \left[(\|\boldsymbol{\mu}_{a_h}\|^2 + \mathrm{tr}(\Sigma_{a_h})) + \sigma^2 c_b^{-2} \right]^{-1} I_L, \tag{17}$$

where I_d denotes the d-dimensional identity matrix. Solving the system of equations (15)–(17), we have the following theorem (its proof is omitted due to lack of space):

Theorem 1. *Let $\widehat{X}$ be the VB posterior mean of X, i.e., $\widehat{X} = \langle BA^\top \rangle_{r(A,B)}$. Let*

$$K = \max(L, M).$$

Let γ_h be the h-th largest singular value of Z and let $\boldsymbol{\omega}_{a_h}$ and $\boldsymbol{\omega}_{b_h}$ be the corresponding right and the left singular vectors. Then $\widehat{X}$ is analytically given by

$$\widehat{X} = \textstyle\sum_{h=1}^H \widehat{\gamma}_h \boldsymbol{\omega}_{b_h} \boldsymbol{\omega}_{a_h}^\top, \quad \textit{where} \quad \widehat{\gamma}_h = \max\left\{0, \left(1 - \frac{K\sigma^2}{\gamma_h^2}\right)\gamma_h - \Delta_h\right\}. \tag{18}$$

Δ_h (≥ 0) in the above is bounded as follows:

$$\Delta_h = \tfrac{\sigma^2}{c_a c_b} \qquad (L = M), \tag{19}$$

$$0 \leq \Delta_h \leq \tfrac{\sigma^2}{c_a c_b}\left(\sqrt{\tfrac{K}{H}} + \tfrac{K}{c_a c_b (K-H) H}\gamma_h\right) \qquad (L \neq M). \tag{20}$$

Furthermore, when $L = M$, the VB posterior is explicitly given by Eq.(14) *with*

$$\boldsymbol{\mu}_{a_h} = \sqrt{\tfrac{c_a}{c_b}\widehat{\gamma}_h}\boldsymbol{\omega}_{a_h}, \qquad \boldsymbol{\mu}_{b_h} = \sqrt{\tfrac{c_b}{c_a}\widehat{\gamma}_h}\boldsymbol{\omega}_{b_h}, \tag{21}$$

$$\Sigma_{a_h} = \tfrac{c_a}{2c_bK}\left\{\sqrt{\left(\widehat{\gamma}_h + \tfrac{\sigma^2}{c_ac_b}\right)^2 + 4\sigma^2K} - \left(\widehat{\gamma}_h + \tfrac{\sigma^2}{c_ac_b}\right)\right\} I_K, \tag{22}$$

$$\Sigma_{b_h} = \tfrac{c_b}{2c_aK}\left\{\sqrt{\left(\widehat{\gamma}_h + \tfrac{\sigma^2}{c_ac_b}\right)^2 + 4\sigma^2K} - \left(\widehat{\gamma}_h + \tfrac{\sigma^2}{c_ac_b}\right)\right\} I_K. \tag{23}$$

Regularization Mechanism of VBEM: From Theorem 1, we have the following interpretation.

If $c_ac_b \to \infty$, Δ_h vanishes and the VB estimator is expressed as

$$\widehat{X} = \textstyle\sum_{h=1}^{H} \max\left\{0, \left(1 - \frac{K\sigma^2}{\gamma_h^2}\right)\gamma_h\right\} \boldsymbol{\omega}_{b_h}\boldsymbol{\omega}_{a_h}^\top. \tag{24}$$

Thus, the *positive-part James-Stein (PJS) shrinkage operator* [13] is applied to singular values in a component-wise manner. The PJS estimator has a regularization effect that the estimation variance is reduced by shrinking the estimator (but the bias is increased in turn). It has been proved that the PJS estimator *dominates* the least-squares estimator under some conditions.

If $L = M$ and γ_h is large enough, the VB estimator is expressed as

$$\widehat{X} \approx \textstyle\sum_{h=1}^{H} \left(\gamma_h - \frac{\sigma^2}{c_ac_b}\right) \boldsymbol{\omega}_{b_h}\boldsymbol{\omega}_{a_h}^\top. \tag{25}$$

Thus, the singular value γ_h is decreased by a constant $\sigma^2/(c_ac_b)$. This may be regarded as a similar effect to the *trace-norm regularization* (i.e., the ℓ_1-norm regularization of singular values; see Section 2.2).

If γ_h is large enough and $\Delta_h \approx c\gamma_h$ $(0 \leq c \leq 1)$, the VB estimator is expressed as

$$\widehat{X} \approx \textstyle\sum_{h=1}^{H} (1-c)\gamma_h \boldsymbol{\omega}_{b_h}\boldsymbol{\omega}_{a_h}^\top. \tag{26}$$

Thus, the singular value γ_h is *shrunk* proportionally. This may be regarded as a similar effect to the *Frobenius-norm regularization* (i.e., the ℓ_2-norm regularization of singular values; see Section 2.2).

Thus, VBEM regularizes the solution based on the combination of PJS shrinkage, trace-norm regularization, and (possibly) Frobenius-norm regularization.

Posterior Mode, Posterior Mean, and Model Non-Identifiability: When the uniform prior (i.e., $c_a^2, c_b^2 \to \infty$) is used and the rank of $\widehat{X}$ is not reduced, one may intuitively think that no reguralization mechanism is involved. This intuition is true when MAP estimation is used—MAP estimation merely results in maximum likelihood (ML) estimation which has no regularization effect. However, in VBEM, the solution involves the PJS-type regularization (see Eq.(24)) and therefore overfitting can be avoided even when the uniform prior is used without rank constraints. This argument is in good agreement with the exprerimental results reported in the paper [12].

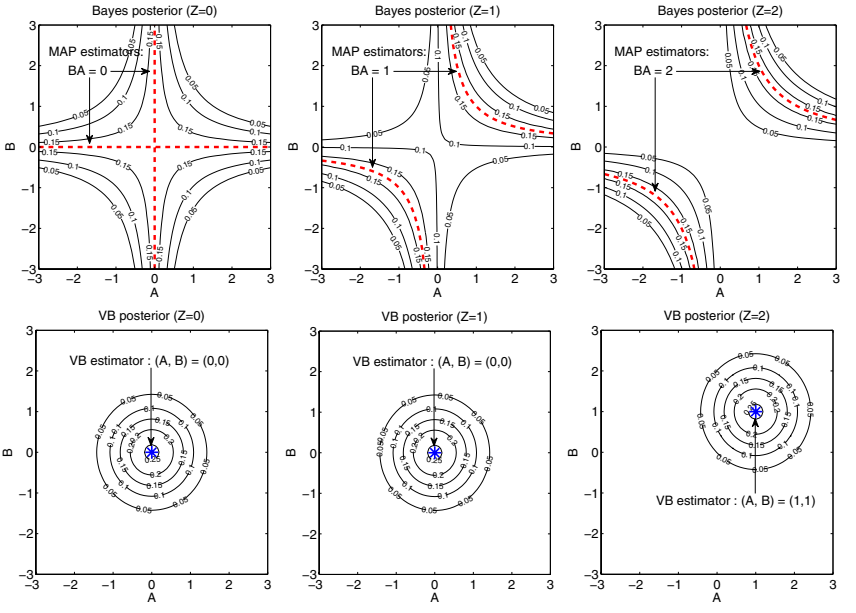


Fig. 1. MAP and VB solutions when $L = M = 1$ (the matrices X, A, and B are scalars)

Based on Theorem 1, we explain the reason for the significant difference between MAP and VB. For illustration purposes, let us start from the simplest case where $L = M = 1$ (i.e., X is a scalar) and the noise variance is $\sigma^2 = 1$. The top graphs in Fig.1 shows the contours of the Bayes posterior with the uniform prior on A and B when $Z = 0, 1, 2$ is observed (the horizontal and vertical axes correspond to A and B, respectively); the MAP estimators are indicated by the dashed curves (all points on the curves are the MAP estimators, which give the same solution $\widehat{X}$). In the bottom graphs of Fig.1, the VB posteriors—which are independent Gaussians—are plotted for $Z = 0, 1, 2$. The asterisks indicate their expectations, i.e., the VB estimators. When $Z = 0$, the MAP and the VB estimators both give the same value $\widehat{X} = \widehat{B}\widehat{A} = 0$. When $Z = 1$, the MAP estimator gives $\widehat{X} = 1$, while the VB estimator is still $\widehat{X} = 0$. When $Z = 2$, the VB estimator is off the origin ($\widehat{X} = 1$), but is still closer to the origin (i.e., strongly regularized) than the MAP solution $\widehat{X} = 2$. More generally, as the observed value Z is increased, the VB estimator approaches to the MAP estimator. However, the VB solution is always closer to the origin than the MAP solution. Note that $(-1, -1)$ is another VB solution when $Z = 2$, although only one VB solution at $(1, 1)$ is depicted in the figure for clear visibility.

The above analysis shows that even with the same uniform prior on A and B, MAP and VB give different solutions—the VB solution tends to be more strongly regularized than the MAP solution. We focused on $L = M = 1$ and $c_a^2, c_b^2 \to \infty$ in the above analysis for illustration purposes. But from Eqs.(18)–(20) we see that the amplitude of each component of the VB estimator is no larger than that of the PJS estimator (24) for any L, M, and $c_a^2, c_b^2 \geq 0$. This means that the VB solution always tends to be more strongly regularized than the MAP solution.

3.2 Generalization Properties of VBEM

Here, we investigate the generalization properties of the VBEM algorithm. First the generalization error of an estimated matrix is defined and it is decomposed into the 'necessary' part and the 'redundant' part. We then elucidate properties of the redundant part, in particular, sparseness of the solution and the generalization performance.

Generalization Error of VBEM: Our analysis is based on the assumption that the fully observed matrix Z is subject to the true distribution $p(Z|X^*)$, which is of the form (2) with the true matrix X^* and Z substituted for Y. Let H^* be the rank of X^* and assume

$$H^* \ll H.$$

This would be acceptable in, e.g., collaborative filtering since H is typically very large.

Let us measure the generalization error of the VB solution $\widehat{X}$ by the average Kullback-Leibler divergence from the true distribution to the estimated distribution:

$$G = \left\langle \log \frac{p(Z|X^*)}{p(Z|\widehat{X})} \right\rangle_{p(Z|X^*)} = \frac{1}{2\sigma^2} \left\langle \|\widehat{X} - X^*\|_{\mathrm{Fro}}^2 \right\rangle_{p(Z|X^*)}. \tag{27}$$

Let $\mathcal{W}_d(m, \Sigma, \Lambda)$ be the d-dimensional Wishart distribution with m degrees of freedom, scale matrix Σ, and non-centrality matrix Λ. Then, it is easy to show that $ZZ^\top$ follows the non-central Wishart distribution:

$$ZZ^\top \sim \mathcal{W}_H(K, \sigma^2 I_H, X^* X^{*\top}) \quad \text{if } L \leq M.$$

If $L > M$, we may simply re-define $X^\top$ as X so that $L \leq M$ holds.

By assumption, X^* consists of only H^* singular components. Let us decompose $\widehat{X}$ into the component projected onto the space spanned by X^* (the 'necessary' part) and its complement (the 'redundant' part):

$$\widehat{X} = \widehat{X}_{\mathrm{nec}} + \widehat{X}_{\mathrm{red}}.$$

Then, Eq.(27) implies that the generalization error can be decomposed as

$$G = G_{\mathrm{nec}} + G_{\mathrm{red}},$$

$$\text{where} \quad G_{\mathrm{nec}} = \left\langle \|\widehat{X}_{\mathrm{nec}} - X^*\|_{\mathrm{Fro}}^2 \right\rangle_{p(Z|X^*)} \quad \text{and} \quad G_{\mathrm{red}} = \left\langle \|\widehat{X}_{\mathrm{red}}\|_{\mathrm{Fro}}^2 \right\rangle_{p(Z|X^*)}.$$

Since $H^* \ll H$ by assumption, the contribution of the necessary components would be negligibly small compared with the contribution of the redundant components. Based on this reasoning, we focus on G_{red} in the following analysis.

Analysis of Eigenvalue Distribution of Redundant Components: Since the Gaussian noise is invariant under rotation, G_{red} can be expressed without loss of generality as

$$G_{\mathrm{red}} = \left\langle \|\widehat{X}(R)\|_{\mathrm{Fro}}^2 \right\rangle_{\mathcal{N}(R)}, \tag{28}$$

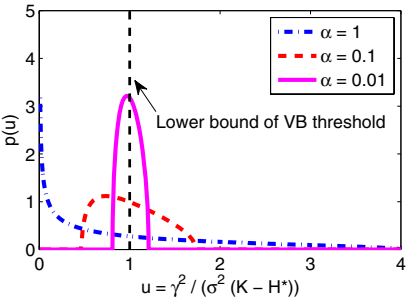


Fig. 2. Normalized eigenvalue (squared singular value of R) distribution of a large-scale Wishart matrix. Singular values smaller than the threshold are eliminated by the PJS operator in VBEM. We can show that the median of γ^2 is less than $\sigma^2(K-H^*)$ for any α. On the other hand, the VB threshold is no less than $\sigma^2 K$, which is always larger than the median. Thus, at least 50% of singular values are zero in VBEM (see Theorem 2 and Fig.3 for detail).

where $\widehat{X}(R)$ denotes the VB estimator given observation R, which is a $(H-H^*)\times(K-H^*)$ random matrix with entries independently subject to the normal distribution (denoted by $\mathcal{N}(R)$) with mean zero and variance σ^2. $RR^\top$ follows the central Wishart distribution:

$$RR^\top \sim \mathcal{W}_{H-H^*}(K-H^*, \sigma^2 I_{H-H^*}).$$

Let $u_1, u_2, \ldots, u_{H-H^*}$ be the eigenvalues of $\frac{1}{\sigma^2(K-H^*)}RR^\top$, and define the empirical distribution of the eigenvalues by

$$p(u) = \frac{\delta(u_1)+\delta(u_2)+\cdots+\delta(u_{H-H^*})}{H-H^*},$$

where $\delta(u)$ denotes the Dirac measure at u. Let α be the 'squareness' index of the target matrix X defined by

$$\alpha = \frac{H-H^*}{K-H^*},$$

which satisfies $0 < \alpha \leq 1$. Then, the following proposition is known regarding the distribution of eigenvalues of the central Wishart distribution.

Proposition 1 (Marcenko-Pastur law). *[16,17] In the large-scale limit where K, H, and H^* go to infinity in the same order, the probability measure of the empirical distribution of the eigenvalue u of $\frac{1}{\sigma^2(K-H^*)}RR^\top$ converges almost surely to*

$$p(u)du = \frac{\sqrt{(u-\underline{u})(\overline{u}-u)}}{2\pi\alpha u}\theta(\underline{u} < u < \overline{u})du, \tag{29}$$

where $\underline{u} = (\sqrt{\alpha}-1)^2$, $\overline{u} = (\sqrt{\alpha}+1)^2$, and $\theta(\cdot)$ denotes the indicator function.

Fig.2 depicts the eigenvalue distribution of a large-scale Wishart matrix for $\alpha = 0.01, 0.1, 1$, where the eigenvalues (or the squared singular values of R) are normalized by $\sigma^2(K-H^*)$ in the graph for better comparison.

Remember that the VB estimator $\widehat{X}(R)$ eliminates the singular values (of R) smaller than a certain positive value, which we call the *VB threshold*. When $c_a c_b \to \infty$, $\gamma^2 = K\sigma^2$ is the VB threshold (see Eq.(24)). Since $H^* \geq 0$, we have

$$\sigma^2 K \geq \sigma^2 (K - H^*),$$

which corresponds to a lower bound of the VB threshold for any $H^*, c_a^2, c_b^2 \geq 0$ (see Eq.(18)). In Fig.2, eigenvalues smaller than this threshold (which is normalized to one in the figure) are discarded.

Analysis of Sparseness of Redundant Components: We can evaluate the proportion of the singular values larger than the VB threshold as follows. Let

$$J_k(u_0) = 2\pi\alpha \int_{u_0}^{\overline{u}} u^k p(u) du. \tag{30}$$

Note that $(2\pi\alpha)^{-1} J_k(\underline{u})$ corresponds to the k-th moment of the Marcenko-Pastur distribution. $J_k(u_0)$ for $k = -1, 0, 1$ has analytic forms as follows:

Proposition 2. *[15] $J_k(u_0)$ has the following analytic forms for $k = -1, 0, 1$.*

$$J_{-1}(u_0) = \begin{cases} 2\sqrt{\alpha}\frac{\sqrt{1-s^2}}{2\sqrt{\alpha}s+1+\alpha} - \cos^{-1} s + \frac{1+\alpha}{1-\alpha}\cos^{-1}\frac{\sqrt{\alpha}(1+\alpha)s+2\alpha}{2\alpha s+\sqrt{\alpha}(1+\alpha)} & (0<\alpha<1), \\ 2\sqrt{\frac{1-s}{1+s}} - \cos^{-1} s & (\alpha = 1), \end{cases}$$

$$J_0(u_0) = -2\sqrt{\alpha}\sqrt{1-s^2} + (1+\alpha)\cos^{-1} s - (1-\alpha)\cos^{-1}\frac{\sqrt{\alpha}(1+\alpha)s+2\alpha}{2\alpha s+\sqrt{\alpha}(1+\alpha)},$$

$$J_1(u_0) = 2\alpha(-s\sqrt{1-s^2} + \cos^{-1} s), \quad \textit{where} \quad s = \frac{u_0-(1+\alpha)}{2\sqrt{\alpha}}.$$

This proposition enables us to calculate the proportion of nonzero redundant components as shown in the following theorem:

Theorem 2. *Let Θ be the proportion of nonzero redundant components in the large scale limit. Then, its upper bound is given by*

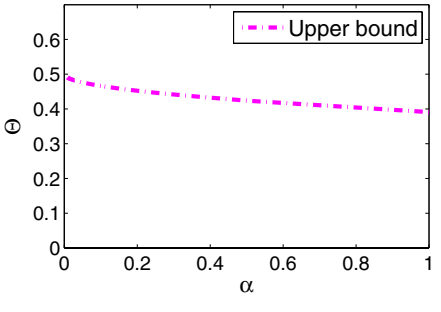


Fig. 3. The proportion Θ of non-zero singular values. Θ is always below 0.5 and it converges to 0.5 as $\alpha \to 0$, $c_a c_b \to \infty$ and $H^* = 0$.

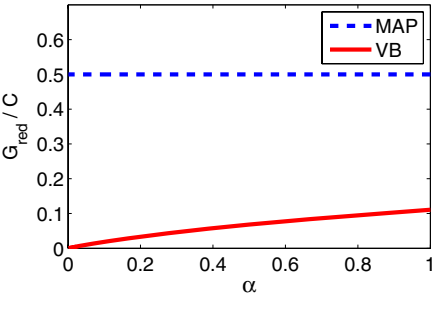


Fig. 4. The behaviors of $G_{\rm red}$ for MAP estimation and VB estimation, when $c_a c_b \to \infty$ and $H^* = 0$ (the values for VB becomes smaller when $H^* > 0$)

$$\Theta \leq (2\pi\alpha)^{-1} J_0(\kappa), \quad \textit{where} \quad \kappa = \tfrac{K}{K-H^*}. \tag{31}$$

The equality holds when $c_a c_b \to \infty$.

This theorem implies that VBEM gives a low-rank solution without explicit rank restriction. The curve in Fig.3 shows the value of $(2\pi\alpha)^{-1} J_0(1)$, which is the upper bound of Θ for any H^*, c_a^2, c_b^2. This value is always below 0.5, which means that at least 50% of singular values always become zero in VBEM; in practice the solution would be even more sparser.

Analysis of Redundant-Component Generalization Error: Next, we obtain the following theorem which enables us to evaluate the value of $G_{\rm red}$:

Theorem 3. *The upper bound of the contribution of the redundant components to the generalization error in the large-scale limit is given by*

$$G_{\rm red} \leq \tfrac{C\{J_1(\kappa)-2\kappa J_0(\kappa)+\kappa^2 J_{-1}(\kappa)\}}{4\pi\alpha}, \quad \textit{where} \quad C = (K-H^*)(H-H^*). \tag{32}$$

The equality holds when $c_a c_b \to \infty$.

Based on the above theorem and Proposition 2, we can compute the value of $G_{\rm red}$ analytically. In Fig.4, $G_{\rm red}/C$ for VBEM estimation and MAP estimation (which is equivalent to ML estimation due to the flat prior) are depicted. $G_{\rm red}/C$ for MAP is independent of α and is equal to 0.5. On the other hand, $G_{\rm red}/C$ for VBEM is increasing with respect to α, but is always much smaller than that of MAP. This implies that VBEM is highly robust against large observation noise.

4 Conclusions

In this paper, we have analyzed a variational Bayesian expectation-maximization (VBEM) method of matrix factorization. In particular, we elucidated the mechanism of inducing a low-rank solution and avoiding overfitting, where the principle of the positive-part James-Stein shrinkage operator and the Marcenko-Pastur law played important roles in the analysis.

Future work is to explicitly treat the missing values in the VBEM procedure, and to directly analyze the generalization error including the 'necessary' part (see Section 3.2).

References

1. Baldi, P.F., Hornik, K.: Learning in Linear Neural Networks: a Survey. IEEE Trans. on Neural Networks 6, 837–858 (1995)
2. Baldi, P., Brunak, S.: Bioinformatics. MIT Press, Cambridge (2001)
3. Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J.: Grouplens: applying collaborative filtering to usenet news. Commun. ACM 40, 77–87 (1997)
4. Funk, S.: Try this at home (2006), `http://sifter.org/~simon/journal/20061211.html`

5. Srebro, N., Jaakkola, T.: Weighted Low Rank Approximation. In: Proc. of ICML (2003)
6. Srebro, N., Rennie, J., Jaakkola, T.: Maximum Margin Matrix Factorization. In: Advances in NIPS, vol. 17 (2005)
7. Rennie, J.D.M., Srebro, N.: Fast Maximum Margin Matrix Factorization for Collaborative Prediction. In: Proc. of ICML (2005)
8. Salakhutdinov, R., Mnih, A.: Probabilistic Matrix Factorization. In: Advances in NIPS, vol. 20 (2008)
9. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Heidelberg (2006)
10. Attias, H.: Inferring Parameters and Structure of Latent Variable Models by Variational Bayes. In: Proc. of UAI (1999)
11. Lim, Y.J., Teh, T.W.: Variational Bayesian Approach to Movie Rating Prediction. In: Proc. of KDD Cup and Workshop (2007)
12. Raiko, T., Ilin, A., Karhunen, J.: Principal Component Analysis for Large Sale Problems with Lots of Missing Values. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS, vol. 4701, pp. 691–698. Springer, Heidelberg (2007)
13. James, W., Stein, C.: Estimation with Quadratic Loss. In: Proc. of the 4th Berkeley Symp. on Math. Stat. and Prob., pp. 361–379 (1961)
14. Watanabe, S.: Algebraic Analysis for Nonidentifiable Learning Machines. Neural Computation 13, 899–933 (2001)
15. Nakajima, S., Watanabe, S.: Variational Bayes Solution of Linear Neural Networks and its Generalization Performance. Neural Computation 19, 1112–1153 (2007)
16. Marcenko, V.A., Pastur, L.A.: Distribution of Eigenvalues for Some Sets of Random Matrices. Mathematics of the USSR-Sbornik 1, 457–483 (1967)
17. Wachter, K.W.: The Strong Limits of Random Matrix Spectra for Sample Matrices of Independent Elements. Annals of Probability 6, 1–18 (1978)

Variational Bayesian Approach for Long-Term Relevance Feedback

Sabri Boutemedjet and Djemel Ziou

Département d'informatique
Université de Sherbrooke, QC, Canada J1K 2R1
{sabri.boutemedjet,djemel.ziou}@usherbrooke.ca

Abstract. This paper presents a Bayesian approach to address two important issues of image recommendation that are: (1) change in long-term needs of users and (2) evolution of image collections. Users are offered a new interaction modality which allows them to provide either positive or negative relevance feedback (RF) data to express their recent needs. Then, an efficient variational Online learning algorithm updates both user and product collection models by favoring recent RF data. The proposed method is general and can be applied in collaborative filtering. Experimental results demonstrate the importance of maintaining most up-to-date user models on the rating's prediction accuracy.

1 Introduction

Most information filtering (IF) systems build user models to "recommend" new and relevant information to users according to their preferences in an overloaded store. For instance, Amazon.com offers a recommendation service to its customers to help them in discovering interesting products and improve their experience. Motivated from consumer psychology researches, the content-based image suggestion (CBIS) is a new type of IF systems which uses both visual (images) and contextual (e.g. user's location and time) information in the definition of user models [2,4,3]. A straightforward application of CBIS in Amazon.com could be, for instance, the personalized recommendation of new fashion clothes and accessories.

Product collections are dynamic and changing continuously. In fact, "new products" are added to inventories and thus made available to customers on a regular basis. On the other hand, users may change their appreciations on some products, for instance, relevant ones become irrelevant and vice versa. The change in user needs has been explained by consumer psychologists by "direct" and "derived" explanations [9]. Direct ones refer to psychological aspects that may be either intra-personal (e.g. need for novelty and information) and inter-personal (e.g. personal identity or affiliation to a group). "Derived" explanations are those motivations such as advertisements, changes in prices and incomes, and user's maturation. The recommender system should offer a new interaction modality which allows users to provide new relevance feedback data (judgements) on the suggested products *"to notify"* recent long-term needs. The new data serve

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 327–338, 2009.
© Springer-Verlag Berlin Heidelberg 2009

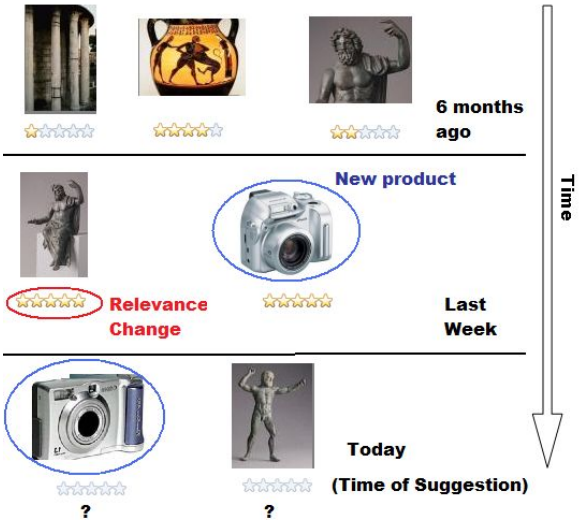


Fig. 1. Illustration of the complexity of updating the modeling of user needs

to track the change in users long-term needs. We have noticed that the majority of existing IF approaches [6,8,13,2] were designed mainly to model "past" data sets. They learn statistical models from the rating data collected from users in the past. Then, the learned models serve to predict "once" all the ratings of products to all users. This scenario has been widely employed as an experimental design to validate existing IF approaches. They have not considered the evolution of user needs over time [9]. To the best of our knowledge, the issue of using user feedbacks in recommender systems has not been addressed so far.

In this paper, we motivate an approach for taking into account "all" user feedbacks in order to maintain up-to-date user models that represent with most fidelity *"recent"* user needs. Even our approach is general in the sense that it can be applied to many IF approaches (either collaborative [6,8,13] or content-based filtering [10]), we focus here on CBIS for image recommendation [3,2]. In order to illustrate the complexity of the problem, let us consider the example outlined by Figure 1 which shows the evolution of user appreciations over time on some products of the collection. For instance, the camera has been "judged" recently (i.e. one week before suggestion) for the first time by the user as relevant. Therefore, it is natural to consider similar cameras in the "near" future as relevant for that user. However, the problem is not easy to handle for the piece of art on which we observe a high change (switch) in the user's appreciation. In such a case, consumer psychologists suggest to favor recent data comparatively with older ones in making future recommendations. Therefore, the rating of the piece of art should be predicted more likely high.

The principle long-term relevance feedback is "to refine" user models from the newly collected rating data. We formulate the problem probabilistically as

follows. Given an old statistical user model (i.e. some probability distribution) and recent rating data (user feedbacks), the goal is to adapt that model such that both old and recent rating data are considered in future image suggestions.

2 Our Method

2.1 CBIS Data

The domains of variables considered by CBIS are: a set of users $\mathcal{U} = \{1, \ldots, N_u\}$, an image collection $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_i}\}$, a rating scale $\mathcal{R} = \{1, \ldots, N_r\}$ and a set of contexts $\mathcal{E} = \{1, \ldots, N_e\}$. $\boldsymbol{x} = (x_1, \ldots, x_D)$ is a D-dimensional descriptor that represents the visual appearance of the image such as the shape and texture. Note that in collaborative filtering (CF) [5], $\boldsymbol{x}$ is considered one-dimensional variable with no information about the content. The history $\mathcal{D}^{ue}$ of each user u and context e contains data about past rating data on some images of the collection i.e. $\mathcal{D}^{ue} = \{< \boldsymbol{x}^{(i)}, r^{(i)} > | \boldsymbol{x}^{(i)} \in \mathcal{X}, r^{(i)} \in \mathcal{R}, i = 1, \ldots, |\mathcal{D}^{ue}|\}$.

2.2 A Bayesian CBIS Model

Before we define our approach for long-term RF, we need to set a model that fits the distribution of the set of user histories $\mathcal{D}^{ue}$. This model can be used to predict the ratings of unseen images. The Visual Content Context aware Flexible Mixture Model (VCC-FMM) [2] is graphical model that integrates both content $\boldsymbol{x}$ and contextual e information in modeling users. It uses the visual information (color, shape, or texture) as an added value comparatively with CF approaches. The main principle of VCC-FMM is that "similar" users prefer "similar" images. Therefore, both users and images are clustered into separate classes i.e. two hidden variables $z \in \{1, \ldots, K\}$ (user class or attitude), $c \in \{1, \ldots, M\}$ (image class) label each rating data $< \boldsymbol{x}, r > \in \mathcal{D}^{ue}$. In addition, since $\boldsymbol{x} \in \mathcal{X}$ is generally high-dimensional, VCC-FMM motivated feature selection to remove noisy and uninformative visual features and improve the quality of clustering. An additional hidden variable $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_D)$ (feature relevancy) labels each image $\boldsymbol{x}$ such that $\phi_l = 0$ $(l = 1, \ldots, D)$ corresponds to the case where the lth feature is irrelevant i.e. it follows a distribution independent of class-labels [4,7,3]. In other words, the class-conditional distribution of each feature is factorized as $p(x_l|c, \phi_l) = p(x_l|\theta_{cl})^{\phi_l} p(x_l|\xi_l)^{1-\phi_l}$ where ξ_l does not depend on the label c. By considering $p(\phi_l = 1) = \theta_l^{\phi}$ and after marginalization over hidden variables of labeled (completed) data (z, c, and $\boldsymbol{\phi}$), the likelihood $p(\mathcal{D}^{ue}|\Theta)$ under VCC-FMM is given by (1). The reader is referred to [2,3] for a comprehensive description of the conditional independence assumptions among random variables.

$$p(\mathcal{D}^{ue}|\Theta) = \prod_{i=1}^{|\mathcal{D}^{ue}|} p(\boldsymbol{x}^{(i)}, r^{(i)}|\Theta) = \sum_{z_{uei}=1}^{K} \sum_{c_{xi}=1}^{M} p(z_{uei}) p(c_{xi}) p(r^{(i)}|z_{uei}, c_{xi}) \times \prod_{l=1}^{D} [\theta_l^{\phi} p(x_l^{(i)}|\theta_{c_{xi}l}) + (1 - \theta_l^{\phi}) p(x_l^{(i)}|\xi_l)] \tag{1}$$

Let $\theta^Z_{ue} = (\theta^Z_{ue1}, \ldots, \theta^Z_{ueK})$, $\theta^C = (\theta^C_1, \ldots, \theta^C_M)$, $\theta^R_{zc} = (\theta^R_{zc1}, \ldots, \theta^R_{zcN_r})$, and θ^ϕ_l, denote the multinomial parameters of $p(z_{uei})$, $p(c_{xi})$, $p(r|z,c)$, and $p(\phi_l)$, respectively. $p(.|\xi_l)$ is a univariate distribution independent of class-labels and $p(.|\theta_{cl})$ is the class-conditional distribution of the lth feature. The authors in [4,3] adopted Beta distributions for $p(.|\theta_{cl})$ and $p(.|\xi_l)$. For convenience, we choose in this work Gaussian distributions $\mathcal{N}(\mu_{cl}, \tau_{cl}^{-1})$ and $\mathcal{N}(\mu_l, {\tau_l}^{-1})$ for $p(.|\theta_{cl})$ and $p(.|\xi_l)$, respectively. Θ is the set of VCC-FMM parameters and $h = (z, c, \boldsymbol{\phi})$ refers to the hidden label associated with a rating data $< \boldsymbol{x}, r > \in \mathcal{D}^{ue}$. The authors in [4,3] proposed to learn VCC-FMM by minimizing a message length (MML) objective using the Expectation-Maximization (EM) algorithm. The result is a point estimate Θ^* that conditions the rating of unseen images i.e. $p(\boldsymbol{x}, r|\Theta^*)$. However, since users are not always willing to "judge" (feedback) sufficient number of images, there is only a small amount of data that is insufficient to "refine" accurately VCC-FMM. In fact, the point estimate Θ^* is unbiased only in the limit of large amount of data. Note that VCC-FMM does not provide generative semantics at the level of users since θ^Z_{ue} are simply multinomial parameters.

We propose to overcome both issues by employing the Bayesian framework as a principled way for rating predictions. We consider all VCC-FMM parameters Θ as random variables with an initial uncertainty defined by a prior distribution $p(\Theta)$. Then, this uncertainty is updated from the data $\mathcal{D}^{ue}$ through a posterior distribution $p(\Theta|\mathcal{D}^{ue})$ which captures the generative semantics of VCC-FMM parameters. Now, the predictive distribution $p(\boldsymbol{x}, r|\mathcal{D}^{ue})$ is no longer dependent on a point estimate since all nuisance parameters Θ are integrated out weighted by their posterior distribution as

$$p(\boldsymbol{x}, r|\mathcal{D}^{ue}) = \int p(\Theta|\mathcal{D}^{ue}) p(\boldsymbol{x}, r|\Theta) d\Theta, \quad \text{with} \quad p(\Theta|\mathcal{D}^{ue}) = \frac{p(\Theta)p(\mathcal{D}^{ue}|\Theta)}{\int p(\Theta)p(\mathcal{D}^{ue}|\Theta)d\Theta} \quad (2)$$

The new model (2) is referred to as Bayesian VCC-FMM which formulates CBIS as a prediction problem.

We employ the exponential-conjugate framework in which both $p(\Theta)$ and $p(\Theta|\mathcal{D}^{ue})$ are conjugate (same functional form) to $p(\mathcal{D}^{ue}|\Theta)$. As we will show in subsection 2.5, this framework allows us to define accurate analytic approximations of $p(\boldsymbol{x}, r|\mathcal{D}^{ue})$ since neither $p(\Theta|\mathcal{D}^{ue})$ nor $\int p(\Theta)p(\mathcal{D}^{ue}|\Theta)d\Theta$, are tractable analytically. Variational methods lower-bound the marginal log-likelihood of all user histories $\log p(\mathcal{D}), \mathcal{D} = \cup_{ue}\mathcal{D}^{ue}$, by a tractable objective $\mathcal{F}$ in the exponential form [1]. These methods propose an approximate posterior distribution $Q(\Theta, \mathcal{H})$ over parameters Θ and hidden variables $\mathcal{H}$. Then, the lower bound is defined by the marginal log-likelihood minus the Kullback-Leibler (KL) divergence between the true posterior and the approximate distribution Q

$$\mathcal{F}(\mathcal{D}, Q) = E_Q[\log p(\mathcal{D}, \mathcal{H}, \Theta)] - E_Q[\log Q(\Theta, \mathcal{H})] \quad (3)$$

$\mathcal{D}$ denotes the data set of all users and all contexts. $\mathcal{H} = \{h_1, \ldots, h_{|\mathcal{D}|}\}$ such that $h_t = (z_t, c_t, \boldsymbol{\phi}_t), t = 1 \ldots, |\mathcal{D}|$, denotes the label associated with the tth rating data. The optimization of (3) with respect to Q is performed using an efficient Variational Bayesian EM (VBEM) algorithm [1]. The result is a distribution Q^* that best approximates the true posterior in terms of KL divergence.

Setting $p(\Theta)$ and $Q(\Theta)$. In the exponential-conjugate framework, we consider multinomial parameters θ^Z_{ue}, θ^C, θ^R_{zc}, and θ^ϕ_l as Dirichlet random variables with parameters α^Z, α^C, α^R, and α^{ϕ_l}, respectively. Also, both $p(\mu_{cl}, \tau_{cl})$ and $p(\mu_l, \tau_l)$ are Normal-Gamma distributions. For instance, $\mu_{cl} \sim \mathcal{N}(m_l, f\tau_{cl}^{-1})$ and $\tau_{cl} \sim \mathcal{G}(\omega, \nu)$ where m_l is set to be the mean of all the data. The parameters f, ν, ω are set to form broad and uninformative priors ($f = \nu = \omega \simeq 0$). The approximate posteriors $Q(\theta^Z|\gamma^Z_{ue})$, $Q(\theta^C|\gamma^C)$, $Q(\theta^R|\gamma^R_{zc})$, and $Q(\theta^\phi_l|\gamma^{\phi_l})$ are Dirichlet distributions. Also, $Q(\mu_{cl}, \tau_{cl}|m_{cl}, f_{cl}, \omega_{cl}, \nu_{cl})$ and $Q(\mu_l, \tau_l|m^\xi_l, f^\xi_l, \omega^\xi_l, \nu^\xi{}_l)$ are both Normal-Gamma distributions. Finally, the approximate posteriors of hidden variables are multinomial distributions i.e. $Q(z_{ue}|\beta^Z_{ue})$, $Q(c_x|\beta^C_x)$, and $Q(\phi_{xl}|\beta^{\phi_l}_x)$.

2.3 Long-Term Relevance Feedback

We define the long-term RF as the mechanism that takes into account recent feedbacks provided by users so that rating predictions are always made based on recent long-term user needs. Long-term RF data provided by user u in context e are also rating data defined as $\mathcal{D}\{s\}^{ue} = \{d_t =< \boldsymbol{x}^{(t)}, r^{(t)} >, t = 1, \ldots, s\}$ except that they are not used in the learning phase, and their temporal ordering (i.e. by t) is important. These data may be either positive (like) or negative (dislike) and constitute an important expression means of the preference change. A user may provide a negative feedback in order "to penalize" a suggested image and notify the CBIS system that he/she would not receive similar images in the future. One possible cause of such event is the satisfaction of the user's need in that content i.e. future consumption of similar images will not generate psychological stimuli. Also, the non-suitability of the context (e.g. busy user at work) may contribute to the generation of negative feedbacks. Positive feedbacks serve "to strengthen" a newly discovered information need (e.g. novel interesting images) that is not fully saturated. To maintain the user's trust over time, CBIS should "refine" user models to handle both positive and negative feedbacks.

Formally, at any time, the posterior distributions $Q(\Theta)$ of Bayesian VCC-FMM should represent recent user needs. Therefore, one possible implementation of long-term RF is to maximize the objective (3) from $\mathcal{D} \cup \mathcal{D}\{s\}^{ue}$ in order to "relearn" a new model parameterized by a new distribution Q^*. However, this solution is impractical due to its prohibitive cost in terms of storage (old history) and computing (learning) as the numbers of users/images are generally high. According to the Bayesian VCC-FMM presented previously, we identify three Q distributions $Q(z_{ue}|\beta^Z_{ue})$, $Q(\theta^Z|\gamma^Z_{ue})$ and $Q(\theta^R|\gamma^R_{zc})$ affected by $\mathcal{D}\{s\}^{ue}$. Therefore, we propose to implement more efficiently the long-term RF as the process that "updates" those distributions. The input data is a model Q (of old user needs) and $\mathcal{D}\{s\}^{ue}$. The output of long-term RF is a new model $\tilde{Q}$ that represents both old and new user long-term needs. The Bayesian framework offers a principled way to update the posterior distributions from new data, for instance, by updating the sufficient statistics of variational parameters using Bayes rule. However, we argue in accordance with psychologists [9], that the straightforward maximization of the variational lower bound (3) w.r.t Q is not

suitable for long-term RF due to the following facts. First, users with "rare" feedbacks will see their models updated slowly comparatively with other users with more RF data. Second, (3) considers all RF data in $\mathcal{D}\{s\}^{ue}$ with equal importance which does not favor recent user needs and may not capture the switch in appreciations. We propose rather the maximization of a *discounted expected* lower bound corresponding to a *"fixed"* amount of T *"weighted"* RF data. The tth RF data $d_t \in \mathcal{D}\{s\}^{ue}$ is weighted by a time-dependent forgetting factor $\Delta(t)$. The following weighting $\Delta(t) = \prod_{e=t+1}^{s} \delta(e), \delta(e) \in [0,1]$ which penalizes older feedbacks fits well our goal [12]. The *"discounted"* lower bound of the expected log-likelihood is

$$\begin{aligned}\mathcal{L}^{\Delta}(\mathcal{D}\{s\}^{ue}, Q) &= \int Q(\Theta) \log \frac{p(\Theta)}{Q(\Theta)} d\Theta \\ &+ T\eta(s) \sum_{t=1}^{s} \Delta(t) \sum_{h_t} \int Q(\Theta) Q(h_t) \log \frac{p(d_t, h_t|\Theta)}{Q(h_t)} d\Theta\end{aligned} \tag{4}$$

where $\eta(s) = (\sum_{t=1}^{s} \Delta(t))^{-1}$ is a normalizing constant. It can be shown that the following mathematical property holds: $\eta(s)\delta(s) = \eta(s-1)\big(1-\eta(s)\big)$.

Now, the new user model $\tilde{Q}$ can be identified by maximizing (4) w.r.t three user-specific Q distributions from each recent RF data $d_t \in \mathcal{D}\{s\}^{ue}$ taken individually and after d_{t-1}. We present in the following, an Online version of VBEM algorithm which identifies $\tilde{Q}$ only from $d_s = \{(x^{(s)}, r^{(s)})\} \in \mathcal{D}\{s\}^{ue}$ given that we have already updated Bayesian VCC-FMM from $\mathcal{D}\{s\}^{ue} - \{d_s\}$. First, in the VBE-step, we maximize $\mathcal{L}^{\Delta}(\mathcal{D}\{s\}^{ue}, Q)$ w.r.t $Q(h_s)$ while $\tilde{Q}(\Theta)$ is set to $Q(\Theta)$ in order to make the discounted lower bound as tight as possible to the true expected marginal. After that, in the VBM-step, $\tilde{Q}(\Theta)$ is obtained by maximizing (4) given $\tilde{Q}(h_s)$. The posteriors of hidden variables are computed in the VBE-step as

$$\tilde{\beta}^{Z}_{uez} \propto \exp\big\{e(\gamma^{Z}_{uez}) + \sum_{c=1}^{M} \beta^{C}_{x^{(s)}c} e(\gamma^{R}_{zcr^{(s)}})\big\} \tag{5}$$

$$\tilde{\beta}^{C}_{x^{(s)}c} \propto \exp\big\{e(\gamma^{C}_{c}) + \sum_{z=1}^{K} \beta^{Z}_{uez} e(\gamma^{R}_{zcr^{(s)}}) + \sum_{l=1}^{D} \beta^{\phi_l}_{x^{(s)}} E_Q[\log p(x^{(s)}_l|\theta_{cl})]\big\} \tag{6}$$

$$\beta^{\phi_l}_{x^{(s)}} \propto \exp\big\{e(\gamma^{\phi_l}_{1}) + \sum_{c=1}^{M} \beta^{C}_{x^{(s)}c} E_Q[\log p(x^{(s)}_l|\theta_{cl})]\big\} \tag{7}$$

To simplify the presentation, we have set $e(a_{nm}) = \Psi(a_{nm}) - \Psi(|a_n|)$ such that $|a_n| = \sum_{m'} a_{nm'}$ and $a_n = (a_{n1}, \ldots, a_{nA})$ is an arbitrary vector. Also, we have $E_Q[\log p(x_l|\theta_{cl})] = \text{const} + \frac{1}{2}[\Psi(\nu_{cl}) - \log \omega_{cl} - (\nu_{cl})/(\omega_{cl})((x_l - m_{cl})^2 + f_{cl}^{-1})]$.

We show how to derive the recursive update formulas in the VBM-step of $\tilde{Q}(\Theta)$ from d_s. We use the superscript (t) on the variational parameters to denote their values after an update from $d_t, t < s$.

$$\log \tilde{Q}(\theta^{Z}|\tilde{\gamma}^{Z}_{ue}) = \text{const} + \sum_{z=1}^{K} (\alpha^{Z}_{z} - 1) \log \theta^{Z}_{z} + T\eta(s) \sum_{t=1}^{s} \Delta(t) \sum_{z=1}^{K} \beta^{Z(t)}_{uez} \log \theta^{Z}_{z} \tag{8}$$

We set $A(s) = \eta(s) \sum_{t=1}^{s} \Delta(t) \sum_{z=1}^{K} \beta_{uez}^{Z(t)} \log \theta_z^Z$, which can be rewritten as

$$\begin{aligned} A(s) &= \eta(s) \Bigg[\sum_{t=1}^{s-1} \prod_{a=t+1}^{s} \delta(a) \sum_{z=1}^{K} \beta_{uez}^{Z(t)} \log \theta_z^Z + \sum_{z=1}^{K} \tilde{\beta}_{uez}^{Z} \log \theta_z^Z \Bigg] \\ &= \eta(s)\delta(s) \sum_{t=1}^{s-1} \prod_{a=t+1}^{s-1} \delta(a) \sum_{z=1}^{K} \beta_{uez}^{Z(t)} \log \theta_z^Z + \eta(s) \sum_{z=1}^{K} \tilde{\beta}_{uez}^{Z} \log \theta_z^Z \qquad (9) \\ &= \big[1 - \eta(s)\big] A(s-1) + \eta(s) \sum_{z=1}^{K} \tilde{\beta}_{uez}^{Z} \log \theta_z^Z \end{aligned}$$

From (8) and (9), and since $\log Q(\theta^Z | \gamma_{ue}^Z) = \text{const} + \sum_{z=1}^{K} (\alpha_z^Z - 1) \log \theta_z^Z + TA(s-1)$ (value after update from d_{s-1}), then $\log \tilde{Q}(\theta^Z | \tilde{\gamma}_{ue}^Z)$ is expressed as the logarithm of a Dirichlet distribution parameterized by

$$\tilde{\gamma}_{uez}^Z = (1 - \eta(s)) \gamma_{uez}^Z + \eta(s) \big[\alpha_z^Z + T \tilde{\beta}_{uez}^Z \big] \qquad (10)$$

Similarly, we get the recursive update formula for $\tilde{\gamma}_{zcr}^R$, α^Z, and α^R

$$\tilde{\gamma}_{zcr^{(s)}}^R = (1 - \eta(s)) \gamma_{zcr^{(s)}}^R + \eta(s) \big[\alpha_{r^{(s)}}^R + T \tilde{\beta}_{x^{(s)}c}^C \tilde{\beta}_{uez}^Z \big] \qquad (11)$$

$$\tilde{\alpha}_z^Z = \Psi^{-1} \Big(\Psi(\alpha_z^Z) + (N_u N_e)^{-1} \big[e(\tilde{\gamma}_{uez}^Z) - e(\gamma_{uez}^Z) \big] \Big) \qquad (12)$$

$$\tilde{\alpha}_{r^{(s)}}^R = \Psi^{-1} \Big(\Psi\Big(\sum_{r'=1}^{N_r} \alpha_{r'}^R \Big) + (MK)^{-1} \sum_{z,c} e(\tilde{\gamma}_{zcr^{(s)}}^R) \Big) \qquad (13)$$

$\Psi(x) = d \log \Gamma(x)$ is the digamma function and Ψ^{-1} is its inverse which can be computed in a fourteen digit accuracy in five iterations [11]. Note that one could also update all Q distributions from $\mathcal{D}\{s\}^{ue}$. Empirically, this method does not provide significant improvement of the modeling accuracy.

2.4 New Image Data

One known weakness of CF approaches is their inability to recommend unrated (i.e. novel) products. On the other hand, VCC-FMM can use the content of those images with an average relative decrease in accuracy about only 25% for 30% of novel images [4]. This decrease in performance is caused by errors in labeling some of these images that *do not fall* exactly into one of the existing image clusters. In practice, with an increasing number of "new images", the clustering structure underlying the collection changes considerably (e.g. shape and density of clusters) which may affect the suggestion accuracy. In addition, the early inaccurate estimators obtained using few image data can be improved from new images in order to form well-identified clusters by penalizing (i.e. discount) old image data. Therefore, we advocate to "readjust" also image clusters from the content of new images using the same methodology employed for long-term RF but with different Q distributions. Indeed, the clustering structure underlying the image collection is defined by $Q(.|\alpha^C)$, $Q(.|\alpha^{\phi_l})$, $Q(.|\gamma^C)$, $Q(.|\gamma^{\phi_l})$ and

Normal-Gamma distributions presented in Subsection 2.2. Since each new image $\boldsymbol{x}$ is unrated, we propose to compute its posterior distribution $Q(c_x|\beta_x^C)$ in the VBE-step from the visual content only. The impact of this assumption on the prediction accuracy is evaluated in experiments. Let us consider a set $\mathcal{X}\{s\}$ of s new images ordered by their arrival time. We update the clustering structure from the last image $\boldsymbol{x} \in \mathcal{X}\{s\}$ as follows. In VBE-step, we compute the posteriors of hidden variables c and ϕ

$$\tilde{\beta}_{xc}^C \propto \exp\left\{e(\gamma_c^C) + \sum_l \beta_x^{\phi_l} E_Q[\log p(x_l|\theta_{cl})]\right\}, \tilde{\beta}_x^{\phi_l} \propto \exp\left\{e(\gamma_1^{\phi_l}) + \sum_c \beta_{xc}^C E_Q[\log p(x_l|\theta_{cl})]\right\}$$

Following long-term RF, one could obtain the updates in VBM-step as

$$\tilde{f}_{cl}\tilde{m}_{cl} = [1-\eta(s)]f_{cl}m_{cl} + \eta(s)\big[mf + T\tilde{\beta}_{xc}^C\tilde{\beta}_x^{\phi_l}x_l\big]$$

$$\tilde{f}_{cl} = (1-\eta(s))f_{cl} + \eta(s)\big[f + T\tilde{\beta}_{xc}^C\tilde{\beta}_x^{\phi_l}\big], \tilde{\alpha}_c^C = \Psi^{-1}\big(\Psi(|\alpha^C|) + e(\gamma_c^C)\big)$$

$$\tilde{\nu}_{cl} = (1-\eta(s))\nu_{cl} + \eta(s)\big(\nu + (T/2)\tilde{\beta}_{xc}^C\tilde{\beta}_x^{\phi_l}\big), \tilde{\gamma}_1^{\phi_l} = (1-\eta(s))\gamma_1^{\phi_l} + \eta(s)[\alpha^\phi + T\tilde{\beta}_x^{\phi_l}]$$

$$\tilde{\omega}_{cl} = (1-\eta(s))\big(\omega_{cl} + \frac{f_{cl}m_{cl}^2}{2}\big) + \eta(s)\big(\omega + \frac{fm^2 + T\tilde{\beta}_{xc}^C\tilde{\beta}_x^{\phi_l}x_l^2}{2}\big) - \frac{\tilde{f}_{cl}\tilde{m}_{cl}^2}{2}$$

$Q(\mu_l, \tau_l|m_l^\xi, f_l^\xi, \omega_l^\xi, \nu_l^\xi)$ can be derived straightforwardly. Note that with an increasing numbers of new images and long-term RF data, the model order defined by M and K may change (e.g. new classes may be added). Our Bayesian formulation of CBIS allows us to identify the optimal model order by monitoring the behavior of the lower bound of the likelihood as a model selection criterion.

2.5 The Predictive Distribution

After training, we approximate $p(\Theta|\mathcal{D}^{ue})$ in (2) by $Q(\Theta)$ i.e. $p(\boldsymbol{x}, r|\mathcal{D}^{ue}) \simeq \int Q(\Theta)p(\boldsymbol{x}, r|\Theta)d\Theta$. The predictive distribution (2) can indeed be calculated analytically as

$$\begin{aligned} p(\boldsymbol{x}, r|\mathcal{D}^{ue}) = &\sum_{z=1}^{K}\sum_{c=1}^{M}\frac{\gamma_c^C\gamma_{uez}^Z\gamma_{zcr}^R}{|\gamma_{ue}^Z||\gamma^C||\gamma_{zc}^R|}\prod_{l=1}^{D}|\gamma^{\phi_l}|^{-1} \\ &\left(\gamma_1^{\phi_l}\mathrm{St}\big(x_l|m_{cl}, \frac{\nu_{cl}f_{cl}}{(f_{cl}+1)\omega_{cl}}, 2\nu_{cl}\big) + \gamma_2^{\phi_l}\mathrm{St}\big(x_l|m_l^\xi, \frac{\nu_l^\xi f_l^\xi}{(f_l^\xi+1)\omega_l^\xi}, 2\nu_l^\xi\big)\right) \end{aligned} \tag{14}$$

where $\mathrm{St}(.|a, b, c)$ denotes a univariate Student's t-distribution with mean a, precision b, and c degrees of freedom. The rating is predicted using the max rule as $\hat{r}_{\boldsymbol{x}}^{ue} = \arg\max p(r|\boldsymbol{x}, \mathcal{D}^{ue})$ with $p(r|\boldsymbol{x}, \mathcal{D}^{ue}) = p(r, \boldsymbol{x}|\mathcal{D}^{ue})/\sum_r p(r, \boldsymbol{x}|\mathcal{D}^{ue})$.

3 Experiments

3.1 Goal and Data Sets

In this section, we evaluate our long-term RF on the real CBIS data set used in [4,3]. This data set $\mathcal{D}$ contains 13446 rating data collected from 27 subjects

during a period of three months. Subjects expressed the relevance of images on a five star rating scale. The context information is defined as a combination of location $\mathcal{L} = \{in-campus, out-campus\}$ and time as $\mathcal{T} = (weekday, weekend)$. Moreover, for each rating data is associated a variable indicating the day of its acquisition. The image collection contains 4775 natural and man-made images (41 classes). Image data is represented using the color correlogram for image texture representation and the edge orientation histogram for shape. Therefore, a visual feature vector is represented in a 40-dimensional space.

We evaluate two evolution settings of Bayesian VCC-FMM. In the first one, the model is updated from only from long-term RF data provided by users. The second setting deals with an image collection growing over time whose model is updated from new images. We refer by "Updated-VCC-FMM" to our proposed approach for both settings. For the forgetting factor $\delta(t)$, many choices can be adopted depending on the relative importance attributed to the new data comparatively with older ones. Moreover, one could consider user-specific forgetting factors (i.e. $\delta_u(t)$) to distinguish those users with highly changing long-term needs from the others. In our experiment, we have defined two variants of "Updated-VCC-FMM" depending on the weighting factor used: $\delta^{(1)}(t) = 1$ (no discount) and $\delta^{(2)}(t) = 1-(100+0.05(t-2))^{-1}$ with $\delta^2(1) = 0.5$ (with discount).

3.2 Design of Experiments and Evaluation Metrics

Due to dependence of long-term RF data on time, we propose a new experimental design to validate the proposed method based on the collected data set $\mathcal{D}$ as follows. First, we sort all rating data (i.e. $< u, e, \boldsymbol{x}, r >$) by their arrival time (in days). For convenience, we introduced the user and context information to each rating data. Let $\mathcal{D}^t$ be the set of ratings provided by users at time t. We denote by $\mathcal{X}^t$ the set of rated images in $\mathcal{D}^t$. We learn Bayesian VCC-FMM from rating data provided during the first 30 days $\mathcal{D}_{train} = \bigcup_{t=1}^{30} \mathcal{D}^t$ to define initial user models. Finally, we measure the rating prediction accuracy in terms of the mean absolute error (MAE) which is the average of the absolute deviation between the recent ratings and the predicted ones. We investigate the importance of long-term RF over time, by updating Bayesian VCC-FMM using $\mathcal{D}^t$ and then computing MAE on $\mathcal{D}^{t+1}$ as

$$MAE_{t+1} = \frac{1}{|\mathcal{D}^{t+1}|} \sum_{i=t_1}^{t_2} |r^{(i)} - \hat{r}^{u^{(i)}e^{(i)}}_{\boldsymbol{x}^{(i)}}| \tag{15}$$

where $t_1 = |\cup_{s=1}^{t} \mathcal{D}^s| + 1$, $t_2 = t_1 + |\mathcal{D}^{t+1}|$, and $\hat{r}^{u^{(i)}e^{(i)}}_{\boldsymbol{x}^{(i)}}$ is the predicted rating for the ith observation (see subsection 2.5). Let $\mathcal{X}^t_{new}$ be the set of "newly" discovered images at time t, those images that have not been rated beforehand. We have $\mathcal{X}^t_{new} = \{\boldsymbol{x} | \boldsymbol{x} \in \mathcal{X}^t, \nexists t' < t : \boldsymbol{x} \in \mathcal{X}^{t'}\}$ and their corresponding ground truth rating data $\mathcal{D}^t_{new} = \{< u, e, \boldsymbol{x}, r > \in \mathcal{D}^t | \boldsymbol{x} \in \mathcal{X}^t_{new}\}$. The importance of using new image data (subsection 2.4) is investigated by first updating Bayesian VCC-FMM from the set of new images $\mathcal{X}^t_{new}$ and computing MAE on $\mathcal{D}^t_{new}$. We have found that only 45% of images were rated during the first 30 days.

3.3 Results

Figure 2 shows the average MAE over successive time intervals (5 days). It is reported that long-term RF data is an important indicator about the evolution of user needs over time since both Updated-VCC-FMM algorithms outperform the non updated variant of the model (at least: 7.74% in average, 15.4% at day 55). When recent RF data is used, Bayesian VCC-FMM predicts more accurately future ratings. It is also reported that the ideal way to use such data, is to "forget" slightly old RFs by attributing more importance to recent ones. For instance, the variant *"Updated-with-discount"* improves after 55 days both its initial MAE_5 ("day 5") and that provided by *"Updated-no-discount"* by 5.65% and 10.81%, respectively. The first improvement can be explained by the availability of more data to refine user models.

From Figure 3, it is reported that updating the model from new image data is also an important process that ensures a better prediction of ratings. This fact is due to the improvement of the quality of the image collection modeling (i.e. well-identified image clusters). Moreover, by attributing more importance to new images, the clustering structure is better uncovered since earlier inaccurate estimators are penalized. However, MAE increases (5.7%) after 680 images due to the unavailability of user judgements on these new images that can be used to fine tune the model. Indeed, recent images inherit the relevance of older ones which does correspond necessarily to any potential change in user needs.

We have also analyzed the behavior of some existing CF approaches: the Aspect model [6], Flexible Mixture Model (FMM) [13], and User Rating Profile (URP) [8]. All these models are learned on $\mathcal{D}_{train}$ (first 30 days). Since these CF approaches are unable to predict the ratings for "new images", we compute the error MAE_{eval} only on rated ones. In other words, the evaluation data is formed as $\mathcal{D}_{eval} = \cup_{t>31}\mathcal{D}^t$ such that it contains only images that have been rated in $\mathcal{D}_{train}$. Note that this experimental setting is different from the commonly used n-fold cross validation which does not consider arrival time of rating data.

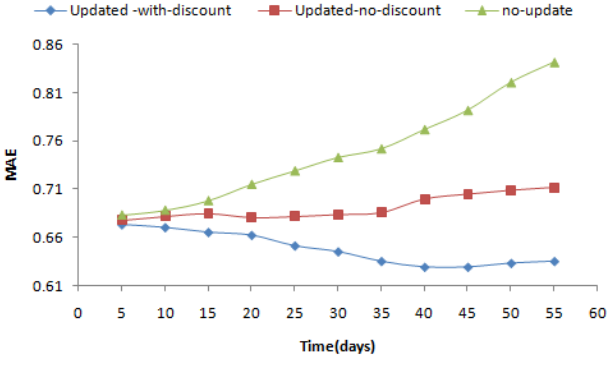


Fig. 2. The behavior of MAE over time reported by Bayesian VCC-FMM variants with/without long-term RF

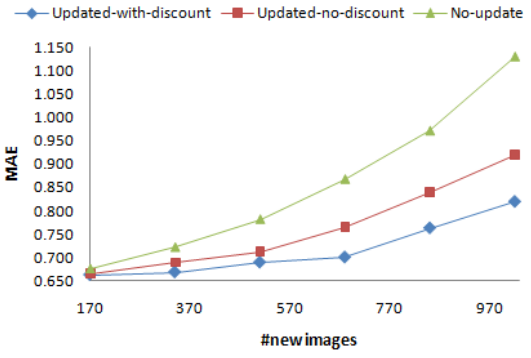


Fig. 3. The behavior of MAE for Bayesian VCC-FMM variants with/without update of the image collection model

Table 1. Performance of approaches on the evaluation data set. MAE_a refers the MAE computed on the data set collected during the day $30+a$. $Loss_a = 100(MAE_a - MAE_5)/MAE_5$, denotes the relative increase in MAE_a comparatively with MAE_5.

	Non-updated approaches				**Updated-VCC-FMM**	
	Aspect	FMM	URP	No-update	no-discount	with-discount
MAE_5(baseline)	1.083	1.054	1.013	0.683	0.678	0.673
MAE_{eval}	1.230	1.188	1.142	0.749	0.691	0.648
MAE_{55}	1.349	1.270	1.216	0.842	0.712	0.635
$Loss_{eval}(\%)$	13.86	12.69	12.73	9.61	1.96	−3.73
$Loss_{55}(\%)$	24.91	20.49	20.04	23.28	5.02	−5.65

Table 1 shows that average MAE_{eval} during a period of 55 days after the learning, increases for all approaches (except for Updated-VCC-FMM with discount) comparatively with their baseline MAE_5. Moreover, for these approaches, we observe a significant increase of the error (at least 20%) in the most recent rating prediction (i.e. MAE_{55}) comparatively with MAE_5. After a long period of time, initial user models are no longer suitable to predict current needs of users. Note that, the performance of VCC-FMM no-update comparatively with CF is explained by the added value of both visual and contextual information, as argued in [4,3].

4 Conclusions

In this paper, we have motivated the problem of tracking both changes of user long-term needs and product collections over time. We formulated the problem using a Bayesian model, learned continuously and efficiently by penalizing old data comparatively with recent ones. The proposed method has been applied successfully in content-based image suggestion. Experiments showed a significant

improvement of the prediction accuracy. This method is general and can be extended straightforwardly to many statistical collaborative filtering models.

Acknowledgements

The completion of this research was made possible thanks to Natural Sciences and Engineering Research Council of Canada (NSERC), Bell Canada's support through its Bell University Laboratories R&D program.

References

1. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Heidelberg (2006)
2. Boutemedjet, S., Ziou, D.: A Graphical Model for Context-Aware Visual Content Recommendation. IEEE Trans. on Multimedia 10(1), 52–62 (2008)
3. Boutemedjet, S., Ziou, D., Bouguila, N.: A Graphical Model for Content Based Image Suggestion and Feature Selection. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS (LNAI), vol. 4702, pp. 30–41. Springer, Heidelberg (2007)
4. Boutemedjet, S., Ziou, D., Bouguila, N.: Unsupervised Feature Selection for Accurate Recommendation of High-Dimensional Image Data. In: Proc. of 21st Conf. on Advances in Neural Information Processing Systems (NIPS) (2007)
5. Breese, J.S., Heckerman, D., Kadie, C.: Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In: Proc. of 14th Conf. on Uncertainty in Artificial Intelligence (UAI), pp. 43–52 (1998)
6. Hofmann, T.: Latent Semantic Models for Collaborative Filtering. ACM Trans. on Information Systems 22(1), 89–115 (2004)
7. Law, M.H.C., Figueiredo, M.A.T., Jain, A.K.: Simultaneous Feature Selection and Clustering Using Mixture Models. IEEE Trans. on Patt. Anal. and Mach. Intell. 26(9), 1154–1166 (2004)
8. Marlin, B.: Modeling User Rating Profiles For Collaborative Filtering. In: Proc. of 17th Conf. on Advances in Neural Information Processing Systems (NIPS) (2003)
9. McAlister, L., Pessemier, E.: Variety Seeking Behavior: An Interdisciplinary Review. The Journal of Consumer Research 9(3), 311–322 (1982)
10. Melville, P., Mooney, R.J., Nagarajan, R.: Content-Boosted Collaborative Filtering for Improved Recommendations. In: Proc. of the 18th Nat. Conf. on Artificial Intelligence (AAAI), pp. 187–192 (2002)
11. Minka, T.: Estimating a Dirichlet Distribution (2003)
12. Sato, M.A.: Online Model Selection Based on the Variational Bayes. Neural Computation 13(7), 1649–1681 (2001)
13. Si, L., Jin, R.: Flexible Mixture Model for Collaborative Filtering. In: Proc. of 20th Int. Conf. on Machine Learning (ICML), pp. 704–711 (2003)

Detecting Link Hijacking by Web Spammers

Young-joo Chung, Masashi Toyoda, and Masaru Kitsuregawa

Institute of Industrial Science, The University of Tokyo,
4-6-1 Komaba Meguro-ku, Tokyo, Japan
{chung,toyoda,kitsure}@tkl.iis.u-tokyo.ac.jp

Abstract. Since current search engines employ link-based ranking algorithms as an important tool to decide a ranking of sites, Web spammers are making a significant effort to manipulate the link structure of the Web, so called, link spamming. Link hijacking is an indispensable technique for link spamming to bring ranking scores from normal sites to target spam sites. In this paper, we propose a link analysis technique for finding link hijacked sites using modified PageRank algorithms. We performed experiments on the large scale Japanese Web archive and evaluated the accuracy of our method. Detection precision of our approach was improved about 25% from a naive approach.

Keywords: Link analysis, Web spam, Information retrieval, Link hijacking.

1 Introduction

In the last decade, search engines have been essential tools for information retrieval. People rely heavily on these tools to find information on the Web, and as a result, most Web sites get a considerable number of visitors via search engines. Since the increase in visitors usually means the increase in financial profit, and approximately 50% of search engine users look at no more than the first 5 results in the list [1], obtaining a high ranking in the search results became crucial for the success of sites.

Web spamming is the behavior that manipulates Web page features to get a higher ranking than the page deserves. Web spamming techniques can be categorized into term spamming and link spamming. *Term spamming* manipulates textual contents of pages by repeating specific keywords that are not related with page contents and by adding irrelevant meta-keywords or anchor text. Search engines which employ textual relevance to rank pages will return these manipulated pages at the top of the result list. *Link spamming* manipulates the link structure of the Web to mislead link-based ranking algorithms such as PageRank [4]. Link-based ranking algorithms consider a link as an endorsement for pages. Therefore, spammers create numerous false links and construct an artificially interlinked link structure, so called a spam farm, where all pages link to the target spam page in order to centralize its link-based importance.

Links from external normal pages to spam pages are needed in order to attract the attention of search engines and feed ranking scores to spam farms. These

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 339–350, 2009.
© Springer-Verlag Berlin Heidelberg 2009

links that are created without any agreements of page owners are called *hijacked link*. To hijack link, spammers post comments including URLs to spam pages on public bulletin boards, buy expired domains and sponsor pages. Hijacked links affect link-based ranking algorithms significantly, when they are pointing to spam farms containing a large amount of spam pages.

In this paper, we propose a novel method for detecting Web sites hijacked by spammers. Most of previous research has focused on demoting or detecting spam, and as far as we know, there has been no study on detecting link hijacking that is important in the following situations:

- Hijacked sites are prone to be attacked continuously by various spammers (e.g. by repetitive spam comments on blogs). Observing such sites will be helpful for the prompt detection of newly created spam sites that might not be filtered by existing anti-spam techniques. Since spam detection has been an arms race, it is important to find out sites with new spamming methods.
- Once we detect hijacked sites, we can modify link-based ranking algorithms to reduce the importance of newly created links from hijacked pages in those sites. It makes the algorithms robust to newly created spam. Though it might temporally penalize links to normal sites, we can correct their importance after we invent spam detection methods for novel spamming techniques.
- Crawling spam sites is a sheer waste of time and resources. Most crawlers have spam filters, but such filters cannot quickly adapt themselves to new spamming methods. By reducing the crawling priority of new links from hijacked pages in detected sites, we can avoid collecting and storing new spam sites, until spam filters are updated.

In order to identify hijacked sites, we consider characteristics of the link structure around hijacked sites. As Figure 1 indicates, hijacked sites are supposed to have a certain number of links to both normal and spam sites, and exist at the boundary of them. To detect this boundary, we take account of trustworthiness and spamicity of whole sites. Normal sites would have high trustworthiness and low spamicity, and in contrast, spam sites would have low trustworthiness and high spamicity. These relations will be reversed at the link between normal sites and spam sites, or where link hijacking occurs. Based on this idea, we detect the point where trustworthiness and spamicity are reversed in order to extract hijacked sites.

In addition, we focus on the fact that hijacked sites have links pointing to both normal and spam sites. Out-neighbors of normal sites will show much more trustworthiness than spamicity, and vice versa. Thus, it would be assumed that overall trustworthiness and spamicity in out-neighbors of hijacked sites are equivalent compared to those of normal or spam.

Trustworthiness and spamicity of a site can be evaluated by some link-based ranking algorithms such as modified versions of PageRank. For each site, we calculate white and spam scores using two different modified PageRanks. Intuitively, these scores represent the degree of trustworthiness and spamicity of sites.

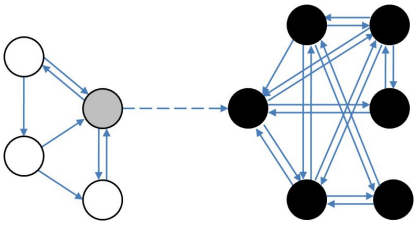


Fig. 1. Link structure around a hijacked site. White, gray, and black nodes represent normal, hijacked and spam sites respectively. A dashed link from the hijacked site to a spam site is the hijacked link.

The rest of this paper proceeds as follows. In Section 2, we review background knowledge for PageRank and link spamming. Section 3 introduces modified PageRank algorithms and several approaches to detect or demote link spamming. Section 4 presents our method for detecting hijacked sites. In Section 5, we report the experimental result of our algorithm. Finally, we conclude and summarize the result of our approach.

2 Background

2.1 Web Graph

The entire Web can be considered as a directed graph. We can denote the Web as $G = (V, E)$, where V is the set of nodes and E is a set of directed edges $< p, q >$. Node v can be a page, host or site.

Each node has some incoming links(*inlinks*) and outgoing links(*outlinks*). $In(p)$ represents the set of nodes pointing to p(the in-neighbors of p) and $Out(p)$ is the set of nodes pointed to by p(the out-neighbors of p). We will use n to describe $\|V\|$, the number of total nodes on the Web.

2.2 PageRank

PageRank [4] is one of the most well-known link-based ranking algorithms. The basic idea of PageRank is that a Web page is important if it is linked by many other important pages. This recursive definition can be showed as following matrix equation:

$$\mathbf{p} = \alpha \cdot \mathbf{T} \times \mathbf{p} + (1 - \alpha) \cdot \mathbf{d}$$

where $\mathbf{p}$ is PageRank score vector, $\mathbf{T}$ is transition matrix. $T(p, q)$ is $1/\|Out(q)\|$ if there is a link from node q to node p, and 0 otherwise. The decay factor $\alpha < 1$ (usually 0.85) is necessary to guarantee convergence and to limit an effect of rank sink. $\mathbf{d}$ is a uniformly distributed random vector. Instead of following links to next pages, we can jump from a page to a random one chosen according to distribution $\mathbf{d}$.

2.3 Link Spamming

After the success of Google which adopted PageRank as the main ranking algorithm, PageRank became a primary target of link spammers. Z. Gyöngyi et al. studied about link spam in [6] and introduced an optimal link structure to maximize PageRank score, a *spam farm*. The spam farm consists of a target page and boosting pages. All boosting pages link to the target page in order to increase the rank score of it. Then, the target page distributes its boosted PageRank score back to supporter pages. By this, members of a spam farm can boost their PageRank scores. Due to the low costs of domain registration and Web hosting, spammers can create spam farms easily, and actually there exist spam farms with thousands of different domain names [10].

In addition to constructing the internal link structure, spammers make external links from outside of spam farms to attract search engines and provide PageRank scores to the target page. To make links from non-spam pages to spam pages, various hijacking techniques are exploited. Spammers send trackbacks that lead to spam sites, or post comments including links pointing to spam pages. Expired domains can be bought by spammers, and then changed to spam sites. Spammers can also sponsor Web sites to insert advertisements of spam sites on their pages.

Examples of link hijacking are shown in Figure 2 and 3. Figure 2 illustrates spam comments on blogs. Spammers post comment containing a large amount of links to spam sites. By this, they can create massive links to spam sites easily in a short period. Figure 3 shows a page containing many links to spam pages. Although it is the page about Java and Linux, it contains links pointing to a lot of sport sites which are densely connected together. Note that major search engines and blog services employ counter-measures like `rel="nofollow"` tags,

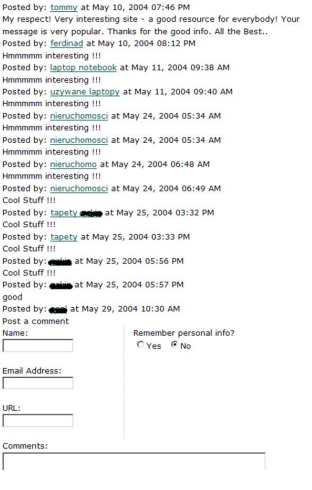


Fig. 2. Spam comments on blog



Fig. 3. Link hijacking by advertisement

which is attached to hyperlinks that should be ignored by link-based ranking algorithms [15]. However, there still exist a number of Web services that do not support such means, and hijacking techniques like buying expired domains cannot be penalized by `"nofollow"` tag.

3 Previous Work

3.1 TrustRank and Anti-trustRank

To improve the PageRank algorithm, Gyöngyi et al. presented the TrustRank algorithm [8]. The basic intuition of TrustRank is that good pages seldom link to spam pages. People trust good pages, and can trust pages pointed to by good pages. Like this, trust can be propagated through the link structure of the Web. Therefore, in TrustRank, a list of highly trustworthy pages is created as a seed set and each of these pages is assigned a non-zero initial trust score, while all the other pages on the Web have initial values of 0. After computation, good pages will get a decent trust score, and spam pages get a lower trust scores.

The matrix notation of TrustRank is following:

$$\mathbf{t} = \alpha \cdot \mathbf{T} \times \mathbf{t} + (1 - \alpha) \cdot \mathbf{d}^{\tau}$$

where $\mathbf{t}$ is TrustRank score vector, α is decay factor(0.85), and $\mathbf{d}^{\tau}$ is a random jump distribution vector where

$$d_p{}^{\tau} = \begin{cases} 1/\|S\|, & \text{if } p \text{ is in trust seed set } S \\ 0, & \text{otherwise} \end{cases} .$$

Krishnan et al. proposed Anti-TrustRank to find out spam pages [11]. Instead of selecting good pages as a seed set, Anti -TrustRank starts score propagation from spam seed pages. Each spam page is assigned Anti-Trust score and this score is propagated along incoming links.

3.2 Core-Based PageRank

Core-based PageRank was suggested by Gyöngyi et al. [10]. When we have a seed set S, we describe a core-based PageRank score of a page p as $\mathbf{PR}'(p)$. A core-based PageRank score vector $\mathbf{p}'$ is :

$$\mathbf{p}' = \alpha \cdot \mathbf{T} \times \mathbf{p}' + (1 - \alpha) \cdot \mathbf{d}^{\nu}$$

where a random jump distribution $\mathbf{d}^{\nu}$ is :

$$d_p{}^{\nu} = \begin{cases} 1/n, & \text{if } p \text{ is in seed set } S \\ 0, & \text{otherwise} \end{cases} .$$

Core-based PageRank is different from TrustRank by random jump vector. Core-based PageRank adopts a random jump distribution $1/n$, which is normalized by the number of whole Web site, instead of $1/\|S\|$.

In this paper, we use two types of core-based PageRank scores.

- $\mathbf{PR}^+$ = a core-based PageRank score with a trust seed set S^+.
- $\mathbf{PR}^-$ = a core-based PageRank score with a spam seed set S^-.

Z. Gyöngyi et al. mentioned a core-based PageRank with a spam seed set in [10]. They refer to blending $\mathbf{PR}^+$ and $\mathbf{PR}^-$ (e.g. compute weighted average) in order to detect spam pages. However, this view is different from ours. We think $\mathbf{PR}^+$ and $\mathbf{PR}^-$ separately and focus on the change in scores through links to discover hijacked links.

3.3 Other Approaches

Several approaches have been also suggested for the purpose of detecting and demoting link spam.

To demote spam pages and make PageRank resilient to link spamming, Wu et al. complemented TrustRank with topicality in [9]. They computed TrustRank score for each topic to solve a bias problem of TrustRank.

To detect link spam, Benczur et al. introduced SpamRank [12]. SpamRank checks PageRank score distributions of all in-neighbors of a target page. If this distribution is abnormal, SpamRank regards a target page as spam and penalizes it. Gyöngyi et al. suggested Mass Estimation in [10]. They evaluated *spam mass*, a measure of how many PageRank scores a page gets through links from spam pages. Saito et al. employed a graph algorithm to detect Web spam [13]. They extracted spam pages by the strongly connected component decomposition and used them as a seed set to separate spam pages from non-spam pages.

Du. et al. discussed an effect of hijacked links on the spam farm in [7]. They introduced an extended optimal spam farm by dropping the assumption of [6] that leakage by link hijacking is constant. Although Du. et al. considered link hijacking, they did not studied features of hijacking and its detection, which is different from our work.

As we reviewed, although there are various approaches to link spam, link hijacking has never been explored closely. In this paper, we propose a new approach to discover hijacked links and sites. With our approach, we expect to contribute to new spam detection techniques and improve the performance of link-based ranking algorithms.

4 Link Hijacking Detection

Based on characteristics of the change in trustworthiness and spamicity around hijacked sites, we compute a hijacked score of sites.

To begin with, we assign white score and spam score to whole Web sites. We employ the notion of $\mathbf{White}(p)$ and $\mathbf{Spam}(p)$, which represent the degree of trustworthiness of site p and the degree of spamicity of site p, respectively. For example, TrustRank score and core-based PageRank score with a white seed

set can be used as white score. Anti-TrustRank score and core-based PageRank score with spam seeds are available for spam scores.

With these scores, we define the *score reversal* relation. Hijacked site p would have a higher white score than its spam score, and spam site would have lower white score than its spam score. These features can be described with *relative trust*, **RT**.

$$\mathbf{RT}(p) = \log\big(\mathbf{White}(p)\big) - \log\big(\mathbf{Spam}(p)\big) - \delta \ .$$

We used log value since PageRank score obeys power law distribution. If the log difference between white score and spam score of site p is bigger than threshold δ, p is more likely to be normal. In contrast, $\mathbf{RT}(p)$ lower than 0 implies p might be spam. When we use a higher δ value, we consider the white score of hijacked sites is much higher than its spam score. Therefore, our algorithm will choose a site nearer to normal side as a hijacked. When we use a lower δ value, we regard the white score of a hijacked site is lower, so select sites close to spam side as hijacked.

As we see in Section 1, if p is hijacked, there must be spam sites in out-neighbors of p by definition. In addition to this, we take account of only sites with a lower white score and higher spam score than those of p. With this condition, we can check sites that are more likely to be spam than p is. We will call this relation *score reversal*.

With the score reversal relation, we define the hijacked score. First, we create $R(p)$, a set of sites which have the score reversal relation with site p.

$$R(p) = \left\{ r \;\middle|\; \begin{array}{l} r \in \mathit{Out}(p) \wedge \mathbf{RT}(r) < 0 \ \ \wedge \\ \mathbf{White}(r) < \mathbf{White}(p) \wedge \mathbf{Spam}(r) > \mathbf{Spam}(p) \end{array} \right\} .$$

Then, we define a set H of hijacked candidates. A hijacked site h will have higher trustworthiness than its spamicity, and will have at least one out-neighbor node that is in the score reversal relation with it.

$$H = \big\{ h \;\big|\; \mathbf{RT}(h) \geq 0 \ \wedge \ R(h) \ \neq \phi \big\} \ .$$

Finally, we compute the hijacked score of h. Two different hijacked detection methods are designed.

As a naive approach, we considered only the total trustworthiness difference between a hijacked candidate and sites in the score reversal relation with it. This can be described as following:

$$\mathbf{H}_{rev}(h) = \sum_{r \in R(h)} \big\{ \log\big(\mathbf{White}(h)\big) - \log\big(\mathbf{White}(r)\big) \big\} \ .$$

We found out that when both trustworthiness and spamicity of around a hijacked site are considered, a higher detection precision can be achieved. Overall out-neighbor trustworthiness of a hijacked site will be obtained by the average **RT** of total normal outnodes. In the same manner, spamicity of out-neighbors

will be computed by the average $-\mathbf{RT}$ of all spam outnodes. To obtain out-neighbor trustworthiness and spamicity of a hijacked site, we divide $Out(h)$ into a set of normal sites $nOut(h)$ and a set of spam sites $sOut(h)$.

$$nOut(h) = \left\{ n \;\middle|\; n \in Out(h) \wedge \mathbf{RT}(n) \geq 0 \right\},$$
$$sOut(h) = \left\{ s \;\middle|\; s \in Out(h) \wedge \mathbf{RT}(s) < 0 \right\}.$$

The following is the improved hijacked score $\mathbf{H}_{all}(h)$.

$$\mathbf{H}_{all}(h) = \frac{\sum_{n \in nOut(h)} |\mathbf{RT}(n)|}{\|nOut(h)\| + \lambda} \cdot \frac{\sum_{s \in sOut(h)} |\mathbf{RT}(s)|}{\|sOut(h)\| + \lambda}.$$

While λ is a smoothing factor which alleviates an effect of $\mathbf{RT}$ when the number of normal or spam out-neighbors is very small.

$\mathbf{H}_{all}(h)$ increases as both trustworthiness and spamicity grow. When either trustworthiness or spamicity is getting lower, $\mathbf{H}_{all}(h)$ decreases since site h seems to be a spam or normal site.

5 Experiments

To test our method, we prepared a large Web data set. White seed set and spam seed set were also generated to compute white and spam scores. As for computations of white and spam scores, we used a core-based PageRank score $\mathbf{PR}^{+}$ and $\mathbf{PR}^{-}$ pair. With white and spam scores, we obtained two kinds of hijacked scores and evaluated the precisions of results.

5.1 Data Set and Seed Set

To evaluate our algorithm, we performed experiments on a large-scale snapshot of our Japanese Web archive built by a crawling conducted in May 2004. Basically, our crawler is based on breadth-first crawling [14], except that it focuses on pages written in Japanese. Pages outside the .jp domain are collected if they were written in Japanese. We used a site as a unit when filtering non-Japanese pages. The crawler stopped collecting pages from a site, if it could not find any Japanese pages on the site within the first few pages. Hence, our dataset contains fairly amount of pages in English or other languages. The percentage of Japanese pages is estimated to be 60%. This snapshot is composed of 96 million pages and 4.5 billion links.

We use an unweighted site level graph of the Web, in which nodes are Web sites and edges represent the existence of links between pages in different sites. In the site graph, we can easily find dense connections among spam sites that cannot be found in the page level graph. To build the site graph, we first chose the representative page of each site that has 3 or more inlinks from other sites, and whose URL is within 3 tiers (e.g. http://A/B/C/). Then, pages below each representative page are contracted to one site. Finally, edges between two sites are created when there exist links between pages in these sites. The site graph

built from our snapshot includes 5.8 million sites and 283 million links. We call this dataset Web graph in our experiments.

To compute white and spam scores, we constructed trust seed set and spam seed set. We used manual and automated selecting methods for both seed sets.

In order to generate the white seed set, we computed PageRank score of whole sites and performed a manual selection on top 1,000 sites with a high PageRank score. Well-known sites (e.g. Google, Yahoo!, and MSN), authoritative university sites and well-supervised company sites are selected as white seed sites. After a manual check, 389 sites are labeled as trustworthy sites. To make up for the small size of a seed set, we extracted sites with specific URL including .gov (US governmental sites) and .go.jp (Japanese governmental sites). In the end, we have 40,396 sites as trust sites.

For the spam seed set, we chose sites with high PageRank score and checked manually. Sites including many unrelated keywords and links, redirecting to spam sites, containing invisible terms and different domains for each menu are judged spam sites. We have 1,182 sites after a manual check. In addition, we used automatically extracted spam sites obtained by [13]. Saito et al. obtained this large spam seed set by following steps. First, they extracted strongly connected components (SCC) from the Web graph. Since spam sites tend to construct a densely connected link structure, it could be assumed that spam sites form SCC. In addition to detecting spam located at the fringe of the Web, Saito et al. counted up maximal cliques in the largest SCC, or a core. Cliques whose sizes are less than 40 were extracted from the core and about 8,000 spam sites were obtained. Finally, they used these spam sites as a reliable spam seed set and expanded them by a minimum cut technique that separates links between spam and non-spam sites. Since this method showed a high extraction precision, we used their spam sites as seed sites. Finally, Total 580,325 sites are used as a spam seed set.

5.2 Types of Hijacking

In order to understand a layout of sites at the boundary of normal and spam, we randomly chose 1,392 sites near to spam seeds. These samples are checked by hand and classified into 4 categories; hijacked, normal, spam and unknown. Unknown sites were written in unrecognizable languages such as Chinese, Dutch, German and so on. Table 1 shows the result of classification. The 33% of total sites was identified as hijacked, and these 465 sites are divided into 8 types as follows.

- Blog sites with spam comments or trackbacks and public bulletin boards containing comments pointing to spam sites.
- Expired sites bought by spammers. Spammers can buy expired domains and use them for spam sites. Since Web sites tend to maintain links pointing to expired domains for a while, spammers are able to get links from them.
- Hosting sites that include spam sites of some customers.

Table 1. Types of evaluation sample site

Site type	Number of sites
Hijacked	465
Normal	345
Spam	576
Unknown	6
Total	1392

Table 2. Types of hijacking

Hijacking type	Number of sites
Blog and bbs	117
Expired sites	77
Hosting sites	64
Link to expired site	60
Link register sites	55
Link to spam by mistake	50
Advertisement to spam	32
Server statistics	10
Total	465

- Normal sites that point to expired hijacking sites. Expired hijacking sites are turned into spam sites by spammers, so links from normal to these expired sites can be considered hijacked links.
- Free link registration sites that allow spammers to register links on them.
- Normal sites that create links to spam sites by mistakes. Authors of some sites make links pointing to spam sites by themselves. Since it is hard for non-experts to identify spam sites, they believe those spam sites are useful.
- Normal sites that contain advertising links pointing to spam sites. Spammers can insert links on normal sites by sponsoring them.
- Sites with public access statistics that show links to referrers. Spammers access such sites frequently, and then plant links to spam sites in the referrer list.

Table 2 shows the number of sites in each hijacking type. We can see that the most frequently used technique is blog and bbs hijacking. Expired hijacking is a quite popular technique among spammers, too. Particularly, domains for official sites of movies and singers are prone to be hijacked because they are used for a while, not permanently.

5.3 Evaluation

Using the white and spam seed sets, we computed core-based PageRank scores for white and spam scores. Hijacked scores were obtained as well, with different δ values. (See Section 4.)

Precision with a naive approach. For δ from -2 to $+2$, we chose top 200 sites with high $\mathbf{H}_{rev}$ scores and checked them by hand. Detected samples are categorized into hijacked, normal, spam, and unknown. The detail is shown in Table 3. The best precision 42.5% was obtained when δ was 1.

Precision with a suggested approach. With different δ values from -5.0 to 1.0, we computed $\mathbf{H}_{all}$ score and evaluated top 200 sites. In order to determine smoothing factor λ, we calculated $\mathbf{H}_{all}$ score of sample sites mentioned in

Table 3. Top 200 precision with a naive approach.

δ	-2	-1	0	1	2
Hijacked	67	74	83	85	72
Normal	14	22	46	62	97
Spam	112	97	70	42	27
Unknown	7	7	5	4	4
Total	200	200	200	200	200
Precision	33.5%	37%	41.5%	42.5%	36%

Table 4. Top 200 precision with a suggested approach

δ	-5	-4	-3	-2	-1	0	1
Hijacked	84	100	106	135	132	132	114
Normal	6	8	10	12	16	31	41
Spam	110	91	81	50	48	33	42
Unknown	0	1	3	3	4	4	3
Total	200	200	200	200	200	200	200
Precision	42%	50%	53%	67.5%	66%	66%	57%

Section 5.2 and examined the result. Since the best precision for top 200 sites was obtained when $\lambda = 40$, we used the same value for the whole sites evaluation. The result is described in Table 4. We detected hijacked sites with the best precision of 67.5% when δ is -2.

As both Table 3, 4 indicate, normal sites increase as δ increases. This is because with a higher δ, hijacked sites should have a higher white score. Likewise, as δ decreases, the proportion of spam sites increases. This means our algorithms become tolerant and consider sites with a relatively high spam score as hijacked. As for λ, we found out that as λ increases, $\mathbf{H}_{all}$ of spam sites decreases. However, if the value of λ exceeds 40, the number of spam sites in the top result hardly changes. The ratio of normal sites with high $\mathbf{H}_{all}$ remain stable regardless of λ.

We computed $\mathbf{H}_{rev}$ score with a TrustRank and Anti-TrustRank score pair and investigated the performance. However, the precision was far worse than that with a core-based PageRank pair.

6 Conclusion

In this paper, we proposed a new method for link hijacking detection. Link hijacking is one of the essential methods for link spamming and massive hijacked links are now being generated by spammers. Since link hijacking has a significant impact on link-based ranking algorithms, detecting hijacked sites and penalizing hijacked links are serious problems to be solved.

In order to identify hijacked sites, we focused on characteristics of the link structure around hijacked sites. Based on the observation that white and spam score reversal occurs between hijacked sites and hijacking sites, we computed hijacked scores.

Experimental result showed that our approach is quite effective. Our best result for finding hijacked sites outperformed about 25% compared to a naive approach.

References

1. Nakamura, S., Konishi, S., Jatowt, A., Ohshima, H., Kondo, H., Tezuka, T., Oyama, S., Tanaka, K.: Trustworthiness Analysis of Web Search Results. In: 11th European Conference on Research and Advanced Technology for Digital Libraries, Budapest, Hungary (2007)
2. Ntoulas, A., Najork, M., Manasse, M., Fetterly, D.: Detecting Spam Web pages through Content Analysis. In: 15th International Conference on World Wide Web, Edinburgh, Scotland, UK (2006)
3. Fetterly, D., Manasse, M., Najork, M.: Spam, Damn Spam, and Statistics: Using Statistical Analysis to Locate Spam Web Pages. In: 7th International Workshop on the Web and Databases, Paris, France (2005)
4. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, Stanford University, Stanford, CA, USA (1998)
5. Gyöngyi, Z., Garcia-Molina, H.: Web Spam Taxonomy. In: 1st International Workshop on Adversarial Information Retrieval on the Web, Chiba, Japan (2005)
6. Gyöngyi, Z., Garcia-Molina, H.: Link Spam Alliance. In: 31st International Conference on Very large Data Bases, Trondheim, Norway (2005)
7. Du, Y., Shi, Y., Zhao, X.: Using Spam Farm to Boost PageRank. In: 3rd International Workshop on Adversarial Information Retrieval on the Web, Banff, Alberta, Canada (2007)
8. Gyöngyi, Z., Garcia-Molina, H., Pedersen, J.: Combating Web spam with TrustRank. In: 30th International Conference on Very Large Data Bases, Toronto, Canada (2004)
9. Wu, B., Goel, V., Davison, B.D.: Topical TrustRank: Using Topicality to Combat Web Spam. In: 15th International Conference on World Wide Web, Edinburgh, Scotland, UK (2006)
10. Gyöngyi, Z., Berkhin, P., Garcia-Molina, H., Pedersen, J.: Link Spam Detection Based on Mass Estimation. In: 32nd international conference on Very Large Data Base, Seoul, Korea (2006)
11. Krishnan, V., Raj, R.: Web Spam Detection with Anti-TrustRank. In: 2nd International Workshop on Adversarial Information Retrieval on the Web. Edinburgh, Scotland, UK (2006)
12. Benczur, A., Csalogány, K., Sarlós, T., Uher, M.: SpamRank-fully automatic link spam detection. In: 1st International Workshop on Adversarial Information Retrieval on the Web, Chiba, Japan (2005)
13. Saito, H., Toyoda, M., Kitsuregawa, M., Aihara, K.: A Large-scale Study of Link Spam Detection by Graph Algorithms. In: 3rd International Workshop on Adversarial Information Retrieval on the Web, Banff, Alberta, Canada (2007)
14. Najork, M., Wiener, J.L.: Breadth-first Crawling Yields High-quality Pages. In: 10th international conference on World Wide Web, Hong Kong (2001)
15. The Official Google Blog, `http://googleblog.blogspot.com/2005/01/preventing-comment-spam.html`

A Data Driven Ensemble Classifier for Credit Scoring Analysis

Nan-Chen Hsieh[1], Lun-Ping Hung[2], and Chia-Ling Ho[3]

[1] Department of Information Management, National Taipei College of Nursing, No. 365, Min-Ten Road 11257, Taipei, Taiwan, R.O.C.,
nchsieh@ntcn.edu.tw
[2] Department of Information Management, Technology and Science Institute of Northern, Taiwan No. 2, Xueyuan Rd., Peitou, 112 Taipei, Taiwan, R.O.C
[3] Department of International Trade, Technology and Science Institute of Northern Taiwan, No. 2, Xueyuan Rd., Peitou, 112 Taipei, Taiwan, R.O.C.

Abstract. This study focuses on predicting whether a credit applicant can be categorized as good, bad or borderline from information initially supplied. Given its importance, many researchers have recently worked on an ensemble of classifiers. However, to the best of our knowledge, unrepresentative samples drastically reduce the accuracy of the deployment classifier. Few have attempted to preprocess the input samples into more homogeneous cluster groups and then fit the ensemble classifier accordingly. For this reason, we introduce the concept of class-wise classification as a preprocessing step in order to obtain an efficient ensemble classifier. This strategy would work better than a direct ensemble of classifiers without the preprocessing step. The proposed ensemble classifier is constructed by incorporating several data mining techniques, mainly involving optimal associate binning to discretize continuous values; neural network, support vector machine, and Bayesian network are used to augment the ensemble classifier. In particular, the Markov blanket concept of Bayesian network allows for a natural form of feature selection, which provides a basis for mining association rules.

Keywords: Clustering, ensemble classifier, neural network, Bayesian network, class-wise classification, credit scoring system.

1 Introduction

Generally, credit scoring problems are related to classification by statistical methods. Investigating more sophisticated classifiers to match the characteristics of the samples is crucial in providing results that meet the needs of particular credit scoring applications. Techniques for developing classifiers have evolved from simple parametric to nonparametric statistical methods. Altman [1] applied simple parametric discriminant analysis and multiple discriminant analysis (MDA) to the corporate credit granting problem, then compared model's performance using linear discriminant analysis and neural networks. Lawrence and Arshadi [2] used a logistic model to analyze the loan management problem using a series of borrower and bank variables. Charitou et al.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 351–362, 2009.
© Springer-Verlag Berlin Heidelberg 2009

[3] used logit and neural network models to predict failed and non-failed UK public industrial firms over the period 1988-1997. These studies concluded that there certainly should be further studies and tests using statistical and artificial intelligence techniques and suggested a combined approach for predictive reinforcement.

Other efforts are leading to the investigation of nonparametric statistical methods for credit scoring applications. For example, McKee [4] used rough sets for bankruptcy prediction and compared the performance of rough sets versus auditor signaling rates. A distinguished classifier, Bayesian networks (BN), has been proposed as a probabilistic white-box classifier which permits higher order relationships between the variables of the problem under study. Sarkar and Sriram [5] utilized Bayesian networks for early warning of bank failures. They found the naïve Bayesian network and the composite attribute Bayesian network classifiers to have superior performance compared to the induced decision tree algorithm.

The idea of conventional classifiers is generally based on a single classifier or a simple combination of these classifiers, showing moderate performance. However, even exquisitely designed classifiers still have deficiencies which cannot appropriately distinguish the samples. One way to alleviate the classifier and data match problem is to use an ensemble of classifiers. A variety of classifiers, either different types of classifiers or different instantiations of the same classifier, are combined before a final classification decision is made. Thus, the ensemble classifier allows different characteristics of the samples to be handled by classifiers suited to their particular needs, and provides an extra degree of bias/variance tradeoff. Recently, the ensemble classifier has been demonstrated to be outperformed by a single classifier in having greater accuracies and smaller prediction errors when applied to the credit scoring datasets [6-7]. However, the potential to reduce the generalization error of a single classifier varies among different training data sets. This study investigates the concept of preprocessing the data set into more homogeneous cluster groups first and then fitting the ensemble classifier for stably predicting the categories of applicants. For this reason, we introduce the concept of class-wise classification as a preprocessing step in order to obtain an efficient ensemble classifier. This strategy would work better than a direct ensemble of classifiers without class-wise classification. The meaning of effective ensemble classifier is twofold, relating to accuracy and easy interpretation of classified results.

This study addresses the following research questions. First, a class-wise classification method is proposed to guide the reclassified samples into more homogeneous cluster groups that can be used to develop a well-performing ensemble classifier for credit scoring applications. Second, the proposed architecture focuses on fusing three types of classifiers – Neural network (NN), Bayesian network (BN), and Support vector machine (SVM) – which are simple to implement and have been shown to perform well in credit prediction [5], [8]. The rationale of employing these classifiers is that NNs are generally superior to the conventional classifiers, BNs can easily model complex relationships among variables, and SVMs can be used as the benchmark technique. Third, NNs and SVMs work well for continuous and discrete valued features, but BNs generally use only discrete valued features. Proper discretization of continuous values is critical for the building of a BN classifier. This study employs a heuristic method based on the assumption of linear dependence, as measured by

correlations between variables to the target features, so an optimal associate binning technique for discretization of continuous values was gained. Through discretization, continuous values are converted into discrete values with several states. Fourth, the use of the same training data set for all individual classifiers may reduce the diversity among individual classifiers, while different training sets for individual classifiers may decrease the accuracy of individual classifiers. It is important to construct appropriate training data sets that maintain a good balance between accuracy and diversity among individual classifiers in an ensemble classifier. We use the class-wise classification mechanism to reclassify samples into homogeneous cluster groups, and used the class-wise bagging ensemble to train individual classifiers. This strategy would increase the diversity and accuracy of an ensemble classifier. Fifthly, the Markov blanket (MB) concept of BN allows for a natural form of feature selection, which provides a basis concept for mining constrained association rules. The learned knowledge is presented in multiple forms, including causal diagram and constrained association rules.

For a better understanding of our study, section 2 describes this study's research motivation. Section 3 describes the analysis methodology. Section 4 discusses the findings of this study and offers observations about practical applications and directions for future research.

2 Establishing Homogeneous Cluster Groups for Building an Effective Ensemble Classifier

The learning of credit scoring classifiers is based on historical samples with known classes, which are usually "good credit" and "bad credit". In theory, every sample has a fixed class membership in the known classes, but this theory is unachievable in the real world, since samples with known classes are quite limited, and some collected samples are unrepresentative of the population to be analyzed. This study focuses on reclassifying training samples into more homogeneous cluster groups, and predicting whether an acquired applicant categorizes not only as "good credit" and "bad credit" but also "borderline credit" (samples tending toward different credit status/samples tending away the same credit status) based on initial applicant characteristics.

Figure 1 shows the ideal classification boundary produced by conventional classifiers. The conventional classifier can be a simple classifier or a combination of classifiers. Sometimes, even with highly accurate classifiers, the classifier's ability to predict new applicants is still limited. This is because misclassification patterns may appear in the training data set, and the uncertainty in class membership will increase the learning complexity. If the quality of the sample's class membership can be improved, the classification ability of classifiers would significantly increase. Therefore, the fundamental requirement of the classifiers we build is to give the correct target class membership that conforms to the feature values of samples. If we adopt the idea of unsupervised classification and classify input samples into homogeneous cluster groups while neglecting the original sample's class membership, then we can reassign each sample to proper target class membership by clustering results. This approach allows us to find similarities in samples and to group samples into homogeneous

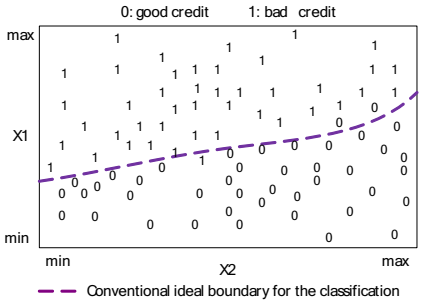


Fig. 1. Conventional ideal boundary for the classification

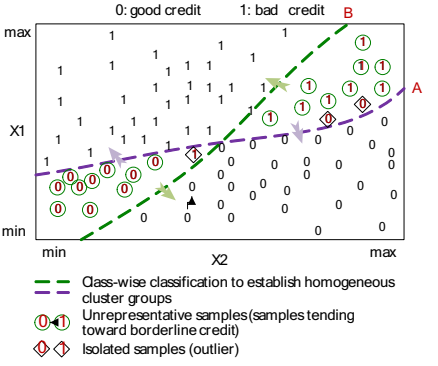


Fig. 2. Class-wise classification to establish homogeneous cluster groups

cluster groups, thus alleviating misclassifications of class membership from the original training data set.

Figure 2 presents the concept behind class-wise classification by using a hybrid-clustering technique. The goal of the learning problem depicted in this figure is to separate the good credit from the bad credit. However, even line A can separate the good credit and bad credit perfectly, it is still possible that line A is not a good classifier due to misclassifications of class membership in the input samples. Suppose that line B further divided the feature space into four regions. The marked samples indicate those samples tending toward different credit statuses or samples tending away from the same credit status. Then, referred to sample's original class memberships, we can reassign the marked samples to new class memberships, and use the new input samples for training classifiers. Preprocessing input samples by the hybrid-clustering technique is useful in defining proper class memberships.

3 Description of the Analysis Methodology

3.1 Discretization of Continuous Features

A real world German credit data set from the UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/datasets.html) was used to test the prediction effectiveness of the ensemble classifiers. The German credit data set consists of a set of loans given to a total of 1000 applicants, consisting of 700 samples of creditworthy applicants and 300 samples where credit should not be extended. For each applicant, 20 variables describe credit history, account balances, loan purpose, loan amount, employment status, and personal information. Each sample contains 13 categorical, 3 continuous, 4 set/binary features, and 1 class feature. The German credit data set is challenging because it is unbalanced and contains a mixture of continuous and categorical values, which confounds the task of classification learning. For a better understanding of our solution, the overall architecture of the decision making process is shown in Figure 3.

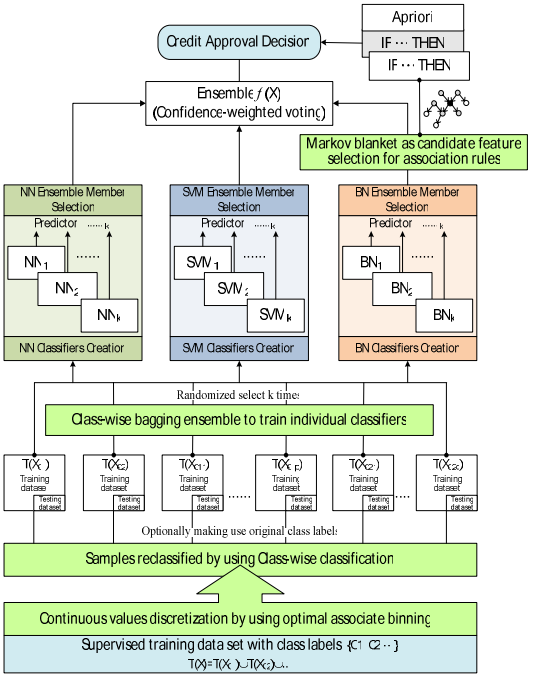


Fig. 3. The overall architecture of the credit approval decision making process

The number of states chosen for discretization drastically affects the prediction utility of classifiers. It is desired to increase the number of states of variables for the sake of reducing the information lost during the discretization processes, but more states require more learning effort and will complicate the learning process. As a result, properly discretized values can simplify the learning process and may improve the utility of the classifiers built. The discretization process can be achieved during the construction of the classifier, but more often discretization is considered as a separate knowledge discovery step before the induction of the classifier, as in this study. The associate discretization technique thus can be used to screen unimportant predictive features. This approach can improve the classifier's utility by selecting the optimal subset of features which best classifies the training data set.

Discretization methods are classified as unsupervised and supervised methods. Unsupervised methods do not consider the value of the target features while supervised methods do. The simplest discretization method, equal interval width, divides the intervals for a variable into k equal sized bins, where k is a user-defined parameter. A relative method, equal frequency intervals, divides a continuous variable into k bins, where each bin contains m adjacent values (m=k). The unsupervised methods do not consider the associations among independent and target features in setting the partition boundaries; it is likely that information will be lost during the binning process. On the other hand, supervised methods consider the associations among predictive and target features. Kerber [9] proposed a statistically justified heuristic method for supervised discretization. Entropy-based methods have recently been developed for feature discretization. Fayyad and Irani [10] used a recursive entropy minimization heuristic method for discretization with a Minimum Description Length criterion to control the number of intervals produced over the continuous space. Dougherty et al. [11] compared several unsupervised and supervised discretization methods. They found that the performance of the Naïve-Bayes algorithm significantly improved when features were discretized using the entropy-based method.

For the German credit data set, we kept the original symbols for each categorical feature. The data type of features Telephone and Foreign-worker was changed from

categorical to binary, since they have only two values. The data type of features Installment-rate-in-percentage-of-disposable-income, Number-of-existing-credits-at-this-bank and Present-residence-since was changed from categorical to set, since they have several values and with small domains. Therefore, the set data type is not suitable for using optimal associate binning technique to discretize them into categorical values. For features Duration-in-month, Credit-amount and Age-in-years, the corresponding continuous value domain is large, and optimal associate binning was used to discretize them into categorical values. After the discretization of continuous features, the average accuracy of a one-layer MLP NN increased from 78.46% to 89.16%.

3.2 Assessing Data Reduction Techniques for Refining the Quality of Samples

Typically, the choice of classifier affects the quality of a credit scoring system. Classifiers for credit scoring system can use linear/logistic regression, multiple discriminant analysis, neural networks, decision trees, support vector machines and Bayesian networks [8], [12-14]. However, the quality of training samples leads to misclassification patterns due to unrepresentative samples that drastically reduce the accuracy of the deployed classifier. One way to solve this problem is data reduction. In data reduction, samples are grouped or clustered to reveal hidden patterns, and samples that are not considered representative can be filtered out, leaving only the potentially interesting ones. Therefore, overhead can be significantly reduced by storing only the characteristics of the clusters instead of the individual data.

We identify the unrepresentative samples in this study by using class-wise classification to look for isolated and inconsistent clusters. Isolated clusters are thinly populated clusters or outlier samples; inconsistent clusters are clusters with inconsistent class values. A fundamental assumption made here is that the isolated clusters are far away from the core clusters. To improve the quality of classification, isolated clusters can be eliminated from training samples. However, inconsistent clusters are the cluster mixture samples with different class memberships, that is, the samples tend toward different credit status or tend away from the same credit status. These samples can be re-classified into more homogeneous cluster groups. We stand to gain if we can keep the feature values of training samples more homogeneous, would allow an efficient classifier to be built. Since the quality of classification is not easy to control, Punj and Steward [15] suggest that a two-stage clustering method consisting of a hierarchical model, such as Ward's minimum variance method, followed by a non-hierarchical method, such as K-means, can provide a better solution.

We have chosen to replace Ward's minimum variance with a two-step cluster analysis. Two-step cluster analysis works fine with mixed continuous/categorical types variables and can effectively and accurately handle small and large data sets. Furthermore, the Bayesian information criterion was used to automatically determine the number of clusters. The second stage follows the non-hierarchical K-means method, to determine the final solution, due to its efficiency. For this reason, this study uses a Two-step and K-means hybrid-clustering method to refine the quality of samples. First, we use the discretized samples as a new data set. Based on the shared characteristics of samples, Two-step cluster analysis automatically determines that the optimal number of clusters is three. We then employ K-means cluster analysis to find cluster-1, cluster-2 and cluster-3 three adhered cluster groups. All three cluster groups

are inconsistent in that they mix samples with good and bad class memberships, and may also contain samples with borderline credit status. By using the class-wise classification concept, we further divide the three cluster groups into six by cluster identifier and original class membership, so as to avoid the possible misclassified/ uncertainty class membership.

Herein, cluster-11, cluster-21 and cluster-31 are cluster groups with good credit status; however, they have dissimilar properties that need to be further separated for the sake of good classification and interpretation. The properties of samples in cluster-12 and cluster-22 are similar, so the behavioral trend of these samples might be tending changed in the near future. Also, the properties of marginal samples in cluster-31 and cluster-32 might be tending toward different credit status in the near future. Cluster groups cluster-11 and cluster-21 are surely the ones with good credit status.

3.3 An Ensemble Classifier for Credit Scoring Analysis

To overcome the performance limitation of single classifiers, the multi-classifier system, also called "ensemble classifier", has been proposed. The ensemble classifier is valuable due to its ability outperform the best individual classifier's performance [16]. Ensemble classifiers use the idea of combining information from multiple sources to reduce the variance of individual estimation errors and improve the overall classification results. Two points should be noted about the selection of individual classifiers. First, the choice of classifiers should mainly derive from the fact that these classifiers have been applied successfully in credit scoring analysis. Also, the design of these algorithms should be based on different theoretical concepts. This is particularly critical in combining classifiers, since they might explore complementary information for the classification task. In this study, the employed individual classifiers for the ensemble classifier are NN, BN, and SVM. The concept of ensembles appeared in the classification literature since the work by Nilson [17], and then in extensive studies beginning in the 1990s [8-9], [16], [18].

Ensemble methods such as cross-validation, bagging, and boosting have been proposed for forming ensembles of classifiers. The simplest method is the cross-validation ensemble where all ensemble members are trained on the same data set. Bagging and boosting [19] are other methods for improving the performance of ensemble classifier systems. For selecting training data, bagging generates replicated boot-strap samples of the data, while boosting adjusts the weights of training instances. Both methods can be combined into an individual classifier by voting. Bagging and boosting create perturbed versions of the training set so that ensemble members learn from different variants of the original training data. These ensemble algorithms not only increase the classification accuracy, but also reduce the chances of overtraining since the committee avoids a biased decision by integrating the different predictions from the individual base classifiers.

Importantly, the key to successfully building an ensemble classifier system is to construct appropriate input training sets and maintain a good balance between diversity and accuracy among individual classifiers in an ensemble. When using these classifiers, we confront two problems. The first is how to choose the optimal input feature subset for the classifier and how to select the best individual classifier for each ensemble classifier. These two problems are crucial because the feature subset choice influences the selection of an appropriate classifier and vice versa. For this reason, the

individual classifiers in this study are primarily optimized with respect to functional feature subspace selection. Feature subspace selection is performed by a data reduction technique in order to retain only the feature subset with the highest discrimination information.

Conventional ensembles train all ensemble members on the same data set. Some works [18] have explored the phenomenon that when a data partitioning strategy is employed, the experimental results vary significantly for different partitions of the same data collection, even when the number of samples in each data set is the same. It is necessary to have a precise specification of the partition in order to replicate an experiment or conduct fair comparisons. To simplify discussion, this study investigates the effectiveness of the class-wise bagging ensemble combination strategy, adapted to efficient combination of classifiers for credit decision applications.

Figure 3 has shown the proposed final credit scoring system, adopting the ensemble approach to generate enhanced results by grouping a set of classifiers of each ensemble: NNs, SVMs, and BNs. Owing to the diversity among individual base classifiers on diverse training data sets, the training data sets of individual base classifiers are derived from class-wise bagging ensemble combination, for the sake of preserving maximum diversity among different classifiers. In the class-wise bagging step, individual classifiers are trained independently via a bootstrap method. According to the proportion of samples, bootstrapping generates K replicas of training/testing data by repeated random re-sampling with replacement from each of the data sets, $T(X_{c1})$, $T(X_{c2})$,…. Then, to achieve maximum diversity of ensembles, each replicated training set is used to train a NN, a BN, and an SVM classifier of each ensemble. In the second step, individually created classifiers are aggregated by an ensemble combination technique to form a final decision. The concept of voting is both simple to implement and appealing, and can generally be applied to any type of classifier without relying on specific interpretations of the output. We thus use confidence-weighted voting in our ensemble of classifiers. The average performance of individual classifiers and the ensemble classifier with/without preprocessing by class-wise classification are shown

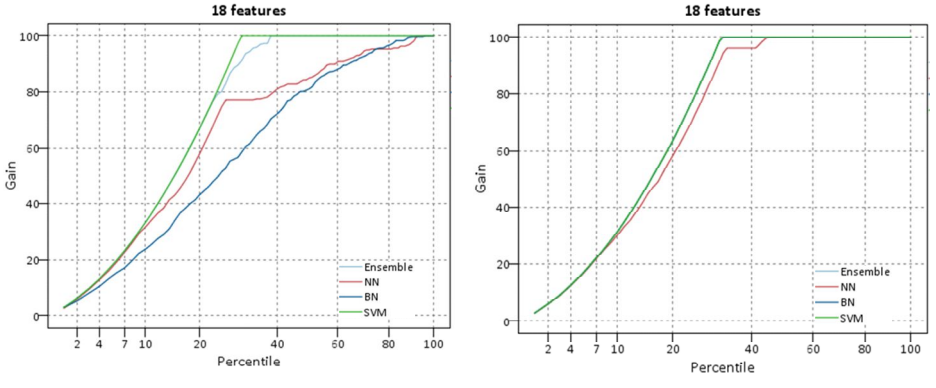


Fig. 4. Credit data set average classification results

as Gain charts in Figure 4. The results of the experiment show that improvements in performance have been achieved by applying the ensemble of classifiers for the training data set preprocessing both with and without class-wise classification.

3.4 Bayesian Learning and Markov Blanket for Modeling Knowledge

Aside from building effective ensemble classifiers, another aim of this study is to try to discover hidden patterns in the data set so that we can better understand the different characteristics of applicants and develop new strategies for credit approval analysis. A well-known Apriori algorithm has been proposed for mining association rules in a database. An association rule is considered relevant for decision making if it has support and confidence at least equal to some minimum support and confidence thresholds defined by the user. However, the extracted association rules are usually very large, to the present of a huge proportion of redundant rules conveying the same information. If we can employ the built classifier as a basis in mining constrained association rules, it can be of benefit in modeling valuable knowledge.

The BN model is a powerful knowledge representation and reasoning algorithm under conditions of uncertainty. The major advantage of BNs over many other types of predictive models, such as neural networks, is that unlike ‘‘black box’’ approaches, the BN structure represents the inter-relationships among the data set features. Thus, human experts can easily understand the network structures. Other advantages of BNs include explicit uncertainty characterization, as well as fast and efficient computation. They are highly adaptive and easy to build, and provide explicit representation of domain specific knowledge in human reasoning frameworks. BNs offer good generalization with limited training samples and easy maintenance when adding new features or new training samples. Several works have presented methods for mining knowledge by means of BNs [20-21].

A novel idea of BN for significant feature selection is the Markov blanket (MB). MB is defined as the set of input features such that all other features are probabilistically independent of the target features. That is, based on a general BN classifier, we

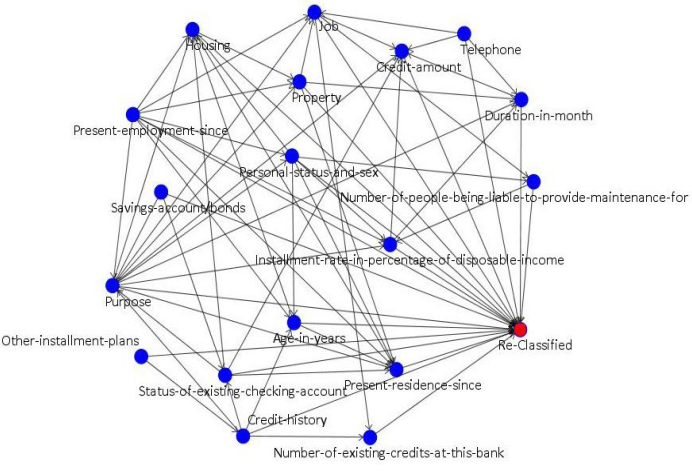


Fig. 5. Markov Blanket for German credit data set

can get a set of features that are in the MB of the target feature, and features provided by the MB are sufficient for perfectly estimating the distribution and for classifying the target feature classes. Suppose that we build a BN classifier with a complete input training data set. Then, the MB of the target feature forms the feature selection mechanism, and all features outside the MB are ignored from the BN. The remained features are then used to induce constrained association rules. In this study, all association rules are in the form, $Y \leftarrow X$, where Y is the target feature and X is a subset of features from the MB. The rationale of this method is that for any feature $x_i \notin \{\mathrm{MB} \cup x_j\}$, $x_i \neq x_j$ and $x_i, x_j \in \mathrm{MB}$, we have $P(x_i|\mathrm{MB},x_j) = P(x_i|\mathrm{MB})$. That is, the MB of a feature xi is a minimal set of features MB such that xi is conditionally independent of any other feature x_j in MB. Herein, for generating the MB, Pearson chi-square was employed as a technique to test independence, i.e. to assess whether paired observations of two features are independent of each other. The MB structure to be used for BN explores not only the relationship between target and predictive features, but also the relationships among the predictive features themselves. The Pearson chi-square test reported that eighteen predictive features are important to the target feature. Figure 5 depicts a MB for the German credit data set, composed of nineteen features, eighteen predictive features, and one target feature.

Obviously, the extracted association rules convey redundant information. Interactive strategies are proposed for pruning redundant association rules on the basis of equivalence relations, in order to enhance readability. The user will be presented with only the most informative non-redundant association rules, where the union of the antecedents (or consequents) is equal to the unions of the antecedents (or consequents) of all the association rules valid in the context. The resulting rules will have minimal antecedents and maximal consequents in the same equivalence class. The extraction of a set of rules without any loss of information will convey all the information in a set of association rules that are all valid according to the context. Table 1 lists some merged association rules of "bad credit," ordered by the length of the rule's antecedent, where each rule represents a property that was dominant or most strongly associated with "bad credit". The support and confidence criteria for filtering association rules are set as 10% and 80%, respectively.

Table 1. The association rules of partial causality for bad credit status

Bad Credit Status (Merged)					
ID	Consequent	Antecedent	Support%	Confidence%	Lift
1	cluster-12	Job = A172 and Number-of-existing-credits-at-this-bank = A2	4.38	61.54	2.90
2	cluster-22	Credit-history = A30	8.42	60.00	1.71
3	cluster-22	Credit-history = A33 and Number-of-existing-credits-at-this-bank = A2 and Job = A173	3.70	63.64	1.82
4	cluster-22	Job = A174 and Credit-history = A32 and Number-of-existing-credits-at-this-bank = A1	6.73	80.00	2.28
5	cluster-22	Job = A174 and Number-of-existing-credits-at-this-bank = A2	5.39	62.50	1.78
6	cluster-32	Credit-history = A32 and Job = A173 and Number-of-existing-credits-at-this-bank = A1	30.98	73.91	1.69
7	cluster-32	Job = A172 and Credit-history = A32 and Number-of-existing-credits-at-this-bank = A1	11.11	66.67	1.52

4 Conclusions

This study proposed a complete procedure for designing the architecture of an ensemble classifier for credit scoring analysis. Due to frequent uncertainty in the real world credit data set, misclassification patterns from the input samples generally restricted the utility of the classifier built. The key idea of this study is to category a newly acquired applicant not only as good or bad, but also as borderline credit status based on the initial information provided by the applicant. For this reason, we introduce class-wise classification as a preprocessing step in order to obtain more homogeneous cluster groups and avoid misclassification patterns. Moreover, there exist many classifier algorithms for credit scoring analysis. For constructing more efficient classifiers, some other approaches have been proposed to combine different classifiers into an ensemble classifier. An important issue in constructing an ensemble classifier is to make each individual classifier as different from the other classifiers as possible. In our approach, we first constructed the individual NN, BN and SVM classifiers by using class-wise bagging as a data augmentation strategy to obtain good generalization performance. Then, the final outputs were decided by a confidence-weighted voting ensemble strategy.

From the experimental results, we concluded that the proposed ensemble approach gave significantly better performance than the conventional ensemble classifiers. The ensemble approach exploits the differences in misclassification by individual classifiers and improves the overall performance. Specifically, optimal associate binning for the discretization of continuous features and the Markov blanket concept of BN allowed for a useful strategy of feature selection, which provided a basis for mining association rules. The learned knowledge was then represented in multiple forms including causal diagram and constrained association rules. The data-driven nature distinguished the proposed system from those existing credit scoring systems based on hybrid/ensemble of various classifiers. To summarize, the advantage of using the proposed multi-classifier system is that decision makers can have practical aid in their daily credit approval task with relatively high accuracy, and decision makers can figure out meaningful relationships among features by causality networks and constrained association rules.

References

1. Altman, E.: Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy. Journal of Finance 23, 589–609 (1968)
2. Lawrence, E., Arshadi, N.: A Multinomial Logit Analysis of Problem Loan Resolution Choices in Banking. Journal of Money, Credit and Banking 27, 202–216 (1995)
3. Charitou, A., Neophytou, E., Charalambous, C.: Predicting Corporate Failure: Empirical Evidence for the UK. European Accounting Review 13(3), 465–497 (2004)
4. McKee, T.E.: Rough Sets Bankruptcy Prediction Models Versus Auditor Signaling Rates. Journal of Forecasting 22, 569–586 (2003)
5. Sarkar, S., Sriram, R.S.: Bayesian Models for Early Warning of Bank Failures. Management Science 47(11), 1457–1475 (2001)
6. Tsai, C.F., Wu, J.W.: Using Neural Network Ensembles for Bankruptcy Prediction and Credit Scoring. Expert Systems with Applications 34, 2639–2649 (2008)

7. West, D., Dellana, S., Qian, J.: Neural Network Ensemble Strategies for Financial Decision Applications. Computers and Operations Research 32, 2543–2559 (2005)
8. Huang, C.L., Chen, M.C., Wang, C.J.: Credit Scoring with a Data Mining Approach Based on Support Vector Machines. Expert Systems with Applications 33, 847–856 (2007)
9. Kerber, R.: Chimerge: Discretization of Numeric Attributes. In: Proceedings of the 10th National Conference on Artificial Intelligence, pp. 123–128. MIT Press, Cambridge (1992)
10. Fayyad, U.M., Irani, K.B.: Multi-interval Discretization of Continuous-Valued Attributes for Classification Learning. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence, Machine Learning, pp. 1022–1027. Morgan Kaufmann, Chambery (1993)
11. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and Unsupervised Discretization of Continuous Features. In: Prieditis, A., Russell, S. (eds.) Proceedings of the 12th International Conference on Machine Learning, pp. 194–202. Morgan Kaufmann, San Francisco (1995)
12. Sun, L., Shenoy, P.P.: Using Bayesian Networks for Bankruptcy Prediction: Some Methodological Issues. European Journal of Operational Research 180, 738–753 (2007)
13. Oza, N.C., Tumer, K.: Classifier Ensembles: Select Real-World Applications. Information Fusion 9, 4–20 (2008)
14. Hsieh, N.C.: Hybrid Mining Approach in the Design of Credit Scoring Models. Expert Systems with Applications 28, 655–665 (2005)
15. Punj, G., Steward, D.W.: Cluster Analysis in Marketing Research: Review and Suggestions for Applications. Journal of Marketing Research 20, 134–148 (1983)
16. Kittler, J., Hatef, M., Duin, R.W., Matas, J.: On Combining Classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence 20, 226–239 (1998)
17. Nilson, N.J.: Learning Machines: Foundations of Trainable Pattern Classifiers. McGraw-Hill, New York (1965)
18. Islam, M.M., Yao, X., Murase, K.: A Constructive Algorithm for Training Cooperative Neural Network Ensembles. IEEE Transactions on Neural Networks 14, 820–834 (2003)
19. Salchenberger, L.M., Cianr, E.M., Lash, N.A.: Neural Networks: a New Tool for Predicting Thrift Failures. Decision Sciences 23, 899–916 (1992)
20. Jaroszewicz, S., Simovici, D.A.: Interestingness of Frequent Itemsets Using Bayesian Networks as Background Knowledge. In: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 178–186. ACM, Seattle (2004)
21. Fauré, C., Delprat, S., Boulicaut, J.-F., Mille, A.: Iterative Bayesian Network Implementation by Using Annotated Association Rules. In: Proceedings of the 15th International Conference on Knowledge Engineering and Knowledge Management, pp. 326–333. Springer, Podebrady (2006)

A Multi-partition Multi-chunk Ensemble Technique to Classify Concept-Drifting Data Streams

Mohammad M. Masud[1], Jing Gao[2], Latifur Khan[1], Jiawei Han[2], and Bhavani Thuraisingham[1]

[1] Department of Computer Science, University of Texas at Dallas
[2] Department of Computer Science, University of Illinois at Urbana-Champaign

Abstract. We propose a multi-partition, multi-chunk ensemble classifier based data mining technique to classify concept-drifting data streams. Existing ensemble techniques in classifying concept-drifting data streams follow a single-partition, single-chunk approach, in which a single data chunk is used to train one classifier. In our approach, we train a collection of v classifiers from r consecutive data chunks using v-fold partitioning of the data, and build an ensemble of such classifiers. By introducing this multi-partition, multi-chunk ensemble technique, we significantly reduce classification error compared to the single-partition, single-chunk ensemble approaches. We have theoretically justified the usefulness of our algorithm, and empirically proved its effectiveness over other state-of-the-art stream classification techniques on synthetic data and real botnet traffic.

1 Introduction

Data stream classification is a major challenge to the data mining community. There are two key problems related to stream data classification. First, it is impractical to store and use all the historical data for training, since it would require infinite storage and running time. Second, there may be concept-drift in the data. The solutions to these two problems are related. If there is a concept-drift in the data, we need to refine our hypothesis to accommodate the new concept. Thus, most of the old data must be discarded from the training set. Therefore, one of the main issues in mining concept-drifting data streams is to choose the appropriate training instances to learn the evolving concept.

One approach is to select and store the training data that are most consistent with the current concept [1]. Some other approaches update the existing classification model when new data appear, such as the Very Fast Decision Tree (VFDT) [2] approach. Another approach is to use an ensemble of classifiers and update the ensemble every time new data appear [3,4]. As shown in [3,4], the ensemble classifier is often more robust at handling unexpected changes and concept drifts. We propose a multi-partition, multi-chunk ensemble classification algorithm, which is a generalization over the existing ensemble methods and can improve the classification accuracy significantly.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 363–375, 2009.
© Springer-Verlag Berlin Heidelberg 2009

We assume that the data stream is divided into equal sized *chunks*. The chunk size is chosen so that all data in a chunk may fit into the main memory. Each chunk, when labeled, is used to train classifiers. In our approach, there are three parameters that control the multi-partition, multi-chunk ensemble: v, r, and K. Parameter v determines the number of partitions (v=1 means single-partition ensemble), parameter r determines the number of chunks ($r = 1$ means single-chunk ensemble), and parameter K controls the ensemble size. Our ensemble consists of $K * v$ classifiers. This ensemble is updated whenever a new data chunk is labeled. We take the most recent *labeled* r consecutive data chunks and train v classifiers using v-fold partitioning of these chunks. We then update the ensemble by choosing the best (based on accuracy) $K * v$ classifiers among the newly trained v classifiers and the existing $K * v$ classifiers. Thus, the total number of classifiers in the ensemble is always kept constant.

It should be noted that when a new data point appears in the stream, it may not be labeled immediately. We defer the ensemble updating process until the data points in the latest data chunk have been labeled, but we keep classifying new unlabeled data using the current ensemble. For example, consider the online credit-card fraud detection problem. When a new credit-card transaction takes place, its class ({fraud,authentic}) is predicted using the current ensemble. Suppose a fraudulent transaction has been miss-classified as "authentic". When the customer receives the bank statement, he will identify this error and report to the authority. In this way, the actual labels of the data points will be obtained, and the ensemble will be updated accordingly.

We have several contributions. First, we propose a generalized multi-partition, multi-chunk ensemble technique that significantly reduces the expected classification error over the existing single-partition, single-chunk ensemble methods. Second, we have theoretically justified the effectiveness of our approach. Finally, we apply our technique on synthetically generated data as well as on real botnet traffic, and achieve better detection accuracies than other stream data classification techniques. We believe that the proposed ensemble technique provides a powerful tool for data stream classification.

The rest of the paper is organized as follows: section 2 discusses related works, section 3 discusses the classification algorithm and proves its effectiveness, section 4 discusses data collection, experimental setup, evaluation techniques, and results, and section 5 concludes with directions to future work.

2 Related Work

There have been many works in stream data classification. There are two main approaches - single model classification, and ensemble classification. Single model classification techniques incrementally update their model with new data to cope with the evolution of stream [2,5,6,7]. These techniques usually require complex operations to modify the internal structure of the model. Besides, in these algorithms, only the most recent data is used to update the model. Thus, contributions of historical data are forgotten at a constant rate even if some of the

historical data are consistent with the current concept. So, the refined model may not appropriately reflect the current concept, and its prediction accuracy may not meet the expectation.

In a non-streaming environment, ensemble classifiers like Boosting [8] are popular alternatives to single model classifiers. But these are not directly applicable to stream mining. However, several ensemble techniques for data stream mining have been proposed [3,4,9,10]. These ensemble approaches have the advantage that they can be more efficiently built than updating a single model and they observe higher accuracy than their single model counterparts [11]. Among these approaches, our ensemble approach is related to that of Wang et al [3]. Wang et al. [3] keep an ensemble of the K best classifiers. Each time a new data chunk appears, a classifier is trained from that chunk. If this classifier shows better accuracy than any of the K classifiers in the ensemble, then the new classifier replaces the old one. When classifying an instance, weighted voting among the classifiers in the ensemble is taken, where the weight of a classifier is inversely proportional to its error.

There are several differences between our approach and the approach of Wang et al. First, we apply multi-partitioning of the training data to build multiple (i.e., v) classifiers from that training data. Second, we train each classifier from r consecutive data chunks, rather than from a single chunk. Third, when we update the ensemble, the v classifiers that are removed may come from different chunks; thus, although some classifiers from a chunk may have been removed, other classifiers from that chunk may still remain in the ensemble. Whereas, in the approach of Wang et al., removal of a classifier means total removal of the knowledge obtained from one whole chunk. Finally, we use simple voting, rather than weighted voting. Thus, our multi-partition, multi-chunk ensemble approach is a generalized form of the approach of Wang et al., and users have more freedom to optimize performance by choosing the appropriate values of these two parameters (i.e., r and v).

3 An Ensemble Built on Multiple Partitions of Multiple Chunks (MPC)

We keep an ensemble $A = \{A_1, A_2, ..., A_{K*v}\}$ of the most recent best $K * v$ classifiers. Each time a new data chunk D_n arrives, we test the data chunk with the ensemble A. We update the ensemble when D_n is labeled. This ensemble training process is illustrated in figure 1 and explained in section 3.1. We will refer to our ensemble approach as the "Multiple-Partition Multiple-Chunk (MPC)" ensemble approach. The ensemble classification process uses simple majority voting. Section 3.2 explains how MPC ensemble reduces classification error over other approaches.

3.1 Ensemble Updating Algorithm

Description of the algorithm (algorithm 1): Let D_n be the most recent data chunk that has been labeled. In lines 1-3 of the algorithm, we compute the error of each

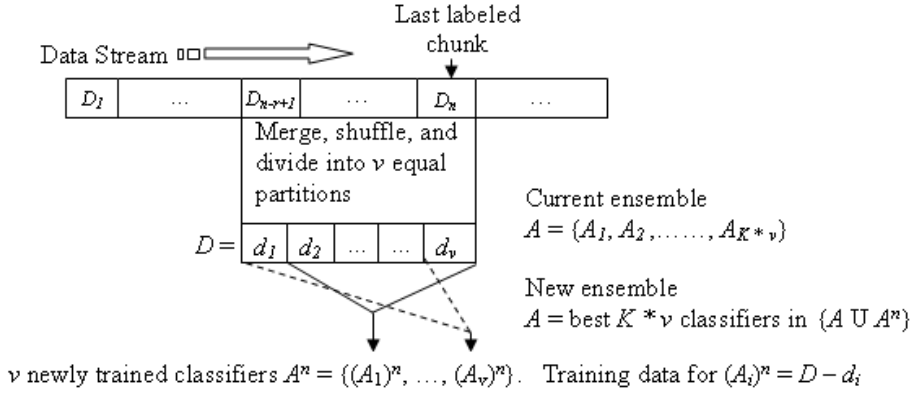


Fig. 1. Illustration: how data chunks are used to build an ensemble with MPC

classifier $A_i \in A$ on D_n. Let D=$\{D_{n-r+1},...,D_n\}$, i.e, the most recently labeled r data chunks including D_n. In line 5, we randomly divide D into v equal parts = $\{d_1, ..., d_v\}$, such that roughly, all the parts have the same class distributions. In lines 6-9, we train a new batch of v classifiers, where each classifier A_j^n is trained with the dataset D - $\{d_j\}$. We compute the expected error of each classifier A_j^n on its corresponding test data d_j. Finally, on line 10, we select the best $K * v$ classifiers from the $K * v + v$ classifiers $A^n \cup A$. Note that any subset of the n^{th} batch of v classifiers may take place in the new ensemble.

Algorithm 1. Updating the classifier ensemble

Input: $\{D_{n-r+1}, ..., D_n\}$: most recently labeled r data chunks
A: Current ensemble of best $K * v$ classifiers
Output: Updated ensemble A
1: **for** each classifier $A_i \in A$ **do**
2: Test A_i on D_n and compute its expected error
3: **end for**
4: Let $D = \{D_{n-r+1} \cup D_n\}$
5: Divide D into v equal disjoint partitions $\{d_1, d_2, ..., d_v\}$
6: **for** j=1 to v **do**
7: $A_j^n \leftarrow$ Train a classifier with training data D-d_j
8: Test A_j^n on its test data d_j and compute its expected error
9: **end for**
10: $A \leftarrow$ best $K * v$ classifiers from $A^n \cup A$ based on expected error

3.2 Error Reduction Using Multi-Partition and Multi-Chunk Training

As explained in algorithm 1, we build an ensemble of $K * v$ classifiers A. A test instance x is classified using a majority voting of the classifiers in the ensemble. We use simple majority voting, as opposed to weighted majority voting used

in [3], since simple majority voting is theoretically proven to be the optimal choice [10]. Besides, in our experiments, we also obtain better results with simple voting. In the next few paragraphs, we show that MPC can significantly reduce the expected error in classifying concept-drifting data streams compared to other approaches that use only one data chunk for training a single classifier (i.e., $r = 1$, $v = 1$), which will be referred to henceforth as "Single-Partition Single-Chunk (SPC)" ensemble approach.

Given an instance x, the posterior probability distribution of class a is $p(a|x)$. For a two-class classification problem, a=+ or −. According to Tumer and Ghosh [11], a classifier is trained to learn a function $f^a(.)$ that approximates this posterior probability: $f^a(x) = p(a|x)+\eta^a(x)$, where $\eta^a(x)$ is the error of $f^a(x)$ relative to $p(a|x)$. This is the error in addition to Bayes error and usually referred to as the "added error". This error occurs either due to the bias of the learning algorithm, and/or the variance of the learned model. According to [11], the expected added error can be obtained from the following formula: $Error = \frac{\sigma^2_{\eta^a(x)}}{s}$ where $\sigma^2_{\eta^a(x)}$ is the variance of $\eta^a(x)$, and s is the difference between the derivatives of $p(+|x)$ and $p(-|x)$, which is independent of the learned classifier.

Let $C = \{C_1, ..., C_K\}$ be an ensemble of K classifiers, where each classifier C_i is trained from a single data chunk (i.e., C is an SPC ensemble). If we average the outputs of the classifiers in a K-classifier ensemble, then according to [11], the ensemble output would be: $f^a_C = \frac{1}{K}\sum_{i=i}^{K} f^a_{C_i}(x) = p(a|x) + \eta^a_C(x)$, where f^a_C is the output of the ensemble C, $f^a_{C_i}(x)$ is the output of the ith classifier C_i, and $\eta^a_C(x)$ is the average error of all classifiers, given by:
$\eta^a_C(x) = \frac{1}{K}\sum_{i=1}^{K} \eta^a_{C_i}(x)$, where $\eta^a_{C_i}(x)$ is the added error of the ith classifier in the ensemble. Assuming the error variances are independent, the variance of $\eta^a_C(x)$ is given by:

$$\sigma^2_{\eta^a_C(x)} = \frac{1}{K^2}\sum_{i=1}^{K} \sigma^2_{\eta^a_{C_i}(x)} = \frac{1}{K}\bar{\sigma}^2_{\eta^a_C(x)} \tag{1}$$

where $\sigma^2_{\eta^a_{C_i}(x)}$ is the variance of $\eta^a_{C_i}(x)$, and $\bar{\sigma}^2_{\eta^a_C(x)}$ is the common variance. In order to simplify the notation, we would denote $\sigma^2_{\eta^a_{C_i}(x)}$ with $\sigma^2_{C_i}$.

Let A be the the ensemble of $K * v$ classifiers $\{A_1, A_2, ..., A_{K*v}\}$,where each A_i is a classifier trained using r consecutive data chunks (i.e., the MPC approach). The following lemma proves that MPC reduces error over SPC by a factor of rv when the outputs of the classifiers in the ensemble are independent.

Lemma 1. *Let σ^2_C be the error variance of SPC. If there is no concept-drift, and the errors of the classifiers in the ensemble A are independent, then the error variance of MPC is $1/rv$ times of that of SPC, i.e., $\sigma^2_A = \frac{1}{rv}\sigma^2_C$.*

Proof. Each classifier $A_i \in A$ is trained on r consecutive data chunks. If there is no concept-drift, then a classifier trained on r consecutive data chunks may reduce the error of the single classifiers trained on a single data chunk by a factor of r [3]. So, it follows that:

$$\sigma^2_{A_i} = \frac{1}{r^2} \sum_{j=i}^{r+i-1} \sigma^2_{C_j} \tag{2}$$

where $\sigma^2_{A_i}$ is the error variance of classifier A_i, trained using data chunks $\{D_i \cup D_{i+1} ... \cup D_{i+r-1}\}$ and $\sigma^2_{C_j}$ is the error variance of C_j, trained using a single data chunk D_j. Combining equations 1 and 2 and simplifying, we get:

$$\begin{aligned} \sigma^2_A &= \frac{1}{K^2v^2} \sum_{i=1}^{Kv} \frac{1}{r^2} \sum_{j=i}^{r+i-1} \sigma^2_{C_j} \qquad (3) \\ &= \frac{1}{K^2v^2r^2} \sum_{i=1}^{Kv} \sum_{j=i}^{r+i-1} \sigma^2_{C_j} = \frac{1}{K^2v^2r} \sum_{i=1}^{Kv} (\frac{1}{r} \sum_{j=i}^{r+i-1} \sigma^2_{C_j}) = \frac{1}{K^2v^2r} \sum_{i=1}^{Kv} \bar{\sigma}^2_{C_i} \\ &= \frac{1}{Krv} (\frac{1}{Kv} \sum_{i=1}^{Kv} \bar{\sigma}^2_{C_i}) = \frac{1}{Krv} \bar{\sigma}^2_C = \frac{1}{rv} (\frac{1}{K} \bar{\sigma}^2_C) = \frac{1}{rv} \sigma^2_C \end{aligned}$$

where $\bar{\sigma}^2_{C_i}$ is the common variance of $\sigma^2_{C_j}, i \leq j \leq i + r - 1$, and $\bar{\sigma}^2_C$ is the common variance of $\bar{\sigma}^2_{C_i}, 1 \leq i \leq Kv$. □

However, since we train v classifiers from each r consecutive data chunks, the independence assumption given above may not be valid since each pair of these v classifiers have overlapping training data. We need to consider correlation among the classifiers to compute the expected error reduction. The following lemma shows the error reduction considering error correlation.

Lemma 2. *Let σ^2_C be the error variance of SPC. If there is no concept-drift, then the error variance of MPC is $(v-1)/rv$ times of that of SPC, i.e., $\sigma^2_A = \frac{v-1}{rv}\sigma^2_C, v > 1$. Proof: see [12].*

For example, if v=2, and r=2 then we can have an error reduction by a factor of 4. However, if there is concept-drift, then the assumption in equation 2 may not be valid. In order to analyze error in presence of concept-drift, we introduce a new term "Magnitude of drift" or ρ_d.

Definition 1. *Magnitude of drift or ρ_d is the is the maximum error introduced to a classifier due to concept-drift. That is, every time a new data chunk appears, the error variance of a classifier is incremented ρ_d times due to concept-drift.*

For example, let $D_j, j \in \{i, i+1, ..., i+r-1\}$ be a data chunk in a window of r consecutive chunks $\{D_i, ..., D_{i+r-1}\}$ and C_j be the classifier trained with D_j. Let the actual error variance of C_j in the presence of concept-drift is $\hat{\sigma}^2_{C_j}$, and the error variance of C_j in the absence of concept-drift is $\sigma^2_{C_j}$. Then we have:

$$\hat{\sigma}^2_{C_j} = (1 + \rho_d)^{(i+r-1)-j} \sigma^2_{C_j} \tag{4}$$

In other words, $\hat{\sigma}^2_{C_j}$ is the actual error variance of the jth classifier C_j in the presence of concept-drift, when the last data chunk in the window, D_{i+r-1} appears. Our next lemma deals with error reduction in the presence of concept-drift.

Lemma 3. *Let $\hat{\sigma}_A^2$ be the error variance of MPC in the presence of concept-drift, σ_C^2 be the error variance of SPC, and ρ_d be the drifting magnitude defined by definition 1. Then $\hat{\sigma}_A^2$ is bounded by: $\hat{\sigma}_A^2 \leq \frac{(v-1)(1+\rho_d)^{r-1}}{rv}\sigma_C^2$.*

Proof. Replacing $\sigma_{C_j}^2$ with $\hat{\sigma}_{C_j}^2$ in equation 3, and using equation 4 and lemma 2, we get

$$\hat{\sigma}_A^2 = \frac{v-1}{K^2v^2}\sum_{i=1}^{Kv}\frac{1}{r^2}\sum_{j=i}^{r+i-1}\hat{\sigma}_{C_j}^2 = \frac{v-1}{K^2r^2v^2}\sum_{i=1}^{Kv}\sum_{j=i}^{r+i-1}(1+\rho_d)^{(i+r-1)-j}\sigma_{C_j}^2$$

$$\leq \frac{v-1}{K^2r^2v^2}\sum_{i=1}^{Kv}(1+\rho_d)^{r-1}\sum_{j=i}^{r+i-1}\sigma_{C_j}^2 = \frac{(v-1)(1+\rho_d)^{r-1}}{K^2r^2v^2}\sum_{i=1}^{Kv}\sum_{j=i}^{r+i-1}\sigma_{C_j}^2$$

$$= \frac{(v-1)(1+\rho_d)^{r-1}}{Krv}\bar{\sigma}_C^2 = \frac{(v-1)(1+\rho_d)^{r-1}}{rv}\sigma_C^2, r > 0 \qquad \square$$

Therefore, we would achieve a reduction of error provided that

$$\frac{(v-1)(1+\rho_d)^{r-1}}{rv} \leq 1 \qquad \text{or, } E_R \leq 1 \tag{5}$$

Where E_R is the ratio of MPC error to SPC error in the presence of concept-drift. As we increase r and v, the relative error keeps decreasing upto a certain point. After that, it becomes flat or starts increasing. Next, we analyze the effect of parameters r and v on error reduction, in the presence of concept-drift.

3.3 Upper Bounds of *r* and *v*

For a given value of v, r can only be increased up to a certain value. After that, increasing r actually hurts the performance of our algorithm, because inequality 5 is violated. The upper bound of r depends on ρ_d, the magnitude of drift. Although it may not be possible to know the actual value of ρ_d from the data, we may determine the optimal value of r experimentally. In our experiments, we found that for smaller chunk-sizes, higher values of r work better, and vice versa. However, the best performance-cost trade-off is found for r=2 or 3. We have used r=2 in our experiments. Similarly, the upper bound of v can be found from the inequality 5 for a fixed value of r. It should be noted that if v is increased, running time also increases. From our experiments, we obtained the best performance-cost trade-off for v=5.

3.4 Time Complexity of MPC

Let m be the size of the data stream and n be the total number of data chunks. Then our time complexity is $O(Kvm + nvf(rm/n))$, where $f(z)$ is the time to build a classifier on a training data of size z. Since v is constant, the complexity becomes $O(Km+nf(rm/n))$ =$O(n.(Ks+f(rs))$, where s is the size of one data chunk. It should be mentioned here that the time complexity of the approach by Wang et al. [3] is $O(n.(Ks + f(s))$. Thus, the actual running time of MPC would be at most a constant factor (rv times) higher than that of Wang et al [3]. But at the same time, we also achieve significant error reduction.

4 Experiments

We evaluate our proposed method on both synthetic data and botnet traffic generated in a controlled environment, and compare with several baseline methods.

4.1 Data Sets and Experimental Setup

Synthetic dataset: Synthetic data are generated with drifting concepts [3]. Concept-drifting data can be generated with a moving hyperplane. The equation of a hyperplane is as follows: $\sum_{i=1}^{d} a_i x_i = a_0$. If $\sum_{i=1}^{d} a_i x_i \leq a_0$, then an example is negative, otherwise it is positive. Each example is a randomly generated d-dimensional vector $\{x_1, ..., x_d\}$, where $x_i \in [0, 1]$. Weights $\{a_1, ..., a_d\}$ are also randomly initialized with a real number in the range [0, 1]. The value of a_0 is adjusted so that roughly the same number of positive and negative examples are generated. This can be done by choosing $a_0 = \frac{1}{2} \sum_{i=1}^{d} a_i$ We also introduce noise randomly by switching the labels of $p\%$ of the examples, where p=5 is used in our experiments. There are several parameters that simulate concept-drift. We use the same parameters settings as in [3] to generate synthetic data. We generate a total of 250,000 records and generate four different datasets having chunk sizes 250, 500, 750, and 1000, respectively. The class distribution of these datasets is: 50% positive and 50% negative.

Real (botnet) dataset: Botnet is a network of compromised hosts or *bots*, under the control of a human attacker known as the *botmaster* [13]. The botmaster can issue commands to the bots to perform malicious actions, such as launching DDoS attacks, spamming, spying and so on. Thus, botnets have appeared as enormous threat to the internet community. Peer-to-Peer(P2P) is the new emerging technology of botnets. These botnets are distributed, and small. So, they are hard to detect and destroy. Examples of P2P bots are Nugache [15], Sinit [16], and Trojan.Peacomm [17].

Botnet traffic can be considered as a data stream having both properties: infinite length and concept-drift. So, we apply our stream classification technique to detect P2P botnet traffic. We generate real P2P botnet traffic in a controlled environment, where we run a P2P bot named Nugache [15]. The details of the feature extraction process are discussed in [12]. There are 81 continuous attributes in total. The whole dataset consists of 30,000 records, representing one week's worth of network traffic. We generate four different datasets having chunk sizes of 30 minutes, 60 minutes, 90 minutes, and 120 minutes, respectively. The class distribution of these datasets is: 25% positive (botnet traffic) and 75% negative (benign traffic).

Baseline methods: For classification, we use the "Weka" machine learning open source package, available at "http://www.cs.waikato.ac.nz/ml/weka/". We apply three different classifiers - J48 decision tree, Ripper, and Bayes Net. In order to compare with other techniques, we implement the followings:

***MPC*:** This is our multi-partition, multi-chunk (MPC) ensemble algorithm.

***BestK*:** This is a single-partition, single-chunk (SPC) ensemble approach, where an ensemble of the best K classifiers is used. Here K is the ensemble size. This ensemble is created by storing all the classifiers seen so far, and selecting the best K of them based on expected error. An instance is tested using simple voting.

***Last*:** In this case, we only keep the last trained classifier, trained on a single data chunk. It can be considered a SPC approach with $K = 1$.

***Wang*:** This is an SPC method implemented by Wang et al. [3].

All: This is also an SPC approach. In this case, we create an ensemble of all the classifiers seen so far, and the new data chunk is tested with this ensemble by simple voting among the classifiers.

4.2 Performance Study

In this section, we compare the results of all the five techniques, MPC, $Wang$, $BestK$, All and $Last$. As soon as a new data chunk appears, we test each of these ensembles /classifiers on the new data, and update its accuracy, false positive, and false negative rate. In all the results shown here, we fix the parameter values of v=5, and r=2, unless mentioned otherwise.

Figure 2(a) shows the error rates for different values of K of each method, averaged over four different chunk sizes on synthetic data, and figure 2(c) shows the same for botnet data. Here decision tree is used as the base learner. It is evident that MPC has the lowest error among all approaches. Besides, we observe that the error of MPC is lower for higher values of K. This is desired because higher values of K means larger ensemble, and more error reduction. However, accuracy does not improve much after $K = 8$. $Wang$ and $BestK$ also show similar characteristic. All and $Last$ do not depend on K, so their error remains the same for any K. Figure 2(b) shows the error rates for four different chunk sizes of each method (also using decision tree) averaged over different values of K (2,4,6,8) on synthetic data, and figure 2(d) shows the same for botnet data. Again, MPC has the lowest error of all. Besides, the error of MPC

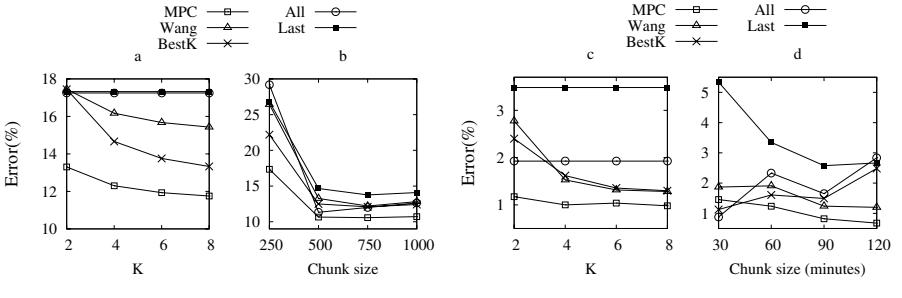


Fig. 2. Error vs K and chunk size on synthetic data (a,b) and botnet data (c,d)

Table 1. Error of different approaches on synthetic data

(a) using decision tree

Chunk size	M_2	W_2	B_2	M_4	W_4	B_4	M_6	W_6	B_6	M_8	W_8	B_8	*All*	*Last*
250	**19.3**	26.8	26.9	**17.3**	26.5	22.1	**16.6**	26.3	20.4	**16.2**	26.1	19.5	29.2	26.8
500	**11.4**	14.8	14.7	**10.6**	13.2	12.4	**10.3**	12.7	11.6	**10.2**	12.4	11.3	11.3	14.7
750	**11.1**	13.9	13.9	**10.6**	12.1	11.9	**10.3**	11.5	11.4	**10.3**	11.3	11.2	15.8	13.8
1000	**11.4**	14.3	14.3	**10.7**	12.8	12.2	**10.5**	12.2	11.7	**10.3**	11.9	11.4	12.6	14.1

(b) using Ripper

Chunk size	M_2	W_2	B_2	M_4	W_4	B_4	M_6	W_6	B_6	M_8	W_8	B_8	*All*	*Last*
250	**19.2**	26.5	26.0	**17.6**	26.2	22.4	**17.1**	26.0	21.3	**16.8**	25.9	20.9	30.4	26.3
500	**11.5**	14.2	13.9	**10.8**	13.0	12.3	**10.6**	12.6	11.8	**10.5**	12.5	11.5	11.6	14.1
750	**11.0**	13.4	13.3	**10.6**	12.1	12.0	**10.5**	11.7	11.6	**10.5**	11.5	11.5	15.7	13.3
1000	**11.1**	13.8	13.7	**10.6**	12.5	12.3	**10.3**	12.1	11.9	**10.2**	11.9	11.8	12.6	13.6

is lower for larger chunk sizes. This is desired because larger chunk size means more training data for a classifier.

Tables 1(a) and 1(b) report the error of decision tree and Ripper learning algorithms, respectively, on synthetic data, for different values of K and chunk sizes. We do not show the results of Bayes Net, which have the similar characteristics, due to space limitation. The columns denoted by M_2, W_2 and B_2 represent MPC, $Wang$ and $BestK$, respectively, for K=2. Other columns have similar interpretations. In all three tables, we see that MPC has the lowest error for all values of K (shown in bold).

Figure 3 shows the sensitivity of r and v on error and running times on synthetic data for MPC. Figure 3(a) shows the errors for different values of r for a fixed value of v (=5) and K (=8). The highest reduction in error occurs when r is increased from 1 to 2. Note that $r = 1$ means single chunk training. We observe no significant reduction in error for higher values of r, which follows from our analysis of parameter r on concept-drifting data in section 3.3. However, the running time keeps increasing, as shown in figure 3(c). The best trade-off between running time and error occurs for r=2. The charts in figures 3(b,d) show a similar trend for parameter v. Note that $v = 1$ is the base case, i.e., the single partition ensemble approach, and $v > 1$ is the multi-partition ensemble approach. We observe no real improvement after $v = 5$, although the running

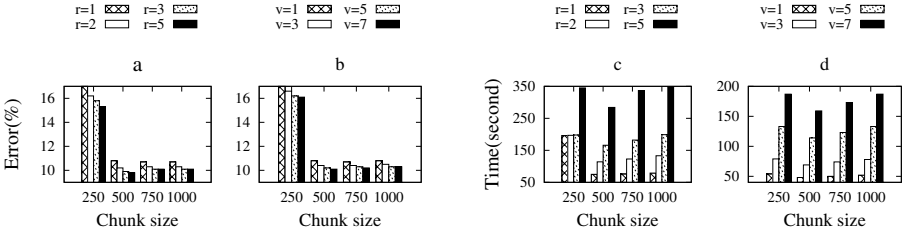


Fig. 3. Sensitivity of parameters r and v: on error (a,b), and running time (c,d)

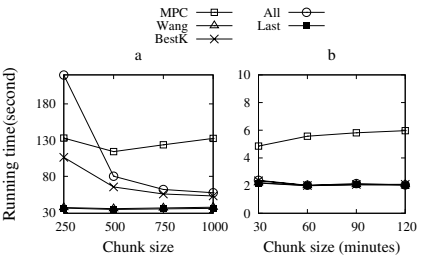


Fig. 4. Total running times on (a) synthetic data, and (b) real data

time keeps increasing. This result is also consistent with our analysis of the upper bounds of v, explained in section 3.3. We choose v=5 as the best trade-off between time and error.

Figure 4(a) shows the total running times of different methods on synthetic data for $K = 8, v = 5$ and $r = 2$. Note that the running time of MPC is within 5 times of that of $Wang$. This also supports our complexity analysis that the running time of MPC would be at most rv times the running time of $Wang$. The running times of MPC on botnet data shown in figure 4(b) also have similar characteristics. The running times shown in figure 4 include both training and testing time. Although the total training time of MPC is higher than that of $Wang$, the total testing times are almost the same in both techniques. Considering that training can be done offline, we may conclude that both these techniques have the same runtime performances in classifying data streams. Besides, users have the flexibility to choose either better performance or shorter training time just by changing the parameters r and v.

We also report the results of using equal number of classifiers in MPC and $Wang$ by setting $K = 10$ in $Wang$, and K=2, v=5, and r=1 in MPC, which is shown in Table 2. We observe that error of MPC is lower than that of $Wang$ in all chunk sizes. The columns M_2(J48), and W_{10}(J48) show the error of MPC (K=2,v=5,$r = 1$) and $Wang$ (K=10), respectively, for decision tree algorithm. The columns M_2(Ripper), and W_{10}(Ripper) show the same for Ripper algorithm. For example, for chunk size 250, and decision tree algorithm, MPC error is 19.9%, whereas $Wang$ error is 26.1%. We can draw two important conclusions from this result. First, if the ensemble size of $Wang$ is simply increased v times (i.e., made equal to $K * v$), its error does not become as low as MPC. Second, even if we use the same training set size in both these methods (i.e., r=1), error of $Wang$ still remains higher than that of MPC. There are two possible reasons

Table 2. Error comparison with the same number of classifiers in the ensemble

Chunk size	M_2(J48)	W_{10}(J48)	M_2(Ripper)	W_{10}(Ripper)
250	**19.9**	26.1	**21.0**	26.1
500	**11.7**	12.5	**12.2**	12.6
1000	**11.4**	12.5	**11.8**	13.0

behind this performance. First, when a classifier is removed during ensemble updating in $Wang$, all information obtained from the corresponding chunk is forgotten, but in MPC, one or more classifiers from a chunk may survive. Thus, the ensemble updating approach in MPC tends to retain more information than that of $Wang$, leading to a better ensemble. Second, $Wang$ requires at least $K*v$ data chunks, whereas MPC requires at least $K+r-1$ data chunks to obtain $K*v$ classifiers. Thus, $Wang$ tends to keep much older classifiers in the ensemble than MPC, leading to some outdated classifiers that can put a negative effect on the ensemble outcome.

5 Conclusion

We have introduced a multi-partition multi-chunk ensemble method (MPC) for classifying concept-drifting data streams. Our ensemble approach keeps the best $K*v$ classifiers, where a batch of v classifiers are trained with v overlapping partitions of r consecutive data chunks. It is a generalization over previous ensemble approaches that train a single classifier from a single data chunk. By introducing this MPC ensemble, we have reduced error significantly over the single-partition, single-chunk approach. We have proved our claims theoretically, tested our approach on both synthetic data and real botnet data, and obtained better classification accuracies compared to other approaches. In the future, we would also like to apply our technique on the classification and model evolution of other real streaming data.

Acknowledgment

This research was funded in part by NASA grant NNX08AC35A and AFOSR under contract FA9550-06-1-0045.

References

1. Fan, W.: Systematic data selection to mine concept-drifting data streams. In: Proc. ACM SIGKDD, Seattle, WA, USA, pp. 128–137 (2004)
2. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proc. ACM SIGKDD, Boston, MA, USA, pp. 71–80. ACM Press, New York (2000)
3. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proc. SIGKDD, Washington, DC, USA, pp. 226–235 (2003)
4. Scholz, M., Klinkenberg., R.: An ensemble classifier for drifting concepts. In: Proc. Second International Workshop on Knowledge Discovery in Data Streams (IWKDDS), Porto, Portugal, pp. 53–64 (2005)
5. Gehrke, J., Ganti, V., Ramakrishnan, R., Loh, W.: Boat–optimistic decision tree construction. In: Proc. ACM SIGMOD, Philadelphia, PA, USA, pp. 169–180 (1999)
6. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proc. ACM SIGKDD, San Francisco, CA, USA, pp. 97–106 (2001)
7. Utgoff, P.E.: Incremental induction of decision trees. Machine Learning 4, 161–186 (1989)

8. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proc. International Conference on Machine Learning (ICML), Bari, Italy, pp. 148–156 (1996)
9. Kolter, J.Z., Maloof, M.A.: Using additive expert ensembles to cope with concept drift. In: Proc. International conference on Machine learning (ICML), Bonn, Germany, pp. 449–456 (2005)
10. Gao, J., Fan, W., Han, J.: On appropriate assumptions to mine data streams. In: Proc. IEEE International Conference on Data Mining (ICDM), Omaha, NE, USA, pp. 143–152 (2007)
11. Tumer, K., Ghosh, J.: Error correlation and error reduction in ensemble classifiers. Connection Science 8(304), 385–403 (1996)
12. Masud, M.M., Gao, J., Khan, L., Han, J., Thuraisingham, B.: Mining concept-drifting data stream to detect peer to peer botnet traffic. Univ. of Texas at Dallas Tech. Report# UTDCS-05-08 (2008), http://www.utdallas.edu/~mmm058000/reports/UTDCS-05-08.pdf
13. Barford, P., Yegneswaran, V.: An Inside Look at Botnets. In: Advances in Information Security. Springer, Heidelberg (2006)
14. Ferguson, T.: Botnets threaten the internet as we know it. ZDNet Australia (April 2008)
15. Lemos, R.: Bot software looks to improve peerage (2006), http://www.securityfocus.com/news/11390
16. Group, L.T.I.: Sinit p2p trojan analysis. lurhq (2004), http://www.lurhq.com/sinit.html
17. Grizzard, J.B., Sharma, V., Nunnery, C., Kang, B.B., Dagon, D.: Peer-to-peer botnets: Overview and case study. In: Proc. 1st Workshop on Hot Topics in Understanding Botnets, p. 1 (2007)

Parameter Estimation in Semi-Random Decision Tree Ensembling on Streaming Data

Peipei Li[1,2], Qianhui Liang[1], Xindong Wu[2,3], and Xuegang Hu[2]

[1] School of Information Systems, Singapore Management University, Singapore, 178902
[2] School of Computer Science and Information Technology, Hefei University of Technology, China, 230009
[3] Department of Computer Science, University of Vermont, USA, 05405

Abstract. The induction error in random tree ensembling results mainly from the strength of decision trees and the dependency between base classifiers. In order to reduce the errors due to both factors, a Semi-Random Decision Tree Ensembling (SRDTE) for mining streaming data is proposed based on our previous work on SRMTDS. The model contains semi-random decision trees that are independent in the generation process and have no interaction with each other in the individual decisions of classification. The main idea is to minimize correlation among the classifiers. We claim that the strength of decision trees is closely related to the estimation values of the parameters, including the height of a tree, the count of trees and the parameter of n_{min} in the Hoeffding Bounds. We analyze these parameters of the model and design strategies for better adaptation to streaming data. The main strategies include an incremental generation of sub-trees after seeing real training instances, a data structure for quick search and a voting mechanism for classification. Our evaluation in the 0-1 loss function shows that SRDTE has improved the performance in terms of predictive accuracy and robustness. We have applied SRDTE to e-business data streams and proved its feasibility and effectiveness.

Keywords: Random decision trees, data streams, parameter estimation.

1 Introduction

Streaming data are a key type of data in business applications, such as website shopping transactions, Internet search requests, and telephone call records. In these continuous, high-volume and open-ended data streams, it is a challenge to perform learning with traditional inductive models or algorithms[11][12][13].

An inductive model of Semi-Random Decision Tree Ensembling (SRDTE) is presented in this paper. We have built our work on SRMTDS[10], where attributes are selected randomly and a heuristic and semi-random strategy is adopted to estimate thresholds of numerical attributes and to generate multiple decision trees incrementally. There are three different features in the strategies compared with SRMTDS. Firstly, it does not grow children branches at one node in the tree generation phase until there are real training instances passed.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 376–388, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Secondly, a new Children-List data structure is designed for a quick search. These strategies are both more suitable for streaming data environments in terms of space and time. Lastly, SRDTE makes use of a voting mechanism with the majority class for classification to improve predictive accuracy. (In order to address these differences, we call the developed algorithm a new name of "SRDTE" instead of the used one "SRMTDS".)

Besides, in the analysis of the strength and correlation of classifiers, Breiman [3] pointed out that the generalization error for a model of random tree ensembling is dependent on the strength of the decision tree structure (i.e., the measure of classification accuracy of the classifier, denoted by s) and the mean correlation (or dependency, marked by $\bar{p}$) between the base classifiers, which is applicable to classifier based ensembling including random trees[5]. Due to the semi-randomized method adopted in SRDTE, the value of s is nondeterministic. Meanwhile, because the generation of a single tree is independent with each other, the immediate corollary is that the generalization error in SRDTE is almost only related to the factor of s, which is mostly impacted by the parameters of the initial height of tree-h_0, the count of trees-N and the minimum number of split-instances-n_{min} used in the Hoeffding Bounds inequality. To obtain their optimal values of these parameters for strengthening the abilities of SRDTE, extensive experiments and analysis have been done in order to take into account of their qualitative and quantitative relations, which is different from SRMTDS as well. The results demonstrate that SRDTE has a higher predictive accuracy estimated by the 0-1 loss function and performs better in time, space and robustness. To further validate its adaptability and feasibility oriented to real world data streams, we have collected on-line shopping data for mining implicit merchant relations and hidden action patterns. The performance study shows that SRDTE is efficient and effective, and provides a significant reference model of classification in real streaming data.

2 Semi-Random Decision Tree Ensembling (SRDTE)

Compared with SRMTDS, the model of SRDTE to be described in this section also creates N semi-random decision trees incrementally till an initial height of h_0. When the number of training examples satisfies a certain threshold, rescan leaves and change some of them with higher error rates of classification by decision nodes to further split. If the total space consumption of decision trees is over a predefined restriction, release spaces of leaves or nodes with numerical attributes, and cut off some sub-trees if necessary. In conclusion, its basic framework is illustrated as follows.

However, in contrast to SRMTDS, three diverse strategies in SRDTE are adopted, which are given in details respectively.

Incrementally Generate Decision Trees After Seeing Training Data Instead of growing branches blindly till up to h_0 in advance for SRMTDS, only the necessary branches are created in SRDTE after the real training data reach, whose details are shown in the function of *GenerateIncreTree*. More specifically,

Input: Training set: *DSTR*; Test set: *DSTE*; Attribute set: A; Initial height of tree: h_0; The number of minimum split-examples: n_{min}; Split estimator function: $H(\cdot)$; Tree count: N; The checked period: CP; The maximum memory consumption: MS;
Output: The error-rate of classification

Procedure SRDTE (*DSTR*, *DSTE*, A, h_0, n_{min}, $H(\cdot)$, N, CP, MS)

```
1. For each tree of T_k, k∈1, ..., N
2.     Generate the root of tree -T_k;
3.     For each training instance TR_i ∈ DSTR
4.         GenerateIncrTree( T_k, TR_i, h_0, n_min, A );
5.         If the count of observed training instances in T_k % CP == 0
6.             If the current total memory overhead of T_k < MS
7.                 Change some leaves into decision nodes;
8.             Else
9.                 Release spaces of leaves or some decision nodes;
10.    For each training instance TE_i ∈ DSTE
11.        Travel T_k tree from its root to a leaf;
12.        Classify with Majority class or Naïve Bayes in T_k;
13. Return VotingFunction();
```

an available training instance follows the branch of tree corresponding to the attribute's value in the instance and traverses from its root to a leaf or a node with a numerical attribute at Step (2). Once the passed node has no split-attribute, select an available attribute of A_j as the split-feature at Step (4). If A_j is discrete, mark the node as *discrete* and generate m_j^A +1 children nodes (m_j^A is the count of different attribute values of A_j); otherwise, denote the node as *numerical* and create only two children branches ("$<$" or "$\geq$" a cut-point value of A_j) in Step (5). Repeat this process for each child node till up to the height of h_0 at Steps (6)$\sim$(7). Otherwise, if the current node is a leaf, update the information associated with the leaf including the total number of training instances, the distributions of class labels and attribute values at Steps (9)$\sim$(10); If it is a *numerical* node without a cut-point, install the split-test to solve a split-threshold based on the heuristic method of Hoeffding Bound inequality [7][15] at Steps (11)$\sim$(13) after the statistic count of instances reaches n_{min}, where the feasibility and efficiency for streaming data in the Hoeffding Bound inequality have been verified in related efforts[6][1][10] and the definition is given below.

Consider a real-valued random variable r whose range is R. Suppose we have made n independent observations of this variable, and computed their mean $\bar{r}$, which shows that, with probability 1 - δ , the true mean of the variable is at least $\bar{r}$ - ε, i.e.,

$$P(r \geq \bar{r} - \varepsilon) = 1 - \delta, \varepsilon = \sqrt{R^2 ln(1/\delta)/2n} \tag{1}$$

where the minimum value of n is n_{min}, the value of R is set to the number of class labels of $\log_2(M(classes))$. Compute the information gain of candidate cut-point values of attribute A_j and suppose the difference of two candidate cut-points with the highest information $\Delta H = H(A_j(x))$ - $H(A_j(y))$. For a given τ

(the value of τ is lower than a standard value of ε in Formula (1). It implies that the top-two candidate cut-points with the highest information gain are either good or bad without obvious deviation, which is given as a *tie* bound of difference values), if $\Delta H > \varepsilon$ or $\Delta H \leq \varepsilon < \tau$, the attribute value of the x^{th} cut-point with the highest gain will be selected as a final split-point and more details are seen in [10].

Function GenerateIncreTree (T_k, TR_i, h_0, n_{min}, A, $H(\cdot)$)
(1)While the height of $T_k \leq h_0$ and the training data are not end
(2) Sort TR_i into a leaf-l_{curk} or a *numerical* node;
(3) If the height of l_{curk} is less than h_0 and a training data comes
(4) Select a splitting attribute $A_j \in A$ at l_{curk};
(5) Create children nodes by attribute values of A_j;
(6) For each child node
(7) Set this child node as the current node of l_{curk} and go to Step (1);
(8) Else
(9) If l_{curk} is a leaf with height of h_0
(10) Update the Information at this Leaf;
(11) If l_{curk} is a *numerical* node without a cut-point
(12) If the count of examples at this node $\geq n_{min}$
(13) Compute the split-point by the function of $H(\cdot)$;

The Structure of Children-List for Quick Search
For quick search of the target nodes, an additional data structure of Children-List (marked as *Clist*) is designed in SRDTE to point to different levels in one tree. For example, *Clist*[i] projects the i^{th} level in a semi-random decision tree, which binds all nodes in this level and records their information. It is applied into the process of scaling up the trees and releasing the node space at Steps 5∼9.

Voting Mechanism for Classification
After each semi-random tree has made its decision on test instances at Steps 10∼12, the judgment for final class labels is made with a majority-class voting or Naïve Bayes in Step 13, whose robustness for data streams has been demonstrated both theoretically and experimentally in [4]. Compute the sum for each class label in the tree ensembling and choose the class label with the maximum value as the one of the current test instance.

Besides, to achieve the optimal performance of SRDTE, the theorem of generation error mentioned above is worth emphasizing before we discuss the estimations of related parameters in the following sections, as shown in Formula (2).

$$GE \leq \bar{p}(1 - s^2)/s^2 = \bar{p}(1/s^2 - 1) \tag{2}$$

Evidently, the generalization error of GE in ensemble classifiers is directly proportional to $\bar{p}$ and is in inverse proportion to s. Though the voting mechanism is adopted in the classification, the generation process of each decision tree in

SRDTE is still independent of each other. Hence, the impact from $\bar{p}$ is much less, which is similar with that in SRMTDS. As regards the value of s, the parameters pertinent to the magnitude of trees refer to the ones in the inequality of Hoeffding Bounds, h_0 and N. In a similar analysis with SRMTDS, the probability ensuring the optimum of an ensemble model in SRDTE is more than $1-(1-1/M(Attrs))^{N \cdot 2^{h_0-1}}$[10], which indicates that the more the values of N and h_0, the less the generalization error is. Therefore, for the least value of GE, it is important to search for optimal parameter values.

3 Experiments

In this section, we first discuss the heuristic method to select the relevant parameter values, then analyze the performances of SRDTE based on optimal estimation values. Last, we apply SRDTE to real shopping data from Yahoo! website for further verification of its efficiency and effectiveness. In our experiments, we have also used several synthetic data sets from UCI[8] that are the representative ones with various types of attributes. Moreover, the different databases varying with the noise rates are generated by the LED data generator for the estimation on the robustness of SRDTE, where the training set of a varying number of examples starts from 100k to 1000k and the test set contains 250k examples. All experiments are performed on a P4, 3.00GHz PC with 1G main memory, running Windows XP Professional with the program platform of Visual C++ in a simulated parallel environment.

3.1 Parameter Estimation

Parameter Analysis for ε, δ and n_{min}

The Hoeffding Bound inequality adopted in SRDTE is applied to find better split thresholds at nodes with numerical attributes, which impacts the classification ability of individual trees. According to Formula 1, the smaller the value of ε, the

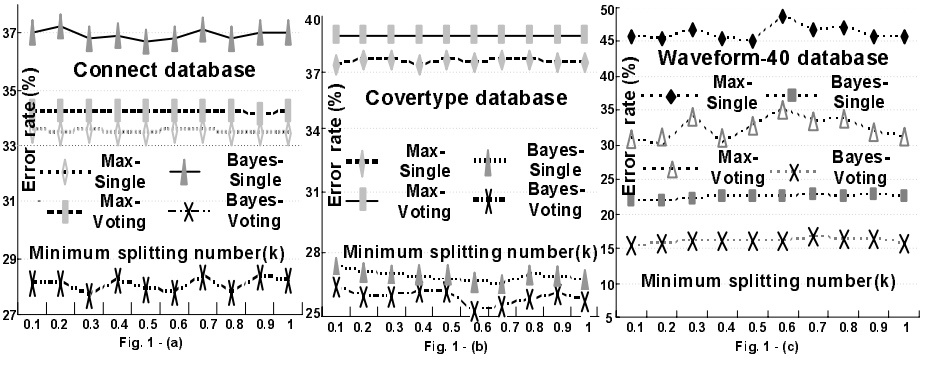


Fig. 1. Relation between n_{min} and error rates for UCI databases

more accurate the generated tree is; however, the larger the value of δ will be. Thus, it is key to achieve a good trade-off of ε and δ and an important goal is to reduce the value of δ for a lower error probability. Supposing the value of ε/R is 0.01, to ensure $\delta = 0.001$, it only requires $n_{min} = 345$. Actually, the greater the value of n_{min} the higher the probability of accuracy is. But it will lead to more computational complexity at *numerical* nodes. Therefore, the study on n_{min} is much more significant for better results, which restricts the value of δ.

Given $\tau = 0.05$, we set values of n_{min} from 0.1k to 1k (1k = 1000) with a 0.1k variance each time, to ensure an invariant value of h_0; and run 100 times for each case and make use of k-single semi-random decision trees to vote (k = 10). Experimental results are plotted in Figure 1 (where "Max-Single" and "Bayes-Single" represent the values of 100-single trees on average in the classification methods of majority-class and Naïve Bayes respectively, "Max-Voting" and "Bayes-Voting" refer to average results of 10-votings), which demonstrate several characteristics. First, generally the voting result is in direct proportion to the average value of single trees and performs better. Second, the prediction accuracy of voting in "Bayes" is best, in which the least improvement rate is also more than 6% averagely as compared with the results in "Max".

However, as the value of n_{min} increases, the classification error presents in a light variance, which is limited to the bound of (0%, 3.54%). It is especially suitable for the case classified in "Bayes". As we know that the larger value of n_{min} indicates the more computation cost when splitting and the fewer instances collected at leaves, which probably results in the reduction on predictive accuracies. Thus, it is better to select a smaller value of n_{min}. The vast majority of experiments show that a candidate optimal value of n_{min} is set to 0.2k.

Analysis of Parameters h_0 and N

Massive experiments are conducted in the cases that the value of h_0 varies from 2 to Min(10, M(Attr)/2) with $N = 100$ and n_{min}= 0.2k (the maximum bound of height is set to M(Attr)/2 here, which is a primary conclusion in [2]). And the error rates averaged in 100-single trees are listed in Table 1, which of the minimum values are high-lighted in bold. Due to the limited space, the figures of change trends are omitted but several conclusions observed from the experimental results are given below. Firstly, the curves go with a stable tendency and the error rates of 100-single trees on average is approximately in direct proportion to the height of h_0, especially for the classification results with "Max". Secondly,

Table 1. The average error rates (%) of 100-single trees

h_0		2	3	4	5	6	7	8	9	10
Connect	Max	34.1	34.1	34.1	34.0	33.8	33.7	33.4	33.2	**33.1**
	Bayes	44.5	43.5	42.3	40.9	39.8	38.5	36.8	35.0	**33.1**
Covertype	Max	38.2	38.0	37.7	37.1	37.0	36.6	36.2	35.9	**35.6**
	Bayes	27.9	27.8	27.3	26.9	26.6	26.5	25.7	25.7	**25.1**
Waveform-40	Max	58.6	53.1	49.0	44.4	43.2	40.5	39.6	37.1	**37.0**
	Bayes	22.1	22.7	22.2	22.0	22.3	**21.9**	22.8	23.7	24.9

Table 2. Space costs(M) varying with h_0

h_0	Memory(M)								
	2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Connect	1			2	3	4	19	55	162
Covertype	2		4			5∼6			9∼13
Waveform-40	3	11		15		24	31	33	34

Table 3. Error rates (%) of voting

Database		**VFDTc**	**SRDTE**
Connect	Max	**33.8**	34.1
	Bayes	36.9	**35.4**
Covertype	Max	**38.6**	**38.6**
	Bayes	27.3	**25.4**
Waveform-40	Max	34.0	**30.4**
	Bayes	26.1	**15.2**

in the case with "Bayes", the predictive accuracy improves continuously as the value of h_0 increases till up to a certain value. Obvious turning points of down-trend appear, which are probably candidate optimal values we need.

To find an optimal value of h_0, not only should the predictive accuracy be ensured; but also the space overhead should be considered. Thus, the space costs of one single tree in different values of h_0 are further estimated, as shown in Table 2. The results demonstrate that space overheads go up slowly with the increasing of h_0 for Covertype and Waveform-40, while it increases sharply for Connect. The analysis shows that if the value of h_0 is initialized to 6 or a larger value, the space consumption increases from 0.5 to 80 times in comparison to that with $h_0 = 5$, while the prediction accuracy only improves less than 1.5%. Thus, our preliminary conclusion is to select the value of $h_0 = \text{Min}(5, M(\text{Attr})/2)$ as an optimal height of the tree for any databases.

Based on the optimal value of $h_0 = 5$, we consider the voting results from N-classifiers and initialize the value of N varying from 3 to 15. The curves of average results based on 10-runs are drawn in Figures 2∼3 (where *vo* refers to the classifier count). Evidently, with the increasing of the number of classifiers, the voting results from 3- to 15-classifiers vary lightly, especially when classifying with "Max". According to the theory of generation error in random ensembling trees, the more the number of trees, the more the probability of relevant attributes to appear in one single tree. However, more classifiers for voting will result in a heavier overhead of space with a less improvement of accuracy; hence, the selection on the value of N also should consider the space cost and the

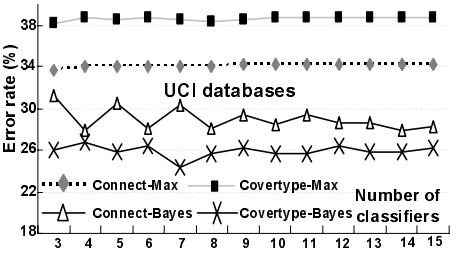


Fig. 2. Incremental results in HyperPlane

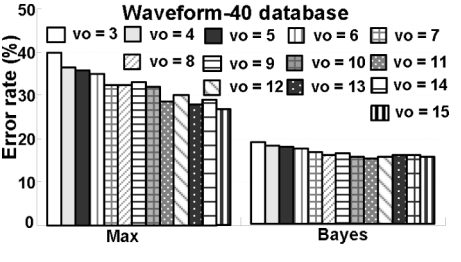


Fig. 3. Relation between N and error rate

requirement on accuracy contemporarily. Finally, we select $N = 10$, though the lowest error-rate occurs when the number of classifiers equals the upper bound of $N = 15$ in the case of classification with "Max".

3.2 Model Evaluation

With the obtained optimal values of parameters from the above analyses ($N = 10$, $n_{min} = 0.2$k and $h_0 = 5$), the performance of a single semi-random decision tree in SRDTE is measured by the 0-1 loss function, in which misclassifications are the loss cost.

Predictive Accuracy

To study the effect of predictive accuracy, we have compared SRDTE with the VFDTc algorithm [1] improved from the state-of-the-art algorithm of VFDT [6] and made use of the "Segment Probability Estimation"(abbreviated as *SPE*) to visualize processes of classification in Figure 4. Where *SPE* refers that testing data are divided into K-segments averagely (suppose the instance number in one segment is C; K=100) and each segment projects a score, i.e., the score of the i^{th} segment is i/K. After finishing a classification decision, count the number of accurate class labels (e.g., c) and obtain a probability of estimation($c/C^*\ i/K$). Hence, a point (i/K, $c/C^*\ i/K$) of estimation is formed in the 2-dimension axes (where the x- axis is the score and the y-axis is the estimated probability). Obviously, the perfect line composed of these points is $y = x$. The more the points approximate this curve, the better the ability of classification is. Figures 4-(c), (e)

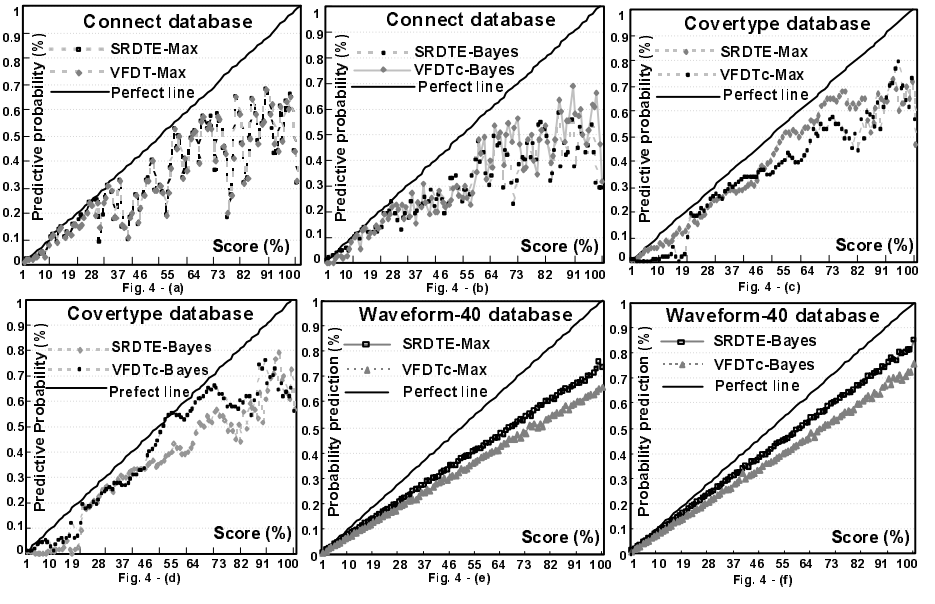


Fig. 4. Predictive probability curves of UCI databases

Table 4. Time & space costs on UCI

Database	Method	TRT+TCT(s) VFDTc	TRT+TCT(s) SRDTE
Connect	Max	3+3	3+3
	Bayes	3+3	3+3
Covertype	Max	13+5	5+4
	Bayes	12+7	5+25
		Memory(M)	
Connect	Max/Bayes	1.2	2.0
Covertype	Max/Bayes	3.0	2.0

Table 5. Predictive accuracy on LED

Noise rate (%)	Predictive accuracy(%) VFDTc Max	VFDTc Bayes	SRDTE Max	SRDTE Bayes
5	40.1	29.4	37.7	99.9
10	40.0	21.2	47.1	99.9
15	39.9	20.6	37.9	99.9
20	30.1	20.5	48.2	99.9
25	30.0	12.7	42.8	99.5
30	29.9	**49.8**	38.3	99.3

and (f) explicitly present that prediction ability of SRDTE outperforms VFDTc. The final accuracies have improved by the largest rate of 3.6% in the case with "Max" and increased from 1.5% to 10.9% in "Bayes", which are demonstrated in Table 3 as well. However, in Figures 4-(a)∼(b), the difference of classification between both algorithms is not explicit, which the average predictive probabilities with "Max" in SRDTE are lower than those in VFDTc by about 0.1% for Connect while it increases by 2% in the case with "Bayes". Moreover, though the result of classification in SRDTE seems worse for Covertype in Figure 4-(d), its average predicative probability is still more than that of VFDTc by 0.8% and the final accuracy of classification with "Bayes" is improved by 1.9%.

Speed and Space

In this subsection, the overheads of space and time are compared between SRDTE and VFDTc. The experimental results are shown in Table 4 and Figure 5. For the medium-sized databases in Table 4 (*TRT*: training time and *TCT*: test time), both algorithms present an unremarkable derivation. The overhead of space demanded in VFDTc is heavier than that in SRDTE, as plotted in Figure 5-(a). Actually, it grows linearly and exceeds SRDTE as the sizes of databases increase, where the ascending rate is from 0.29- to 1.04- times. With respect to the total time consumption in Figure 5-(b), we can observe explicitly that both of the costs in the training time are directly proportional to the sizes of databases.

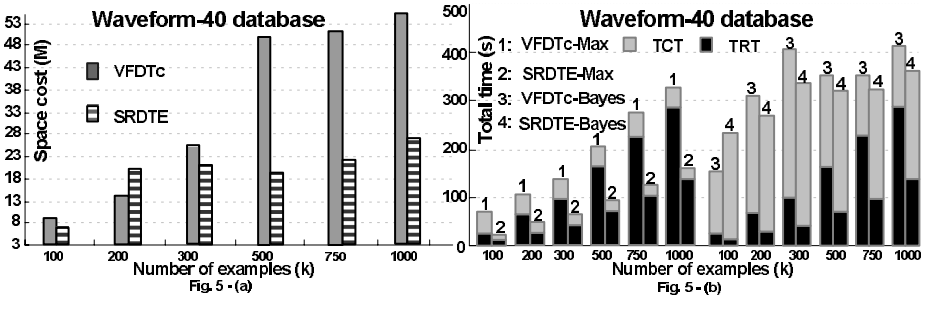


Fig. 5. Overheads of space and time on the database of Waveform-40

Besides, the overheads of total time in SRDTE with "Max" are least in the test time carrying a little variance; while the total time consumed in VFDTc increases rapidly, which is due to the fact that all of the available attributes are required computation for the best split-feature.

Robustness

It is inevitable to be affected from noise in the classification of data streams. Thus, to inspect the robustness to the noise for SRDTE compared with VFDTc, a large number of experiments are estimated on the LED database with discrete attributes only, which owns 17 irrelevant attributes varying with noise rates in attribute values. Experimental results given in Table 5 demonstrate that the performance of "SRDTE-Bayes" on anti-noise is the best almost maintaining the accuracy of 100%, which of the reason is similar to that of [10]. However, in the case with "Max", though the predictive accuracies in SRDTE are higher than those in VFDTc with a rate of 7.03% on average, it does not perform well in consideration of the demand on accuracy from user. We try to increase the values of h_0 and an interesting observation from further experiments is that the accuracy in "SRDTE-Max" could be improved continuously with the increasing of h_0. Thus, an additional experimental conclusion is given now. For several special databases with a large number of irrelevant attributes or more discrete attributes, if the space restriction is satisfied, the initial height of tree should approach to the value of Max(5, M(Attr)/2) in the case of classification with "Max".

Moreover, we observe with surprise that the accuracy of "VFDTc-Byaes" is the highest when the noisy rate is up to 30% compared with the results in the case of lower noise rates. A reason is caused that the distribution of class labels in LED is even. If the noise rate amounts to a certain value, a new distribution of attribute values would be formed possibly, which adapts to the case of Naïve Bayes. Actually, a similar accuracy will keep up until the noise rate is up to 37%. After that value, the accuracy of classification in "VFDTc-Bayes" is much worse, which only approaches to 10.03%.

Application

We had a try to apply SRDTE into mining streaming shopping data from Yahoo! website via its web services interface. The database used in our experiments is sampled from their shopping database of product offers and merchants, including 113k records of transaction items with 23 attributes. The features contain the information of products (*Price*, *NumRating*, *ShippingCost*, etc.) and related

Table 6. Time & space costs on real data

TRT+TCT(s)				**Memroy(M)**	
VFDTc		SRDTE			
Max	Bayes	Max	Bayes	VFDTc	SRDTE
3+1	3+3	3+1	3+48	2	<1

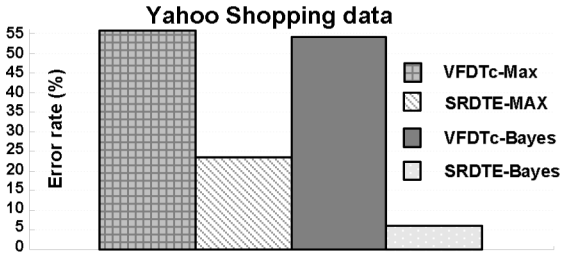


Fig. 6. The error rate of classification

information of merchants, which consist in attributes of (*NumRatingsOfMerchant*, *PriceSatisfaction-Rating*, *IsCertifiedMerchant*, etc.) (See [9] for more details). In order to mine the relations between credibility of merchants and possible factors, we define the attribute of *OverallRating* with different scores as our class labels, which are divided into five values. We make use of 2/3 of total records as the training data and the rest as the test data. The experimental study shows that our model could judge merchants' credibility in a higher accuracy, which is up to 76.49% increased by 32.17% compared with that of VFDTc in the case with "Max". Meanwhile, when classifying with "Bayes", its accuracy is improved by 47.90% in comparison to VFDTc with the error rate of 54.08%, which is described in Figure 6. Besides, Table 6 lists the estimation results on the time and space costs, which demonstrate that the total space or time cost to deal with these data is much less for SRDTE; especially in the method of Naïve Bayes, the maximum time consumption of one single tree is less than one minute.

4 Related Work

Fan et al. proposed *Random Decision Trees* (RDTs) [2] in 2003. The model of RDTs selects the splitting feature randomly without any heuristics and has good performances on space, runtime and the robustness (also seen in [14]), which is more suitable for medium and large databases with discrete attributes. However, it requires discretization on numerical attributes beforehand so it does not adapt to process data streams in real time directly. Hu et al's work[10] in 2007 was on the classification of data streams based on RDTs, which adopts a new method to deal with numerical attributes of data streams in the inequality of Hoeffding Bound and introduces the Naïve Bayes method for better predictive accuracy. However, some important factors of the impact on the model abilities, such as the initial height of a tree, the count of trees and several parameters used in the Hoeffding Bound inequality, are not analyzed systematically and further mechanisms still need to be developed. Besides, this algorithm does not take into account of real-time streaming data.

5 Conclusions

In this paper, we have proposed a model of Semi-Random Decision Tree Ensembling (SRDTE) for classification on streaming data. It generates each semi-random decision tree incrementally after seeing the training instances, and takes a new data structure for a quick search of target objects. Meanwhile, it adopts the voting mechanism of majority class, which has improved the predictive accuracy largely. Besides, the optimal values of significant parameters, which have an impact on the model's abilities, are estimated in an extensive study. Based on these obtained optimal values, we have evaluated SRDTE on its accuracy, run time, space cost and the robustness. Our experimental results show that SRDTE has better performance compared to VFDTc. Also, an experiment to apply SRDTE to some Yahoo! shopping data demonstrates that SRDTE is efficient and effective on real data streams. Our future work is to obtain massive data via the Yahoo! Shopping Web Services Interface to find potential patterns of transactions continuously with consideration of concept drift in real shopping data.

References

1. Gama, J., Rocha, R., Medas, P.: Accurate decision trees for mining high-speed data streams. In: 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 523–528 (2003)
2. Fan, W., Wang, H., Yu, P.S., Ma, S.: Is random model better? On its accuracy and efficiency. In: 3rd IEEE International Conference on Data Mining, pp. 51–58 (2003)
3. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
4. Gao, J., Fan, W., Han, J.: On Appropriate Assumptions to Mine Data Streams: Analysis and Practice. In: 7th IEEE International Conference on Data Mining, Omaha, Nebraska, USA, pp. 143–152 (2007)
5. Ho, T.K.: The Random Subspace Method for Constructing Decision Forests. IEEE Trans. Pattern Analysis and Machine Intelligence 20(8), 832–844 (1998)
6. Domingos, P., Hulten, G.: Mining High-Speed Data Streams. In: 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, USA, pp. 71–80 (2000)
7. Hoeffding, W.: Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association 58, 13–30 (1963)
8. Blake, C., Keogh, E., Merz, C.: UCI repository of machine learning databases (1998), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
9. Yahoo! shopping web services, `http://developer.yahoo.com/everything.html`
10. Hu, X., Li, P., Wu, X., Wu, G.: A Semi-Random Multiple Decision-Tree Algorithm for Mining Data Streams. Journal of Computer Science and Technology 22(5), 711–724 (2007)
11. Gehrke, J., Ganti, V., Ramakrishnan, R., Loh, W.-Y.: BOAT-optimistic decision tree construction. In: 1999 ACM SIGMOD International Conference on Management of Data, pp. 169–180 (1999)
12. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)

13. Shafer, J., Agrawal, R., Mehta, M.: SPRINT: A scalable parallel classifier for data mining. In: 22nd International Conference onVery Large Databases, pp. 544–555. Morgan Kaufmann, San Francisco (1996)
14. Fan, W.: On the Optimality of Probability Estimation by Random Decision Trees. In: 9th National Conference on Artificial Intelligence (AAAI 2004), pp. 336–341. AAAI Press, San Jose (2004)
15. Maron, O., Moore, A.W.: Hoeffding races: Accelerating model selection search for classification and function approximation. In: Cowan, J.D., Tesauro, G., Alspector, J. (eds.) Advances in Neural Information Processing Systems, pp. 59–66. Morgan Kaufmann, San Mato (1994)

Exploiting the Block Structure of Link Graph for Efficient Similarity Computation

Pei Li[1,2], Yuanzhe Cai[1,2], Hongyan Liu[3], Jun He[1,2], and Xiaoyong Du[1,2]

[1] Key Labs of Data Engineering and Knowledge Engineering, Ministry of Education, China
[2] School of Information, Renmin University of China, Beijing, China
{lp,yzcai,hejun,duyong}@ruc.edu.cn
[3] Department of Management Science and Engineering, Tsinghua University, Beijing, China
hyliu@tsinghua.edu.cn

Abstract. In many real-world domains, link graph is one of the most effective ways to model the relationships between objects. Measuring the similarity of objects in a link graph is studied by many researchers, but an effective and efficient method is still expected. Based on our observation of link graphs from real domains, we find the block structure naturally exists. We propose an algorithm called *BlockSimRank*, which partitions the link graph into blocks, and obtains similarity of each node-pair in the graph efficiently. Our method is based on random walk on two-layer model, with time complexity as low as $O(n^{4/3})$ and less memory need. Experiments show that the accuracy of *BlockSimRank* is acceptable when the time cost is the lowest.

Keywords: Block, link graph, SimRank, object ranking.

1 Introduction

Many applications require executing "find-similar-object" query in their transactions, which implies the computation of similarity between objects. As an example, thinking about clustering a collection of documents, we should compute the similarity scores of each pair documents first. In collaborative filtering, similar items are clustered based on their similarities [6].

In many real-world domains, linkage among objects can be the most useful information for similarity measuring. One obvious example is the WWW. Due to the limited effect of understanding web pages automatically, link analysis is still a popular approach to exploit the web structure. If we treated these object-to-object links as edges, the real-world domain can be best described as a heterogeneous graph. In this paper, we focus our research target on the similarity computation of link graph, which is the abstract model of many real-world domains.

There have been many similarity computing methods [2, 3, 4, 5], which will be introduced in related work. Thereinto, Jeh and Widom [2] propose a notable method called *SimRank* for assessing the similarity that says "two objects are similar if they are related to similar objects". *SimRank* provides a wonderful definition for similarity

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 389–400, 2009.
© Springer-Verlag Berlin Heidelberg 2009

on a link graph, but unfortunately, it computes the similarity between every pair of objects, which results in high complexity in both time and space.

The study of real-world datasets indicates that block structure naturally exists among objects. Taking the web graph for example, web pages of host *A* are more likely to cite pages belonging to the same host *A*. Viewing a host as a block, the web graph can be regarded as a block structure, and the similar phenomenon can be found in other domains. As another example, based on the statistics of words frequencies occurring in articles, the density of linkages between collections of articles and words is shown in Figure 1(a) (adapted from Figure 2 in [7]).

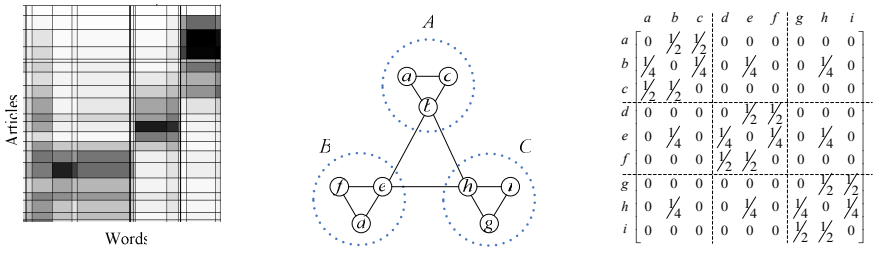


Fig. 1. (a) Density of linkages between articles and words. (b) The graph of a heterogeneous dataset. (c) The corresponding transition matrix.

Taking advantage of the block structure, we propose an algorithm called *BlockSimRank* to effectively measure similarity by following 4 steps: (1) partition the link graph into blocks properly; (2) compute the similarity of each block-pair; (3) for each block, use *SimRank* to obtain local similarity scores; (4) estimate the "inter-block" nodes similarity using block-pair similarity and local similarity.

Comparing with *SimRank*, our method can significantly speed up the identification of similar objects. *SimRank* computes the similarity of each node-pair. However, *BlockSimRank* only computes the similarity between blocks and the similarity between nodes in the same block. The similarity between nodes in different blocks is estimated reasonably. For example, in a link graph shown in Figure 1(b), we can gain its transition matrix shown in Figure 1(c). Because most edges are intra-block links, only the dense area of transition matrix (diagonal area in Figure 1(c)) is worth considering. For a graph having n nodes, supposing this graph has m blocks and let d denote the average number of neighbors, the time complexity of *SimRank* and *BlockSimRank* are $O(kn^2d^2)$ and $O(kd^2(m^2+n^2/m))$ respectively, where k is the number of iterations. When setting m properly, the least time cost of *BlockSimRank* is $O(n^{4/3})$.

Experiments on real datasets are conducted to test the accuracy and efficiency of *BlockSimRank*. By partitioning the link graph, our method can remarkably speed up the similarity computation and win great advantage in efficiency with the accuracy retained, compared with other methods such as *SimRank* and *SimFusion*[4].

The rest of the paper is organized as follows. Section 2 surveys related work. In Section 3, we exploit the block structure of link graph in real domains. Afterwards, Section 4 describes the *BlockSimRank* method and explains its theoretical model. The results of experiments are shown in Section 5 and this study is concluded in Section 6.

2 Related Work

Similarity measuring has been extensively studied in different disciplines for a long time, with multiple methods proposed [2, 3, 4, 5, 9]. Traditional approaches that calculate the similarity of documents using Vector Space Model (VSM) (included in [3]) and its variations can be viewed as a general algorithm, which map documents and queries into a vector space. They differ in how the vectors are constructed and how weights are assigned [4]. These methods are useful to compute the similarity between sets of objects (e.g., a collection of documents), whereas they are not good at using the external information about objects (e.g., the references among documents).

In some situations, linkages among objects can be the only or the most explicit information available. Many datasets are heterogeneous [1], in which there may be multiple object and link types, and are best described as networks or graphs. Naively applying traditional approaches may bring on invalid conclusions.

Some researchers use multiple relationships to compute the similarity between objects based on different theoretical foundations. In the WWW, [5] explains an approach using hyperlinks among pages to calculate the similarity of web objects. To realize the global ranking of objects in a static graph, Jeh and Widom [2] proposed a strategy called *SimRank*, in which the similarity between two objects is updated iteratively by the average similarity between objects linked with them. For example, in Figure 1(b), the similarity between *a* and *d* (defined as $S(a, d)$) is decided by the similarity $S(c, f)$, $S(c, e)$, $S(b, f)$ and $S(b, e)$. The intuitive underlying model of *SimRank* is "random surfer-pairs", a concept derived from *PageRank* algorithm [8]. Unfortunately, *SimRank* takes $O(N^2)$ time and $O(N^2)$ space, making it impractical for large datasets. The authors of [2] also discuss a pruning technology to approximate *SimRank*, but it is a challenge to select the right node pairs in initial stage.

Yin et al. [7] proposed a hierarchical structure called *SimTree* to represent similarities between objects in a compact way. *SimTree* only computes and stores the similarities of sibling nodes and the ratio of each node compared with its parent, thus, it obtains great efficiency compared with *SimRank*. Besides, Xi et al. [4] use a Unified Relationship Matrix (*URM*) to represent a collection of heterogeneous objects and their interrelationships. By iteratively computing over the *URM*, they present a similarity-calculating algorithm called *SimFusion*. *SimFusion* can effectively integrate relationships from multiple sources to measure the similarity, and takes $O(kn^2d)$ time, where d is a constant with respect to n.

In this issue, Sun et al. [9] proposed a correlation degree computation approach for relevance search and anomaly detection, which combines random walk and graph partitioning together to improve scalability. Although the principle of [9] is similar to our work, it is only suitable for bipartite graph, while our method is applicable to arbitrary graph, even if this graph is not a connected graph.

Our method, *BlockSimRank*, is originated from the observation of block structure concealed in the link graph. A similar work has been done by [11], which exploit the block structure of the web for computing *PageRank*. The underlying model of *BlockSimRank* based on random walk [10], a concept that simulates walkers randomly surf the graph along edges; and k-way graph partitioning, a method introduced by [12], which is a high-quality and computationally inexpensive refinement algorithm. Our method can be mainly applied in link-based object ranking.

3 Block Structure of Link Graph in Real Datasets

The taxonomy or hierarchical categories are ubiquitous among data objects in many domains. Usually, two objects of the same type are more similar than objects in different categories, and it makes similar objects gathered into the same block.

More precisely, for two blocks A and B, let $N(A)$ denote the number of links of block A (all "intra-block" and "inter-block" links are included), and $N(A, B)$ denote the number of links between A and B. Obviously, $N(A, B) = N(B, A)$, and $N(A, A)$ represents the number of intra-block links of A. We take

$$R(A,B) = \frac{N(A,B)}{N(A)} \tag{1}$$

to measure the ratio between $N(A, B)$ and $N(A)$. It is easy to know $0 \le R(A,B) \le 1$ and if $R(A, B) = 1$, all links starting from objects in block A point to objects in block B. For Example, in Figure 1(b), the value of $R(A, B)$ is 1/5.

Block Structure of the Citation Graph. Considering a corpus of scientific papers, each paper can be regarded as an identically distributed node, and the references or cited-by relationships correspond to edges between nodes. Our dataset is crawled from ACM Computing Classification System (CCS) [13], which is a subject classification system for Computer Science. Taking Section *F* in ACM CCS for an example, there are 6 main categories in it, and each category includes some subtopics.

Treating each category as a block, We use $R(A, B)$ defined in equation (1) to represent the flow of links from A to B, and color the plot in the grid with black or grey according to the value of $R(A, B)$. The width of a plot corresponds to the number of papers in that category. The block structure of citation graph is shown in Figure 2.

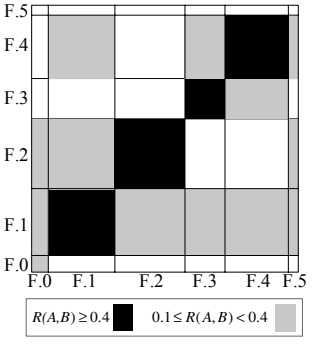


Fig. 2. Block structure of the citation graph of Section *F* in ACM CCS

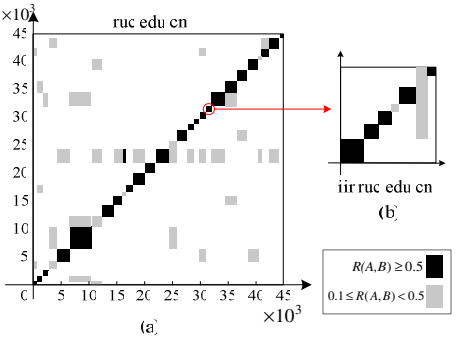


Fig. 3. (a) Block structure of *ruc.edu.cn* at host level. (b) Block structure of *iir.ruc.edu.cn* at directory level.

Block Structure of the Web Graph. The WWW is naturally modeled as a huge graph, with nodes representing pages and edges representing hyperlinks. We do a crawl of the *ruc.edu.cn* in July 2008 to generate a link graph for analysis. This graph contains roughly 45,000 nodes, and more than 0.5 million links with ineffectual

Table 1. Terms for a hierarchical view of URL on the web

Terms	Example
domain	ruc.edu.cn
host	iir.ruc.edu.cn
directory	iir.ruc.edu.cn/people/
page	iir.ruc.edu.cn/people/students.html

hyperlinks (links citing other pages out of the graph) removed. To investigate the structure of the web, we use the terms given in Table 1.

A similar work has been done by Kamvar et al. [11] by using dot-plots to visualize the link matrix at page level. Our work is done at host level (Figure 3 (a)) and directory level (Figure 3 (b)) based on the measurement of links flow among blocks. A view of nested block structure is shown in Figure 3, by filling plots with different color according to $R(A,B)$. The aggregation of black plots on the diagonal line indicates that most links are "intra-block" links.

4 The *BlockSimRank* Method

4.1 Preliminaries

Transition Matrix. Given a undirected graph $G(V, E)$, supposing there is a random surfer standing on node a, he has identical probability to visit each node direct-connected to a on next step, and possibility of *zero* to other nodes. Note that edge weights may be imported to represent varying importance of links, but it's not the key point of our work. We only focus on undirected graphs and it's easy to extend our method to directed graphs.

A transition matrix or stochastic matrix is introduced to describe the transition probability of a Markov Chain [14]. We define transition matrix T as a square matrix each of whose rows consists of nonnegative values, with each row summing to 1. In an undirected graph, T is symmetric. For example, the transition matrix shown in Figure 1 (c) corresponds to the graph in Figure 1 (b). Based on the block feature we discussed in Section 3, we use

$$T = \begin{bmatrix} P_{AA} & P_{AB} & P_{AC} \\ P_{BA} & P_{BB} & P_{BC} \\ P_{CA} & P_{CB} & P_{CC} \end{bmatrix}$$

to generalize it, where P_{IJ} is the probability matrix between all nodes in block I and all nodes in J. Note that only the dense valued areas (e.g. P_{AA}, P_{BB} and P_{CC}) have significant effect on similarity computation. This feature is utilized by our method.

Graph Partitioning. The graph partitioning problem is to divide the vertices of a graph into n roughly equal parts, with the number of edges connecting nodes in different parts minimized. In our method, blocks are detected by graph partitioning. Many algorithms [12, 15, 16] have been developed to find reasonably partitions. We use the multilevel k-way partitioning method proposed by Karypis et al. [12], whose

strategy is performing a *k*-way partitioning on the smaller graph, and then refine it to construct a *k*-way partitioning of the original graph. This algorithm partitions a graph $G = (V, E)$ in $O(|E|)$ time and is implemented in METIS package [18].

4.2 Assessing Similarity between Nodes

Basic *SimRank*. As algorithm *SimRank* [2] is closely related to our work, we describe it here. Given an undirected graph $G = (V, E)$, we denote the *k*-th iterative similarity between nodes V_a and V_b by $S_k(V_a, V_b) \in [0,1]$. If $a=b$ then $S_k(V_a, V_b) = 1$. Otherwise, given transition matrix T and initial similarity matrix Sim_0, we get

$$S_k(V_a, V_b) = c \cdot \sum_{i=1}^{O(T)} \sum_{j=1}^{O(T)} T_{ai} \cdot T_{bj} \cdot S_{k-1}(V_i, V_j) \tag{2}$$

where c is a decay constant (usually set to be 0.8) describing the attenuation of link influences. $O(T)$ is the order of transition matrix T. At the same time, T_{ai} is the element of T on the a-th row and i-th column, and analogously for T_{bj}. A slight notice here is the initialization of $S_k(V_a, V_b)$ when k=0. Since we can't foreknow the similarity between two objects before iteration, it is reasonable to simply define $S_0(V_a, V_b) = 1$ for $a=b$, and $S_0(V_a, V_b) = 0$ for $a \neq b$.

We give the major steps of *SimRank* algorithm as follows.

Algorithm 1. (**Basic *SimRank***)
Input: *transition matrix T, initial similarity matrix Sim_0, tolerance factor t.*
Output: c*onvergent similarity matrix Sim_k.*

$k \leftarrow 1$
while ($\max(|S_k(V_a, V_b) - S_{k-1}(V_a, V_b)|) > t$)
  $k \leftarrow k + 1$
  $Sim_{k-1} \leftarrow Sim_k$
  for each *element* $S_k(V_a, V_b)$ *in* Sim_k

$$S_k(V_a, V_b) = c \cdot \sum_{i=1}^{O(T)} \sum_{j=1}^{O(T)} T_{ai} \cdot T_{bj} \cdot S_{k-1}(V_i, V_j)$$

  end for
end while
return Sim_k

Local Similarity and Global Similarity. Local similarity is the similarity between objects in the same block without considering objects in other blocks. Let $LSim(B)$ denote the local similarity of block B. $LSim(B)$ is the iteration convergence of all node-pairs' similarities in block B, which means given transition matrix P_{BB} (defined in Section 4.1) and initial similarity matrix $LSim_0(B)$ (a identity matrix), we can get $LSim(B)$ recursively by Equation (2). As an example, for block A in Figure 4(a), relationships between A and its nodes can be better visualized in another view shown in Figure 4 (b), which indicates $LSim(A)$ is a 3-by-3 matrix.

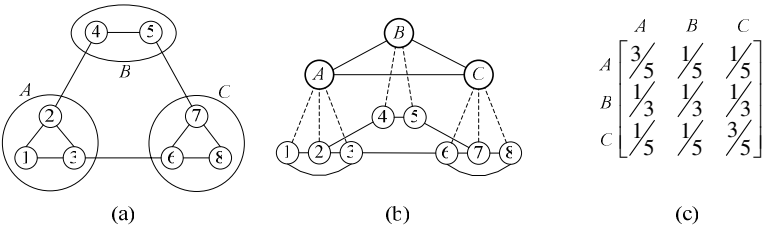


Fig. 4. (a) A view of block structure. (b) Another view of block structure. (c) Transition matrix between blocks.

Definition 1. (**Block Similarity**) Treating each block as an object, we can compute the similarity between blocks as we do between objects by Equation (2). The transition probability between block *A* and *B* is *R*(*A, B*) (defined by Equation (1)), so we can get transition matrix *T*. Figure 4 (c) shows an transition matrix between blocks. Let *BSim*(*A, B*) denote the similarity between block *A* and *B*.

Definition 2. (**Inter-Block Nodes Similarity**) Inter-block nodes similarity is the similarity between nodes in different blocks. Supposing node V_a and V_b are in block *A* and *B* respectively, we define the similarity between V_a and V_b on the *k*-th iteration by

$$S_k(V_a, V_b) = S_k(V_a, A) \cdot BSim(A, B) \cdot S_k(V_b, B) \tag{3}$$

where $S_k(V_x, X)$ represents the similarity between object *x* and the center of block *X*. Usually the center of block *X* is virtual and hard to ascertain, so we take the average similarity between *x* and every object in block *X* to estimate it. That is

$$S_k(V_x, X) = \frac{1}{|X|} \sum_{i=1}^{|X|} S_k(V_x, V_i) \tag{4}$$

where |*X*| is the number of objects in block *X*. We will explain equation (3) in section 4.4 later. Further study indicates that it's not necessary to compute inter-block nodes similarity on every iteration.

Theorem 1. If block similarity and local similarity are convergent, inter-block nodes similarity will be convergent too.

Proof. If local similarity $S_k(V_x, V_i)$ is convergent, $S_k(V_x, X)$ will be convergent too according to Equation (4). Since block similarity *BSim*(*A, B*) is convergent, we can get the Theorem 1 soon from Equation (3).

Definition 3. (**Global Similarity**) Given an undirected graph *G* = (*V, E*), as stated above, for a node-pair V_a and V_b, if they belong to the same block, their similarity $S(V_a, V_b)$ is the local similarity decided by Equation (2), otherwise, it is the inter-block nodes similarity decided by Equation (3). We call the similarity of any node-pair calculated this way global similarity, and use *GSim* to denote it.

The *BlockSimRank* Algorithm. The block structure suggests a fast algorithm. We propose *BlockSimRank* for more effective and efficient similarity computation, and its major steps are summarized as follows:

1. Split the graph into n roughly equal blocks using METIS [18];
2. For the k-th iteration, regarding each block as an object, compute the Block Similarity Matrix using basic *SimRank* algorithm;
3. Compute the Local Similarity Matrix of each block by basic *SimRank* algorithm;
4. If block similarity and local similarity are not convergent, let $k = k+1$ and then jump to step 2; else continue;
5. Estimate the Inter-Block Nodes Similarity using Equation (3);
6. Obtain global similarity matrix *GSim*.

4.3 Complexity Analysis

For simplicity, we assume the link graph $G = (V, E)$ has n objects and can be split into m roughly identical blocks.

Time Complexity. The time consumed by step 1 of *BlockSimRank* Algorithm usually can be ignored, because it takes $O(|E|)$ time and the real graphs are often sparse. The time cost of *BlockSimRank* is mainly composed of block similarity computation (step 2) and local similarity computation (step 3). Noting that the time cost of *SimRank* is $O(kd^2n^2)$, we write the equation for time complexity of *BlockSimRank* as follows:

$$Time(m) = kd^2m^2 + mkd^2(n/m)^2 = kd^2(m^2+n^2/m) \tag{5}$$

k is the number of iteration and d^2 is the average direct-connected neighbor pairs of a block-pair or node-pair (usually a constant with respect to n^2). Calculating the derivative of $Time(m)$ and setting $dTime(m)/dm=0$, we get $m_0 = (2n)^{2/3}/2$, and when $m=m_0$, the time complexity is $O(n^{4/3})$, that is the lowest.

Space Complexity. Since it is not necessary to put the entire global similarity matrix in RAM, *BlockSimRank* takes $O(m^2+n^2/m)$ space to store the block similarity matrix and local similarity matrices. When $m=m_0$, the space complexity is the smallest too.

Our experiment in Section 5 shows that the accuracy is still comparable to the accuracy of *SimRank* when $m=m_0$, so it is our suggestion to set $m=m_0$ when using *BlockSimRank* method. Besides, the computations of local similarity matrices are independent, which indicates the possibility of parallelization.

4.4 Theoretical Model

The block structure implies that a random surfer is more likely to meet another surfer from the same block. The meeting of two surfers from different blocks also needs to consider, but the probability is little and we can estimate it. We develop an intuitive model based on two kinds of random surfers: the surfer traveling among blocks (we call it *BSurfer*) and the surfer traveling among local nodes (we call it *NSurfer*).

Random Walk on Two Layers Model. In Figure 5 (a), all nodes belong to the same block. The similarity $S(a, b)$ is determined by similarities of all its direct-connected

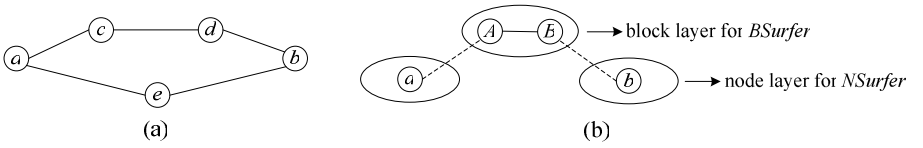


Fig. 5. (a) The graph for *NSurfer*. (b) The meeting of two *NSurfers* in different blocks

neighbor pairs and the probability from (a, b) to these node-pairs. That is to say, $S_k(a, b) = 0.25(S_{k-1}(c, e)+S_{k-1}(c, d)+S_{k-1}(e, e)+S_{k-1}(d, e))$.

If nodes a and b are in different blocks as shown in Figure 5 (b), *NSurfer* at node a and *Nsurfer* at node b can't meet without the help of *BSurfer*. Regarding *BSurfer* as a transmitter or bridge, the similarity $S(a, b)$ is determined by $BSim(A, B)$ and the probability from (a, b) to block-pair (A, B). We use Equation (4) to approximate the probability from object x to its block X. Thus we have the following theorem.

Theorem 2. Global similarity of any node-pair in Graph G can be computed based on the Random Walk on two layers model.

Proof. Given a node-pair (a, b), its similarity is determined by similarities of all its in-neighbor node-pairs (or block-pair) and the probability from (a, b) to these node-pairs (or block-pair).

5 Experimental Evaluation

We described above an approach for efficient similarity computation on link graph. In this section, the accuracy and efficiency of our approach will be tested, compared with the following similarity measuring methods: (1) *SimRank* [2], an approach that iteratively computes the similarity of each node-pair; (2) *SimFusion* [4], an approach that reinforces and propagates the similarity between objects.

All experiments are performed on a PC with a 1.86G Intel Core 2 processor, 2 GB memory, and Windows XP Professional. All algorithms are implemented using Java.

Datasets. A good evaluation of similarity measuring methods is difficult, because the similarity between two objects is hard to ascertain without performing extensively user studies. ACM CCS [13] is a credible subject classification system for Computer Science. It can provide an identification of similar papers by organizing these papers in the same category. Our dataset is crawled from Section *F* in ACM CCS. With the

Table 2. (a) Node number of each category. (b) Edge number between two categories.

Category	Number of nodes
F.1	2738
F.2	4431
F.4	2380

(a)

	F.1	F.2	F.4
F.1	9513	4084	2828
F.2	4084	18843	1046
F.4	2828	1046	8508

(b)

ineffectual papers removed, we get a citation graph using the reference and "cited-by" information. This undirected graph has 9549 nodes and 44822 edges. The detailed node number of each category and edge number between each pair of categories are shown in Table 2.

The Selection of Partition Number *m*. When partitioning a graph into blocks at the beginning of *BlockSimRank* algorithm, the number of partitions should be decided. From Equation (5) we know *m* influences time complexity remarkably. In the meantime, too many partitions imply too many edges between different blocks, which induce the decline of accuracy.

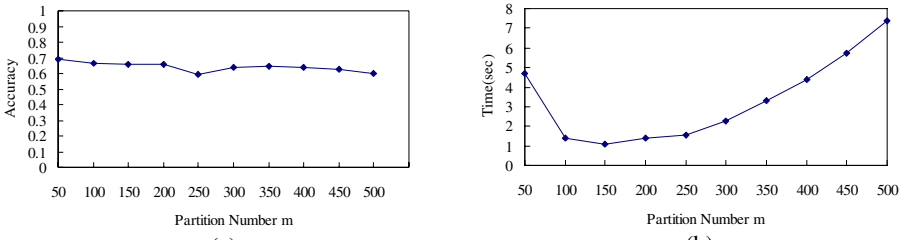


Fig. 6. (a) Accuracy of different partition numbers. (b) Time/iteration.

There is a trade-off between accuracy and efficiency. The accuracy and the time consumed on each iteration of different partition numbers are shown in Figure 6 (a) and (b) respectively. With the increase of partition numbers, the accuracy descends slowly. As we discussed in Section 4.3, if the assumed conditions are satisfied, when $m = m_0 = (2n)^{2/3}/2$, time cost is the lowest. In real situations, we set $m_0 = e(2n)^{2/3}/2$, where e is an adjustment factor and in our ACM dataset, $e \approx 0.4$, making $m_0 \approx 150$.

Experiments show that the accuracy of *BlockSimRank* is acceptable when time cost is the lowest. So we suggest setting $m = m_0$ when using *BlockSimRank* algorithm.

Accuracy. We use PAM [17], a *k*-medoids clustering approach, to cluster papers into groups based on global similarity matrix. In the meantime, ACM CCS provides a credible classification on these papers. Comparing these groups with CCS categories, let *C* denotes the number of correct classified papers, and *N* denotes the total number of papers. We define accuracy as the ratio between *C* and *N*.

In this experiment, we compare *BlockSimRank* with *SimRank* and *SimFusion* on accuracy. We set $m = m_0$ in *BlockSimRank* (now time cost is the lowest). As Table 3 shows, *BlockSimRank* wins slight advantage to other methods, which is because graph partitioning can eliminate noise to some extent.

Table 3. Accuracy of different approaches

Approach	Accuracy
BlockSimRank($m=m_0$)	0.6609
SimRank	0.6419
SimFusion	0.6073

Table 4. Performances of different approaches

Approach	Time/Iteration	Total
BlockSimRank	1.2 sec	× 10
SimRank	13222.7 sec	× 9
SimFusion	5763.2 sec	× 15

Performances. The motivation of *BlockSimRank* is exploiting the block structure to improve the performance of *SimRank*. By partitioning a global matrix into several local matrices, the performance is enhanced markedly. The time complexities on each iteration of *SimRank*, *BlockSimRank* and *SimFusion* are $O(d^2n^2)$, $O(d^2(m^2+n^2/m))$ and $O(dn^2)$ respectively, where d is related to n and explained above.

When we set $m = m_0$, the time cost of *BlockSimRank* is $O(n^{4/3})$, which wins notable advantage to *SimRank*. We list the time consumed on each iteration and the total time before convergence in Table 4. The tolerance factor of convergence is set to be 0.001, which is the maximum difference (explained in next subsection) of similarity scores between adjacent iterations. From Table 4 we can see the improvement of performances from *SimRank* to *BlockSimRank* is huge, which is from $O(n^2)$ to $O(n^{4/3})$, where n is the node number 9549.

Convergence Rate. We measure the convergence rate from two aspects: maximum difference and accuracy. $S_k(i, j)$ denoting a similarity score on the k-th iteration, maximum difference M_k can be described as $max(|S_k(i, j) - S_{k-1}(i, j)|)$. If M_k is less than the tolerance factor of convergence (set to be 0.001), iterative process will finish.

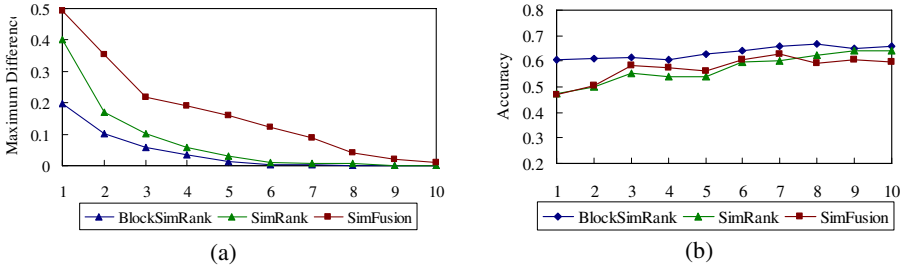


Fig. 7. (a) Maximum differences of each iteration. (b) The accuracy of each iteration.

By analyzing similarity matrices of the first ten iterations of each approach, we get the maximum difference of each iteration shown in Figure 7 (a), and the accuracy of each iteration shown in Figure 7(b). Comparing with other methods, *BlockSimRank* holds advantage at the beginning of iterative process. Considering the first iteration in Figure 7, the maximum difference of *BlockSimRank* is the smallest and the accuracy is the highest. This advantage is attributed to graph partitioning, because it can detect potential groups via partitioning a graph into blocks. The similarity between objects in the same block is usually higher than the one in different blocks, which results in more accurate clustering. Due to this advantage, *BlockSimRank* algorithm has higher convergence rate than *SimFusion*, and is as fast as *SimRank* in most cases.

6 Conclusions

In this paper, we propose a high efficient similarity computation method by exploiting the block structure of link graph. Many datasets involve relationships between objects and can be best described as link graphs. Using the block structure implied in the link graph, we propose an algorithm called *BlockSimRank*, which partitions a graph into

blocks, and iteratively computes the similarity between blocks and the similarity between objects in the same block until similarity matrices are convergent, and then estimate the similarity between objects in different blocks properly. The time cost of our method is $O(n^{4/3})$, whereas *SimRank* takes $O(n^2)$ time. Experimental results show *BlockSimRank* achieves high efficiency and acceptable accuracy in computing similarity of a link graph.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China under Grant No. 70871068, 70621061, 70890083, 60873017, 60573092 and 60496325.

References

1. Getoor, L., Diehl, C.P.: Link mining: A survey. In: SIGKDD 2005 Explorations, vol. 7(2), pp. 3–12 (2005)
2. Jeh, G., Widom, J.: SimRank: A measure of structural-context similarity. In: SIGKDD 2002, pp. 538–543 (2002)
3. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. ACM Press/Addison-Wesley (1999)
4. Xi, W., Fox, E.A., Fan, W., Zhang, B., Chen, Z., Yan, J., Zhuang, D.: SimFusion: measuring similarity using unified relationship matrix. In: SIGIR 2005, pp. 130–137 (2005)
5. Dean, J., Henzinger, M.R.: Finding Related Pages in the World Wide Web. In: WWW 1999, pp. 1467–1479 (1999)
6. Shardanand, U., Maes, P.: Social information filtering: Algorithms for automating "word of mouth". In: Proceedings of the Conference on Human Factors in Computing Systems, Denver, Colorado (1995)
7. Yin, X., Han, J.: Yu. P.S.: Linkclus: Efficient clustering via heterogeneous semantic links. In: VLDB 2006, pp. 427–438 (2006)
8. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: Bringing order to the Web. Technical report, Stanford University Database Group (1998)
9. Sun, J., Qu, H., Chakrabarti, D., Faloutsos, C.: Relevance search and anomaly detection in bipartite graphs. SIGKDD Explorations 7(2), 48–55 (2005)
10. Lovasz, L.: Random walks on graphs: a survey. Combinatorics, Paul Erdos is Eighty, vol. 2, Keszthely (Hungary), pp. 1–46 (1993)
11. Kamvar, S.D., Haveliwala, T.H., Manning, C.D., Golub, G.H.: Exploiting the Block Structure of the Web for Computing PageRank. Technical Report, Stanford University, Stanford, CA (2003)
12. Karypis, G., Kumar, V.: Multilevel k-way partitioning scheme for irregular graphs. Journal of Parallel and Distributed Computing 48(1), 96–129 (1998), `http://www.cs.umn.edu/~karypis`
13. ACM Computing Classification System, `http://portal.acm.org/ccs.cfm`
14. Kallenberg, O.: Foundations of Modern Probability. Springer, New York (1997)
15. Meila, M., Shi, J.: Learning Segmentation by Random Walks. Advances in Neural Information Processing Systems (2001)
16. Fischer, I., Poland, J.: Amplifying the block matrix structure for spectral clustering. In: Proceedings of the 14th Annual Machine Learning Conference of Belgium and the Netherlands, pp. 21–28 (2005)
17. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, Chichester (1990)
18. Karypis, G., Kumar, V.: METIS: Unstructured Graph Partitioning and Sparse Matrix Ordering System. Technical Report, Department of Computer Science, University of Minnesota (1995)

Online Feature Selection Algorithm with Bayesian ℓ_1 Regularization

Yunpeng Cai[1], Yijun Sun[2], Jian Li[1], and Steve Goodison[3]

[1] Department of Electrical and Computer Engineering
[2] Interdisciplinary Center for Biotechnology Research
[3] Department of Surgery, University of Florida, Gainesville, FL 32610 USA*

Abstract. We propose a novel online-learning based feature selection algorithm for supervised learning in the presence of a huge amount of irrelevant features. The key idea of the algorithm is to decompose a nonlinear problem into a set of locally linear ones through local learning, and then estimate the relevance of features globally in a large margin framework with ℓ_1 regularization. Unlike batch learning, the regularization parameter in online learning has to be tuned on-the-fly with the increasing of training data. We address this issue within the Bayesian learning paradigm, and provide an analytic solution for automatic estimation of the regularization parameter via variational methods. Numerical experiments on a variety of benchmark data sets are presented that demonstrate the effectiveness of the newly proposed feature selection algorithm.

1 Introduction

High-throughput technologies now routinely produce large datasets characterized by unprecedented numbers of features. This seriously undermines the performance of many data analysis algorithms in terms of their speed and accuracy. Accordingly, across various scientific disciplines, there has been a surge in demand for efficient feature selection methods for high-dimensional data. Not only can its proper design enhance classification performance and reduce system complexity, but it can also provide significant insights into the nature of the problems under investigation in many applications.

Feature selection for high-dimensional data is considered one of the current challenges in statistical machine learning [1]. Existing algorithms are traditionally categorized as wrapper or filter methods, with respect to the criterion used to search for relevant features [2]. One major issue with wrapper methods is their high computational complexity. Many heuristic algorithms (e.g., forward and backward selection [3]) have been proposed to alleviate this issue. However, due to the heuristic nature, none of them can provide any guarantee of optimality. In the presence of many thousands of features, a hybrid approach is usually adopted, wherein the number of features is first reduced by using a filter method, and then a wrapper method is used on the reduced feature set. Nevertheless, it still may take several hours to perform the search, depending on the classifier used in a wrapper method. Embedded methods [4] have recently received

* This work is supported in part by the Komen Breast Cancer Foundation under grant No. BCTR0707587. Please address all correspondence to sunyijun@biotech.ufl.edu.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 401–413, 2009.
© Springer-Verlag Berlin Heidelberg 2009

an increasing interest. In contrast to wrapper methods, embedded methods incorporate feature selection directly into the learning process of a classifier. A feature weighting strategy is usually adopted that uses real-valued numbers, instead of binary ones, to indicate the relevance of features in a learning process. This strategy has several advantages. For example, there is no need to pre-specify the number of relevant features. Also, standard optimization techniques can be used to avoid combinatorial search. Consequently, embedded methods are usually computationally more tractable than wrapper methods. Yet, computational complexity is still a major issue when the number of features becomes excessively large.

We recently developed a new feature selection algorithm, referred to as LOFE (LOcal learning based FEature selection), that addresses several major issues with existing methods [5]. The key idea is to decompose an arbitrary nonlinear problem into a set of locally linear ones through local learning, and then estimate the relevance of features globally in a large margin framework with ℓ_1 regularization. The algorithm is computationally very efficient. It allows one to process many thousands of features within a few minutes on a personal computer, yet maintains a very high accuracy that is nearly insensitive to a growing number of irrelevant features. Theoretical analysis suggests that the algorithm have a logarithmical sample complexity with respect to data dimensionality [5]. That is, the number of samples needed to maintain the same level of learning accuracy grows only logarithmically with respect to data dimensionality.

In this paper, we extend LOFE for online learning, where data arrives sequentially and the estimate of feature relevance is improved with new data without total recalculation. While it has an increasing demand in many real-time systems [6,7], the issue of online learning for feature selection is rarely addressed in the literature. We develop a new online learning algorithm by using stochastic approximation. The algorithm does not make any assumption on data distributions, and thus is applicable for general problems. One major challenge in designing online learning algorithms is the estimation of model parameters, specifically the regularization parameter in our case. Unlike batch learning where the parameter can be estimated through cross-validation by analyzing entire training data, online learning has to automatically tune the parameter on-the-fly with the increasing of training data. We address this issue within the Bayesian learning paradigm, and provide an analytic solution for automatic estimation of the regularization parameter via variational methods. Numerical experiments based on a variety of benchmark data sets are presented that demonstrate the effectiveness of the newly proposed algorithm.

2 Local Learning Based Feature Selection Algorithm

This section presents a review of the LOFE algorithm. Let $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$ denote a training dataset, where $\boldsymbol{x}_n \in \mathbb{R}^M$ is the n-th data sample and $y_n \in \{0, 1\}$ is its corresponding class label. We are interested in the problems where $M \gg N$. We start by defining the margin. Given a distance function, we find two nearest neighbors of each sample $\boldsymbol{x}_n$, one from the same class (called *nearest hit* or NH), and the other from the different class (called *nearest miss* or NM) [8]. Following the work of [4], the margin of $\boldsymbol{x}_n$ is defined as $\rho_n = d(\boldsymbol{x}_n, \mathrm{NM}(\boldsymbol{x}_n)) - d(\boldsymbol{x}_n, \mathrm{NH}(\boldsymbol{x}_n))$, where $d(\cdot)$ is the distance

function. For the purpose of this paper, we use the block distance to define a sample's margin and nearest neighbors, while other standard definitions may also be used. An intuitive interpretation of this margin is a measure as to how much $\boldsymbol{x}_n$ can "move" in the feature space before being misclassified. By the large margin theory [9], a classifier that minimizes a margin-based error function usually generalizes well on unseen test data. One natural idea then is to scale each feature, thus obtaining a weighted feature space, parameterized by a nonnegative vector $\boldsymbol{w}$, so that a margin-based error function in the *induced* feature space is minimized. The margin of $\boldsymbol{x}_n$, computed with respect to $\boldsymbol{w}$, is given by

$$\rho_n(\boldsymbol{w}) = d(\boldsymbol{x}_n, \mathrm{NM}(\boldsymbol{x}_n)|\boldsymbol{w}) - d(\boldsymbol{x}_n, \mathrm{NH}(\boldsymbol{x}_n)|\boldsymbol{w}) \ . \tag{1}$$

By defining $\boldsymbol{z}_n = |\boldsymbol{x}_n - \mathrm{NM}(\boldsymbol{x}_n)| - |\boldsymbol{x}_n - \mathrm{NH}(\boldsymbol{x}_n)|$, where $|\cdot|$ is an element-wise absolute operator, $\rho_n(\boldsymbol{w})$ can be simplified as $\rho_n(\boldsymbol{w}) = \boldsymbol{w}^T\boldsymbol{z}_n$, which is a linear function of $\boldsymbol{w}$ and has the same form as the sample margin defined in SVM using a kernel function. An important difference, however, is that by construction the magnitude of each element of $\boldsymbol{w}$ in the above margin definition reflects the relevance of the corresponding feature in a learning process. This is not the case in SVM except when a linear kernel is used, which however can capture only linear discriminant information. Note that the margin thus defined requires only information about the neighborhood of $\boldsymbol{x}_n$, while no assumption is made about the underlying data distribution. This implies that by local learning we can transform an *arbitrary* nonlinear problem into a set of locally linear problems.

The local linearization of a nonlinear problem enables us to estimate the feature weights by using a linear model that has been extensively studied in the literature. The main problem with the above margin definition, however, is that the nearest neighbors of a given sample are unknown before learning. In the presence of thousands of irrelevant features, the nearest neighbors defined in the original space can be completely different from those in the induced space. To account for the uncertainty in defining local information, we develop a probabilistic model where the nearest neighbors of a given sample are treated as *latent variables*. Following the principles of the expectation-maximization algorithm [10], we estimate the margin through taking the expectation of $\rho_n(\boldsymbol{w})$ by averaging out the latent variables:

$$\begin{aligned}\bar{\rho}_n(\boldsymbol{w}) &= \boldsymbol{w}^T\left(\sum_{i\in\mathcal{M}_n} P(\boldsymbol{x}_i{=}\mathrm{NM}(\boldsymbol{x}_n)|\boldsymbol{w})|\boldsymbol{x}_n - \boldsymbol{x}_i| - \sum_{i\in\mathcal{H}_n} P(\boldsymbol{x}_i{=}\mathrm{NH}(\boldsymbol{x}_n)|\boldsymbol{w})|\boldsymbol{x}_n - \boldsymbol{x}_i|\right) \\ &= \boldsymbol{w}^T\bar{\boldsymbol{z}}_n \ ,\end{aligned}$$

where $\mathcal{M}_n = \{i : 1 \le i \le N, y_i \ne y_n\}$, $\mathcal{H}_n = \{i : 1 \le i \le N, y_i = y_n, i \ne n\}$, $P(\boldsymbol{x}_i{=}\mathrm{NM}(\boldsymbol{x}_n)|\boldsymbol{w})$ and $P(\boldsymbol{x}_i{=}\mathrm{NH}(\boldsymbol{x}_n)|\boldsymbol{w})$ are the probabilities that sample $\boldsymbol{x}_i$ is the nearest miss or hit of $\boldsymbol{x}_n$, respectively. These probabilities are estimated through the standard kernel density estimation method:

$$P(\boldsymbol{x}_i = \mathrm{NM}(\boldsymbol{x}_n)|\boldsymbol{w}) = \frac{k(\|\boldsymbol{x}_n - \boldsymbol{x}_i\|_{\boldsymbol{w}})}{\sum_{j\in\mathcal{M}_n} k(\|\boldsymbol{x}_n - \boldsymbol{x}_j\|_{\boldsymbol{w}})}, \forall i{\in}\mathcal{M}_n \tag{2}$$

$$P(\boldsymbol{x}_i = \mathrm{NH}(\boldsymbol{x}_n)|\boldsymbol{w}) = \frac{k(\|\boldsymbol{x}_n - \boldsymbol{x}_i\|_{\boldsymbol{w}})}{\sum_{j\in\mathcal{H}_n} k(\|\boldsymbol{x}_n - \boldsymbol{x}_j\|_{\boldsymbol{w}})}, \forall i{\in}\mathcal{H}_n \tag{3}$$

where $k(\cdot)$ is a kernel function. Specifically, we use exponential kernel $k(d)=\exp(-d/\delta)$ where kernel width δ determines the resolution at which the data is locally analyzed.

After the margins are defined, the problem of learning feature weights can be directly solved within a margin framework. For computational convenience, we perform the estimation in the logistic regression formulation. In applications with a huge amount of features (e.g., molecular classification [11]), we expect that most of features are irrelevant. To encourage the sparseness, one commonly used strategy is to add ℓ_1 penalty of $\boldsymbol{w}$ to an objective function [12,13], which leads to the following optimization problem:

$$\min_{\boldsymbol{w}} \frac{1}{N}\sum_{n=1}^{N} \ln\left(1+\exp(-\boldsymbol{w}^T \bar{\boldsymbol{z}}_n)\right) + \lambda\|\boldsymbol{w}\|_1 \quad \text{s.t.} \quad \boldsymbol{w} \geq 0, \tag{4}$$

where λ is a parameter that controls the penalty strength and consequently the sparseness of the solution.

Since $\bar{\boldsymbol{z}}_n$ implicitly depends on $\boldsymbol{w}$ through the probabilities $P(\boldsymbol{x}_i{=}\mathrm{NH}(\boldsymbol{x}_n)|\boldsymbol{w})$ and $P(\boldsymbol{x}_i = \mathrm{NM}(\boldsymbol{x}_n)|\boldsymbol{w})$, we use a fixed-point iteration method to solve for $\boldsymbol{w}$. In each iteration, $\bar{\boldsymbol{z}}_n$ is first computed by using the previous estimate of $\boldsymbol{w}$, which is then updated by solving the optimization problem (4). The iterations are carried out until convergence. It is interesting to note that though local learning is a highly nonlinear process, in each iteration we deal with a linear model.

For fixed $\bar{\boldsymbol{z}}_n$, (4) is a constrained convex optimization problem. Due to the nonnegative constraint on $\boldsymbol{w}$, it cannot be solved directly by using a gradient descent method. To overcome this difficulty, we reformulate the problem slightly as:

$$\min_{\boldsymbol{v}} \frac{1}{N}\sum_{n=1}^{N} \ln\left(1+\exp\left(-\sum_m v_m^2 \bar{z}_{n(m)}\right)\right) + \lambda\|\boldsymbol{v}\|_2^2\,, \tag{5}$$

thus obtaining an unconstrained optimization problem. Here v_m is the m-th element of $\boldsymbol{v}$. It is easy to show that at the optimum solution we have $w_m = v_m^2, 1 \leq m \leq M$. The solution of $\boldsymbol{v}$ can be readily found through gradient descent with a simple update rule:

$$\boldsymbol{v} \leftarrow \boldsymbol{v} - \eta\left(\lambda\mathbf{1} - \frac{1}{N}\sum_{n=1}^{N} \frac{\exp(-\sum_m v_m^2 \bar{z}_{n(m)})}{1+\exp(-\sum_m v_m^2 \bar{z}_{n(m)})}\bar{\boldsymbol{z}}_n\right) \otimes \boldsymbol{v}\,, \tag{6}$$

where $\otimes$ is Hadamard operator, and η is the learning rate determined by a line search.

Note that the objective function (5) is no longer a convex function, and thus a gradient descent method may find a local minimizer or a saddle point. It can be shown that if all elements of a initial point are non-zero, the solution obtained when the gradient vanishes is a global minimizer [5]. Moreover, by using the Banach fixed point theorem [14], it can be proved that the algorithm converges to a unique solution for any nonnegative initial feature weights, under a loose condition that a kernel width is sufficiently large [5]. It is interesting to note that even if the initial feature weights were wrongly selected and the algorithm started computing erroneous nearest neighbors for each sample, the theorem assures that the algorithm will eventually converge to the same solution obtained when one had *perfect* prior knowledge.

The computational complexity of LOFE is $\mathcal{O}(N^2M)$, which is linear with respect to feature dimensionality. In contrast, some popular greedy search methods (e.g., forward search) require of the order of $\mathcal{O}(M^2)$ moves in feature space.

3 Online Learning

LOFE is based on batch learning – that is, feature weights are updated after seeing all of the training data. In case the amount of training data is enormous, or training data arrives sequentially, online learning is computationally much more attractive than batch learning.

We propose a new online learning algorithm by using stochastic approximation techniques [15]. The key idea is to estimate the gradient of the objective function of individual samples, and then perform one gradient-descent step to obtain a solution that reduces the objective function. The theory of stochastic gradient assures that by using a carefully selected step size, the algorithm converges to a fixed-point solution identical to that obtained by using batch learning.

To distinguish sequentially arrived samples from those used in batch learning, we use k, instead of n, to index samples in the sequel. At the k-th sampling, we approximate the gradient of the objective function of (5) as

$$\boldsymbol{Q}(\boldsymbol{v}_k) = \left(\lambda\mathbf{1} - \sigma\left(-\boldsymbol{w}^T\bar{\boldsymbol{z}}_k\right)\bar{\boldsymbol{z}}_k\right) \otimes \boldsymbol{v} \ , \tag{7}$$

where $\sigma(x)$ is the sigmoid function, defined as $\sigma(x) = 1/(1+\exp(-x))$. The stochastic gradient method gives the following updating rule for $\boldsymbol{v}$:

$$\boldsymbol{v}_{k+1} = \boldsymbol{v}_k - \alpha_k\boldsymbol{Q}(\boldsymbol{v}_k) \ , \tag{8}$$

where $\alpha_k = \eta/k$, and η is a fixed step size. In each updating step, the vector $\bar{\boldsymbol{z}}_k$ is calculated from the k-th training samples based solely on the current value of $\boldsymbol{v}_k$.

4 Bayesian ℓ_1 Regularization with Variational Methods

Although the above online-learning formulation is straightforward, the estimation of the regularization parameter is not trivial. Unlike batch learning where the parameter can be estimated through cross validation by analyzing the full data set, online learning has to determine the parameter on-the-fly with the increasing of training data. This issue is rarely addressed in the literature.

4.1 Bayesian Estimation of Regularization Parameters

It has been recently suggested that parameter estimation can be performed by applying full Bayesian treatment to a parametric model, which is also called evidence approximation in the literature [16,17]. The basic idea is to connect a hyper-parameter to the prior distribution of a model, and then select a parameter that makes the model most consistent to the observed data. We below give a brief review of evidence approximation.

Bayesian learning treats a penalized loss function of the form $\mathcal{L} = \mathcal{L}_D + \lambda\mathcal{L}_W$ as the log-likelihood of the following posterior distribution

$$p(\boldsymbol{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w}|\lambda)}{p(\mathcal{D}|\lambda)} = \frac{1}{\mathcal{Z}}\exp(-\mathcal{L}_D)\exp(-\lambda\mathcal{L}_W)\ , \tag{9}$$

where $\mathcal{D}$ is the training data, and $\mathcal{Z}$ is some normalization constant. Thus, the empirical loss $\mathcal{L}_D$ is mapped to the likelihood function $p(\mathcal{D}|\boldsymbol{w})$, and the regularization term $\mathcal{L}_W$ to the prior distribution $p(\boldsymbol{w}|\lambda)$ of $\boldsymbol{w}$.

For the ℓ_1 regularization, $p(\boldsymbol{w}|\lambda) \propto \exp(-\lambda\|\boldsymbol{w}\|_1)$. The prior distribution $p(\boldsymbol{w}|\lambda)$ is usually chosen to be the exponential distribution or Laplace distribution, depending on the range of $\boldsymbol{w}$. Since in our case $\boldsymbol{w} \geq \mathbf{0}$, we use the isotropic exponential distribution for $\boldsymbol{w}$, given by

$$p(\boldsymbol{w}|\lambda) = \lambda^M \exp(-\lambda\|\boldsymbol{w}\|_1)\ . \tag{10}$$

In evidence approximation, $p(\mathcal{D}|\lambda) = \int p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w}|\lambda)d\boldsymbol{w}$ is called the evidence function. By assuming a prior distribution $p(\lambda)$, evidence approximation calculates the posterior distribution as

$$p(\lambda|\mathcal{D}) = \frac{p(\mathcal{D}|\lambda)p(\lambda)}{\int p(\mathcal{D}|\lambda)p(\lambda)d\lambda}\ ,$$

and picks the most probable value λ_{MAP} that maximizes the *a posteriori* distribution. It has been suggested that one can assume λ to be sharply peaked near λ_{MAP} [18,19], and maximize $p(\mathcal{D}|\lambda)$ to obtain λ_{MAP}. In this paper we adopt this simplification.

Obtaining a closed-form expression of a evidence function that allows for direct maximization is difficult. Approximation methods are usually adopted to simplify the optimization. Two commonly used approaches to approximating a probability distribution function are Laplace approximation [18] and variational methods [16,20]. It has been shown in [20] that variational methods usually produce more accurate results than Laplace approximation. Moreover, Laplace approximation is only applicable at local optimal points of a function, while variational methods can be used to approximate a function at arbitrary points. For the purpose of this paper, we use variational methods.

4.2 Variational Methods

Variational methods seek a local approximation of a convex or concave function via its first order Taylor expansion. For example, for a concave function $f(x)$, its Taylor expansion is expressed by $y(x|\xi) = f(\xi) + f'(\xi)(x-\xi)$, and $y(x|\xi) \geq f(x)$ with equality at $x = \xi$. By varying y with ξ, $f(x) = \min_\xi y(x|\xi)$. Denoting $\zeta = f'(\xi)$, we have $f(x) = \min_\zeta(\zeta x + H(\zeta))$.

For the sigmoid function, $\ln\sigma(x)$ is a concave function. Hence, $\ln\sigma(x) = \min_\zeta(\zeta x + H(\zeta))$, where $H(\zeta) = \zeta\ln\zeta + (1-\zeta)\ln(1-\zeta)$, and thus $\sigma(x) = \min_\zeta \exp(\zeta x + H(\zeta))$.

By using variational methods, we represent an objective function with a simple linear approximation, with equality at designated point $\xi = x$. Also, variational methods transform a univariate function $f(x)$ to a bivariate one $y(x|\xi)$, thus introducing extra degrees of freedom to the original problem we aim to solve. Denote $\mathfrak{F}(y(x), \alpha)$ as a *functional* of $y(x)$ with parameter α. The problem of optimizing $\mathfrak{F}(y(x), \alpha)$ with respect to α is transformed into a problem of optimizing an approximated functional

$\mathfrak{F}(y(x|\xi), \xi, \alpha)$ with respect to ξ and α individually. The later one is often mathematically more tractable.

4.3 Parameter Estimation

We apply variational methods to the objective function (4) to obtain its evidence function, and then estimate the regulariation parameter by maximizing the so-obtained evidence function. Although our goal is to derive a parameter estimation method for online learning, we first work on the batch learning case, and then extend the result to the online learning case.

We first rewrite Eq. (4) into the standard form of logistic regression by multiplying it by sample size N, which yields:

$$\min_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}) = \sum_{n=1}^{N} \ln\left(1 + \exp(-\boldsymbol{w}^T \bar{\boldsymbol{z}}_n)\right) + \tilde{\lambda}\|\boldsymbol{w}\|_1 \quad \text{s.t.} \quad \boldsymbol{w} \geq 0, \tag{11}$$

where $\tilde{\lambda} = N\lambda$. For notational simplicity, in the following we still use λ, instead of $\tilde{\lambda}$, when discussing batch learning, unless otherwise specified. We also define $\boldsymbol{\phi}_n = \bar{\boldsymbol{z}}_n$ if $y_n = 1$, and $\boldsymbol{\phi}_n = -\bar{\boldsymbol{z}}_n$ if $y_n = 0$. The likelihood function $p(\mathcal{D}|\boldsymbol{w})$ can then be expressed by

$$p(\mathcal{D}|\boldsymbol{w}) = \prod_{n=1}^{N} \sigma(\boldsymbol{w}^T\boldsymbol{\phi}_n)^{y_n}(1 - \sigma(\boldsymbol{w}^T\boldsymbol{\phi}_n))^{1-y_n} = \prod_{n=1}^{N} \exp(\boldsymbol{w}^T\boldsymbol{\phi}_n y_n)\sigma(-\boldsymbol{w}^T\boldsymbol{\phi}_n). \tag{12}$$

By using the variational approximation of the sigmoid function and Eqs. (10) and (12), the evidence function is given by

$$p(\mathcal{D}|\lambda) = \int_{\boldsymbol{w}\geq\boldsymbol{0}} \lambda^M e^{-\lambda\|\boldsymbol{w}\|_1 + \sum_{n=1}^{N} \boldsymbol{w}^T\boldsymbol{\phi}_n y_n} \cdot \min_{\{\zeta_1,\cdot,\zeta_N\}} \left\{ e^{\sum_{n=1}^{N}\left(-\boldsymbol{w}^T\boldsymbol{\phi}_n\zeta_n + H(\zeta_n)\right)} \right\} d\boldsymbol{w}. \tag{13}$$

Note that the minimization is inside the integrate operation, which makes the optimization mathematically intractable. Following the principles of variational methods [20], also explained in Sec. 4.2, we treat the parameters $\boldsymbol{\zeta} = [\zeta_1, \cdots, \zeta_N]$ to be independent of $\boldsymbol{w}$ so that we can move the minimization out of the integration. Integrating out Eq. (13) yields

$$p(\mathcal{D}|\lambda) = \min_{\{\zeta_1,\cdot,\zeta_N\}} \left\{ \lambda^M \exp\left(\sum_{n=1}^{N} H(\zeta_n)\right) \prod_{m=1}^{M} \frac{1}{\left(\lambda + \sum_{n=1}^{N}(\zeta_n - y_n)\phi_{n(m)}\right)} \right\}. \tag{14}$$

Denote $\boldsymbol{\zeta}^*$ as the variational parameters that minimize the likelihood function (14). By using Eqs. (10), (12) and (14), the posterior distribution of $\boldsymbol{w}$ can then be written as

$$p(\boldsymbol{w}|\mathcal{D}) = \prod_{m=1}^{M} \left(\lambda + \sum_{n=1}^{N}(\zeta_n^* - y_n)\phi_{n(m)}\right) \exp\left(-\boldsymbol{w}^T(\lambda\mathbf{1} + \sum_{n=1}^{N}(\zeta_n^* - y_n)\boldsymbol{\phi}_n)\right), \tag{15}$$

which is a non-isotropic exponential distribution. Denote $\bar{\boldsymbol{w}}$ as the mean of $\boldsymbol{w}$, the m-th element of which is

$$\bar{w}_{(m)} = \left(\lambda + \sum_{n=1}^{N}(\zeta_n^* - y_n)\phi_{n(m)}\right)^{-1}. \tag{16}$$

We then seek the optimal estimates of the variational parameters $\boldsymbol{\zeta}$ of $p(\mathcal{D}|\lambda)$. Taking the derivative of the logarithm function of $p(\mathcal{D}|\lambda)$ with respect to ζ_n and forcing it to zero produce

$$\frac{\partial \ln p(\mathcal{D}|\lambda)}{\partial \zeta_n} = \ln\left(\frac{\zeta_n}{1-\zeta_n}\right) - \bar{\boldsymbol{w}}^T\boldsymbol{\phi}_n = 0. \tag{17}$$

Hence, for a fixed λ, the optimal estimates of $\boldsymbol{\zeta}$ are given by

$$\zeta_n^* = \sigma(\bar{\boldsymbol{w}}^T\boldsymbol{\phi}_n), \quad n = 1, \cdots, N. \tag{18}$$

It is easy to prove that Eq. (18) has an unique solution and can be solved either by an iterative or Newton's methods.

After obtaining an approximation to the evidence function $p(\mathcal{D}|\lambda)$, we are able to estimate the hyper-parameter λ by maximizing the evidence function. The logarithm of Eq. (14) takes the form

$$\ln p(\mathcal{D}|\lambda) = -\sum_{m=1}^{M}\ln\left(\lambda + \sum_{n=1}^{N}(\zeta_n^*(\lambda) - y_n)\phi_{n(m)}\right) + \sum_{n=1}^{N}H(\zeta_n^*(\lambda)) + M\ln\lambda. \tag{19}$$

Since $\bar{\boldsymbol{w}}$, given by Eq. (16), is a function of λ, we denote it by $\bar{\boldsymbol{w}}(\lambda)$ in the sequel. Taking the derivative of Eq. (19) with respect to λ and forcing it to zero, we obtain the following iterated solution of λ, given by

$$\lambda = \left(\frac{1}{M}\sum_{m=1}^{M}\bar{w}(\lambda)_{(m)}\right)^{-1}. \tag{20}$$

With Eqs. (18) and (20), we are now able to determine the optimal choice of regularization parameter λ by using an iterative method.

We now proceed to extend the above derivation for online learning. Using k, instead of N, to denote the sample size and applying $\tilde{\lambda} = k\lambda$ to Eq. (20) yield

$$k\lambda = \left(\frac{1}{M}\sum_{m=1}^{M}(k\lambda + \sum_{n=1}^{k}(\zeta_n^* - y_n)\phi_{n(m)})^{-1}\right)^{-1}. \tag{21}$$

Define the k-th estimate of the weight mean $\bar{\boldsymbol{w}}$ as $\bar{\boldsymbol{w}}_k$. Using $\lambda = \tilde{\lambda}/k$ in Eq. (16) yields

$$\bar{w}_{k(m)} = \left(k\lambda + \sum_{n=1}^{k}(\zeta_n^* - y_n)\phi_{n(m)}\right)^{-1}. \tag{22}$$

Hence, the k-th estimate of λ can be calculated as

$$\lambda_k = \left(\frac{1}{M} \sum_{m=1}^{M} k \bar{w}_{k(m)} \right)^{-1}, \tag{23}$$

where $\bar{w}_{k(m)}$ can be computed in an online manner

$$\begin{aligned} (\bar{w}_{k(m)})^{-1} &= (\bar{w}_{k-1(m)})^{-1} + (u_{k(m)})^{-1}, \\ u_{k(m)} &= (\lambda_{k-1} + (\zeta_k^* - y_k)\phi_{k(m)})^{-1}. \end{aligned} \tag{24}$$

Note that Eq. (14) makes sense only if $\bar{w}_{(m)} > 0$. If this condition is violated, it implies that the currently estimated λ does not provide sufficient penalty to force the feature weights to zero, and the so-estimated feature weights do not follow the exponential distribution. In this case, we artificially set $\bar{w}_{k(m)} = 0$ to increase λ in the next iteration.

5 Experiments

This section presents several numerical experiments to demonstrate the effectiveness of the newly proposed algorithm. We first perform a simulation study on the well-known Fermat's spiral data. It is a binary classification problem. Each class has 230 samples distributed in a two-dimensional space, forming a spiral shape. In addition to the first two relevant features, each sample is contaminated by a varying number of irrelevant features randomly sampled from the standard normal distribution. The spiral problem, though simple, when contaminated by thousands of irrelevant features, poses a serious challenge for existing algorithms. Fig. 1 presents the feature weights learned by our algorithm and three competing algorithms, including SIMBA [4], RELIEF-F [8], and I-RELIEF [21]. We observe that our algorithm performs remarkably well over a wide range of feature-dimensionality values, yielding always the largest weights for the first two relevant features, while all other weights are less than 10^{-4}. The solution is nearly insensitive to a growing number of irrelevant features. In contrast, the three competing algorithms perform substantially worse than ours.

In the second experiment, we apply our algorithm to eight UCI benchmark data sets. The data information is summarized in Table 1. For each data set, the set of original features is augmented by 5000 artificially generated irrelevant features randomly sampled from the standard normal distribution. It should be noted that some features in the

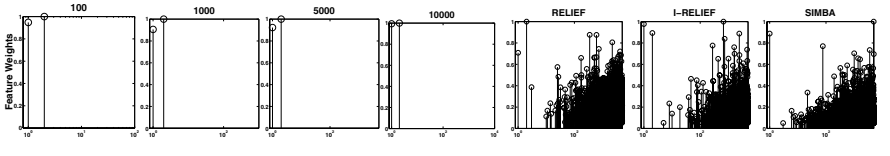


Fig. 1. Feature weights learned on spiral data containing a varying numbers of irrelevant features, ranging from 100 to 10000, by our algorithm (left 4 columns), and by three competing algorithms (with 5000 irrelevant features). The x- and y-axis represent the number of features and the values of feature weights, respectively. Only the first two features are useful.

Table 1. Summary of UCI data sets. The number of irrelevant features artificially added to the original ones is indicated in parentheses.

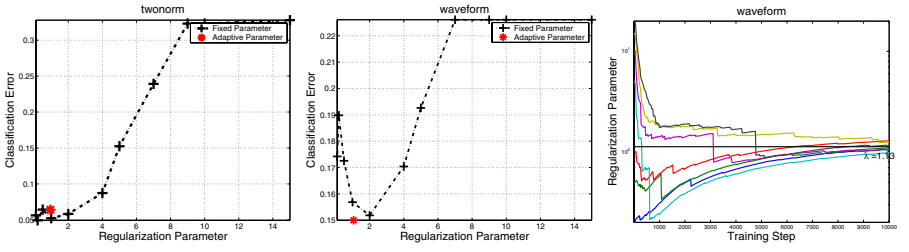
Data set	Training	Test	Feature	Data set	Training	Test	Feature
twonorm	400	7000	20(5000)	*waveform*	400	4600	21(5000)
flare-solar	666	400	9(5000)	*thyroid*	140	75	5(5000)
diabetes	468	300	8(5000)	*heart*	170	100	13(5000)
splice	1000	2175	60(5000)	*breast-cancer*	200	77	9(5000)

Table 2. Classification accuracy and standard deviation of SVM performed on UCI data sets using all features, original features, and features selected by our algorithms. False discovery rate (FDR) and CPU time (measured on Core2 2.0G PC) of each algorithm are also reported.

Data set	Original Features	All Features	Batch Learning			Online Learning		
	Acc(%)	Acc(%)	Acc(%)	FDR(‰)	Time(s)	Acc(%)	FDR(‰)	Time(s)
twonorm	97.4 ± 0.2	50.0 ± 0.0	97.4 ± 0.2	0	162	94.5 ± 1.2	0.4	49
waveform	89.9 ± 0.6	67.1 ± 0.2	88.3 ± 0.9	1.6	157	85.1 ± 2.5	2.7	39
diabetes	77.3 ± 1.7	65.7 ± 1.9	76.3 ± 1.1	1.8	267	74.1 ± 2.9	0.8	48
breast-cancer	63.7 ± 6.1	69.3 ± 4.5	69.5 ± 4.6	2.2	68	70.3 ± 5.6	3.1	18
thyroid	95.3 ± 2.1	69.9 ± 3.6	94.7 ± 2.4	0.1	13	91.5 ± 3.0	0.1	15
heart	80.8 ± 3.5	54.4 ± 3.4	82.8 ± 4.2	0.4	73	80.2 ± 4.3	2.6	18
splice	88.1 ± 0.8	51.5 ± 1.7	90.5 ± 1.2	1.1	2639	88.3 ± 3.0	0.3	167
flare-solar	65.3 ± 1.7	55.9 ± 1.5	62.0 ± 2.8	0.6	948	63.7 ± 5.1	0.8	76

original feature sets may be irrelevant or weakly relevant, and hence may receive zero weights in our algorithm. Unlike the spiral problem, however, the relevance information of the original features is unknown. To verify that our algorithm indeed identify all relevant features, we set a high standard by comparing the classification performance of SVM (with the RBF kernel) in two cases: (1) when only the original features are used (i.e., without 5000 useless features), and (2) when the features selected by our algorithm are used. It is well known that SVM is very robust against noise, and that the presence of a few irrelevant features in the original feature sets should not significantly affect its performance. Hence, the classification performance of SVM in the first case should be very close to that of SVM performed on the optimal feature subsets that are unknown to us *a priori*. Essentially, we are comparing our algorithm with the *optimal* feature selection algorithm. If SVM performs similarly in both cases, we may conclude that our algorithm achieves close-to-optimum solutions.

The structural parameters of SVM are estimated through ten-fold cross validation using training data. For both online and batch learning algorithms, the kernel width δ used in Eqs. (2) and (3) is set to 2. Though not considered in this paper, the kernel width can also be treated as a hyper-parameter and estimated similarly using the proposed algorithm. For the batch learning algorithm, the regularization parameter λ is set to 1. For the online learning algorithm, the regularization parameter is estimated by using the Bayesian parameter estimation algorithm proposed in Sec. 4. To reduce statistical variations, both batch and online learning algorithms are run 10 times for each dataset. In each run, a dataset is randomly partitioned into training and test sets. After a feature weight vector is learned, only the features with weights larger than 10^{-4} are used for classification. The averaged classification errors and standard deviations of SVM are



(a) (b) (c)

Fig. 2. (a-b) Classification performance of online learning algorithm using different but fixed regularization parameters, and the parameter tuned by Bayesian estimation on *twonorm* and *waveform* data sets, and (c) the convergence paths of lambda in one sample run

reported in Table 2. The false discovery rate (FDR), defined as the ratio between the number of artificially added, irrelevant features identified by our algorithms as useful ones and the total number of irrelevant features (i.e., 5000), is reported in Table 2. For reference, the classification performance of SVM using all features (i.e., the original features plus the useless ones) is also reported. From these experimental results, we observe the followings:

(1) SVM using all features performs poorly, while SVM using the features identified by our batch learning algorithm performs similarly or even slightly better (e.g., *breast-cancer* and *splice*) than SVM using the original features. This suggests that our batch learning algorithm can achieve a close-to-optimum solution in the presence of a huge number of irrelevant features. This result is consistent with that reported in Fig. 1.

(2) Online learning performs slightly worse than batch learning, but with a much lower computational complexity. For most data sets, it only takes batch learning a few minutes to process more than 5000 features. Note however that the CPU times of batch learning on *splice* and *flare-solar* are much larger than other data sets. This is due to the fact that the computational complexity of batch learning is quadratic in the number of samples. We should emphasize that though batch learning performs slightly better than online learning, they are used in different scenarios.

(3) In addition to successfully identifying relevant features, both batch and online learning algorithms perform remarkably well in removing irrelevant ones. From Table 2, we observe that for both algorithms, the false discovery rates are very low. For example, for *splice*, there are only less than 4 out of 5000 irrelevant features that are identified by our algorithms as useful ones.

In the third experiment, we compare the performance of the online learning algorithm using different but fixed regularization parameters, and the parameter tuned by Bayesian estimation on the *twonorm* and *waveform* data sets. The results are shown in Fig. 2(a-b). For both data sets, the classification errors are heavily influenced by the choice of λ. We also observe that with the tuned regularization parameter the performance of the algorithm is very close to the optimal one that can be achieved by using a fixed parameter. This result clearly demonstrates the effectiveness of our proposed parameter estimation method.

We also conduct an experiment to study the convergence behavior of the proposed parameter estimation algorithm by applying the online learning algorithm to *waveform* with different initial values of λ. The learning paths of the parameter are depicted in Fig. 2(c). We observe that the regularization parameter converges regardless of its initial points.

6 Conclusion

This paper addressed the issue of finding sparse solutions for large-scale feature selection problems, and developed a computationally efficient method that is applicable for both online and batch learning applications. The batch learning version exhibits a near optimal performance with affordable computational complexity, and the online learning one provides a means to balance between speed and accuracy. We also proposed a Bayesian regularization method for online learning that performs very well with the specified feature selection algorithm.

References

1. Lafferty, J., Wasserman, L.: Challenges in statistical machine learning. Stat. Sinica 16, 307–322 (2006)
2. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res 3, 1157–1182 (2003)
3. Pudil, P., Novovicova, J.: Novel methods for subset selection with respect to problem knowledge. IEEE Intell. Sys. 13(2), 66–74 (1998)
4. Gilad-Bachrach, R., Navot, A., Tishby, N.: Margin based feature selection - theory and algorithms. In: Proc. 21st Int. Conf. Mach. Learn., pp. 43–50 (2004)
5. Sun, Y., Todorovic, S., Goodison, S.: A feature selection algorithm capable of handling extre -mely large data dimensionality. In: Proc. 8th SIAM Conf. Data Mining, pp. 530–540 (2008)
6. Collins, R.T., Liu, Y., Leordeanu, M.: Online selection of discriminative tracking features. IEEE Trans. Pattern Anal. Mach. Intell. 27(10), 1631–1643 (2005)
7. Jiang, W., Er, G., Dai, Q., Gu, J.: Similarity-based online feature selection in content-based image retrieval. IEEE Trans. Image Proc. 15(3), 702–712 (2006)
8. Kira, K., Rendell, L.: A practical approach to feature selection. In: Proc. 9th Int. Conf. Mach. Learn., pp. 249–256 (1992)
9. Vapnik, V.: Statistical Learning Theory. Wiley, Chichester (1998)
10. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. Ser. B 39(1), 1–38 (1977)
11. Sun, Y., Goodison, S., Li, J., Liu, L., Farmerie, W.: Improved breast cancer prognosis through the combination of clinical and genetic markers. Bioinformatics 23(1), 30–37 (2007)
12. Ng, A.Y.: Feature selection, L1 vs. L2 regularization, and rotational invariance. In: Proc. 21st Int. Conf. Mach. Learn., pp. 78–86 (2004)
13. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Ser. B 58(1), 267–288 (1996)
14. Kress, R.: Numerical Analysis. Springer, Heidelberg (1998)
15. Spall, J.C.: Introduction to Stochastic Search and Optimization: Estimation, Simulation and Control. John Wiley, Chichester (2003)

16. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Heidelberg (2006)
17. MacKay, D.J.: The evidence framework applied to classification networks. Neural Comp. 4(5), 720–736 (1992)
18. Cawley, G.C., Talbot, N.L.: The evidence framework applied to sparse kernel logistic regression. Neurocomputing 64, 119–135 (2005)
19. MacKay, D.J.: Bayesian interpolation. Neural Comp. 4(3), 415–447 (1992)
20. Jaakkola, T.S., Jordan, M.I.: Bayesian parameter estimation via variational methods. Stat. and Comp. 10, 25–37 (2000)
21. Sun, Y.: Iterative RELIEF for feature weighting: Algorithms, theories, and applications. IEEE Trans. Pattern Anal. Mach. Intell. 29, 1035–1051 (2007)

Feature Selection for Local Learning Based Clustering

Hong Zeng and Yiu-ming Cheung

Department of Computer Science, Hong Kong Baptist University, Hong Kong SAR, China
{hzeng,ymc}@comp.hkbu.edu.hk

Abstract. For most clustering algorithms, their performance will strongly depend on the data representation. In this paper, we attempt to obtain better data representations through feature selection, particularly for the Local Learning based Clustering (LLC) [1]. We assign a weight to each feature, and incorporate it into the built-in regularization of LLC algorithm to take into account of the relevance of each feature for the clustering. Accordingly, the weights are estimated iteratively with the clustering. We show that the resulting weighted regularization with an additional constraint on the weights is equivalent to a known sparse-promoting penalty, thus the weights for irrelevant features can be driven towards zero. Experiments on several benchmark datasets demonstrate the effectiveness of the proposed method.

1 Introduction

In many pattern recognition and data mining problems, the data is usually represented by a huge number of features. However, the abundance of features often makes the distinction among patterns much harder and less accurate. For instance, in genomic data analysis, it is not uncommon to have thousands of gene expression coefficients as features for a single sample, but only a small faction is discriminant among different tissue classes. Those irrelevant features for the prediction may largely degrade the performance of a learning algorithm. Therefore, developing an effective feature selection algorithm, whose goal is to identify those features relevant to the inference task at hand, is highly necessary for achieving a good performance.

In the literature, most feature selection algorithms have been developed for supervised learning, rather than the unsupervised learning. It is believed that the unsupervised feature selection is more challenging due to the absence of ground truth class labels that can guide the search for relevant features. Until very recently, several algorithms have been proposed to address this issue for clustering. In general, they can be categorized as the *filter* and *wrapper* methods. The *filter* approaches [2,3,4] leave out uninformative ones before the clustering. They demonstrate great computational efficiency because they do not involve clustering when evaluating the feature quality. However, the issue of determining how many relevant features should be selected is rarely addressed in the literature, thus may cause difficulty in practical applications [5]. The *wrapper* approaches [5,6,7,8,9] first construct a candidate feature subset, and then assess its goodness by investigating the performance of a specific clustering on this subset according to some criteria, the two steps are repeated till convergence. In general, the *wrapper* approaches are computationally demanding, but they are expected to be

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 414–425, 2009.
© Springer-Verlag Berlin Heidelberg 2009

more accurate than the *filter* ones, due to the performance feedback of the clustering. Some *wrapper* approaches, e.g. [6,7], employ the greedy (i.e., non-exhaustive) search through the whole space of all feature subsets, thus cannot guarantee all the relevant features are selected. This drawback, as well as the issue of determining how many features to be selected for *filter* approaches, can be alleviated by assigning each feature a nonnegative weight [5,8,9], rather than a binary indicator, to indicate its relevance to the clustering. By casting the feature selection as an estimation problem, the combinatorial explosion of the search space can be avoided as well. Nevertheless, all the existing algorithms [5,8,9] are constructed by global models built with all the available samples, they may get degenerated on high-dimensional datasets because the similarities among samples may be less discriminable from global view.

In this paper, we propose a novel method of the *wrapper* manner, which perform the feature selection in a Local Learning based Clustering (LLC) [1]. The LLC algorithm searches for a solution that ensures the cluster labels in the neighborhood of each point are as pure as possible. Therefore, when the samples seem resembling each other as a whole as in the high dimensional space, searching for more similar samples locally may help obtaining a more reliable intermediate clustering result to aid the feature selection. Furthermore, through extending the ridge regression, a supervised method, to unsupervised learning, LLC has a built-in regularization for the model complexity [1]. Consequently, such a built-in regularization is modified in this paper, in order to take into account the relevance of each feature or kernel for the clustering. We show the equivalence between the modified penalty term and existing sparse-promoting one subject to some constraints, thus the resulting weights for features are guaranteed to be very sparse. Eventually, the feature weights are estimated iteratively with the local learning based clustering.

The remainder of this paper is organized as follows: Section 2 overviews of local learning based clustering algorithm. We present the proposed feature selection method in Section 3. Furthermore, we prove the sparse norm equivalence in Section 4. In Section 5, experimental results on several benchmark datasets are reported. Finally, the paper is concluded in Section 6.

2 Overview of the Local Learning Based Clustering Algorithm

The indicator matrix that will be used later is introduced first. Given n data points $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n (\mathbf{x}_i \in \mathbb{R}^d)$, the dataset will be partitioned into C clusters. The clustering result can be represented by a *cluster assignment indicator matrix* $\mathbf{P} = [p_{ic}] \in \{0,1\}^{n \times C}$, such that $p_{ic} = 1$ if $\mathbf{x}_i$ belongs to the cth cluster, and $p_{ic} = 0$ otherwise. The *scaled cluster assignment indicator matrix* used in this paper is defined by: $\mathbf{Y} = \mathbf{P}(\mathbf{P}^T\mathbf{P})^{-\frac{1}{2}} = [\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^C]$, where $\mathbf{y}^c = [y_{1c}, \ldots, y_{nc}]^T \in \mathbb{R}^n (1 \le c \le C)$, is the c-th column of $\mathbf{Y} \in \mathbb{R}^{n \times C}$. $y_{ic} = p_{ic}/\sqrt{n_c}$ can be regarded as the confidence that $\mathbf{x}_i$ is assigned to the cth cluster, where n_c is the size of the cth cluster. It is easy to verify that $\mathbf{Y}^T\mathbf{Y} = \mathbf{I}$, where $\mathbf{I} \in \mathbb{R}^{n \times n}$ is the identity matrix.

The starting point of the LLC [1] is that the cluster assignments in the neighborhood of each point should be as pure as possible. Specifically, it assumes that the cluster indicator value at each point should be well estimated by a regression model trained

locally with its neighbors and their cluster indicator values. Suppose an arbitrary $\mathbf{Y}$ exists at first. For each $\mathbf{x}_i$, the model is built with the training data $\{(\mathbf{x}_j, y_{jc})\}_{\mathbf{x}_j \in \mathcal{N}_i} (1 \leq c \leq C, 1 \leq i, j \leq n)$, where $\mathcal{N}_i$ denotes the set of neighboring[1] points of $\mathbf{x}_i$ (not including $\mathbf{x}_i$ itself). The output of the local model is of the following form: $f_i^c(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\theta}_i^c, \forall \mathbf{x} \in \mathbb{R}^d$, where $\boldsymbol{\theta}_i^c \in \mathbb{R}^d$ is the vector of local regression coefficients. Here, the bias term is ignored for simplicity, assuming that one of the features is always 1. In [1], $\boldsymbol{\theta}_i^c$ is solved by:

$$\min_{\boldsymbol{\theta}_i^c} \sum_{c=1}^{C} \sum_{i=1}^{n} \Big[\sum_{\mathbf{x}_j \in \mathcal{N}_i} \beta (y_{jc} - \mathbf{x}_j^T \boldsymbol{\theta}_i^c)^2 + \|\boldsymbol{\theta}_i^c\|^2 \Big], \tag{1}$$

where β is a trade-off parameter. Denoting the solution to the linear ridge regression problem (1) as $\boldsymbol{\theta}_i^{c*}$, the predicted cluster assignment for the test data $\mathbf{x}_i$ can then be calculated by:

$$\widehat{y}_{ic} = f_i^c(\mathbf{x}_i) = \mathbf{x}_i^T \boldsymbol{\theta}_i^{c*} = \boldsymbol{\alpha}_i^T \mathbf{y}_i^c, \tag{2}$$

and

$$\boldsymbol{\alpha}_i^T = \beta \mathbf{x}_i^T (\beta \mathbf{X}_i \mathbf{X}_i^T + \mathbf{I})^{-1} \mathbf{X}_i. \tag{3}$$

where $\mathbf{X}_i = [\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, \ldots, \mathbf{x}_{i_{n_i}}]$ with $\mathbf{x}_{i_k}$ being the k-th neighbor of $\mathbf{x}_i$, n_i is the size of $\mathcal{N}_i$, and $\mathbf{y}_i^c = [y_{i_1 c}, y_{i_2 c}, \ldots, y_{i_{n_i} c}]^T$.

After all the local predictors (2) have been constructed, LLC aims to find an optimal cluster indicator matrix $\mathbf{Y}$ that minimizes the overall prediction error:

$$\begin{aligned} &\sum_{c=1}^{C} \sum_{i=1}^{n} (y_{ic} - \widehat{y}_{ic})^2 = \sum_{c=1}^{C} \|\mathbf{y}^c - \mathbf{A}\mathbf{y}^c\|^2 = trace[\mathbf{Y}^T (\mathbf{I} - \mathbf{A})^{\mathbf{T}} (\mathbf{I} - \mathbf{A}) \mathbf{Y}] \\ &= trace(\mathbf{Y}^T \mathbf{T} \mathbf{Y}), \end{aligned} \tag{4}$$

where $\mathbf{T} = (\mathbf{I} - \mathbf{A})^{\mathbf{T}} (\mathbf{I} - \mathbf{A})$, $\mathbf{A}$ is an $n \times n$ sparse matrix with its (i, j)-th entry a_{ij} being the corresponding element in $\boldsymbol{\alpha}_i$ by (3) if $\mathbf{x}_j \in \mathcal{N}_i$ and 0 otherwise.

Similar to the spectral clustering [10,11], $\mathbf{Y}$ is relaxed into the continuous domain while keeping the property $\mathbf{Y}^T \mathbf{Y} = \mathbf{I}$ for (4). LLC then solves:

$$\min_{\mathbf{Y} \in \mathbb{R}^{n \times C}} \quad trace(\mathbf{Y}^T \mathbf{T} \mathbf{Y}) \;\; s.t. \; \mathbf{Y}^{\mathbf{T}} \mathbf{Y} = \mathbf{I}. \tag{5}$$

A solution to $\mathbf{Y}$ is given by the first C eigenvectors of the matrix $\mathbf{T}$, corresponding to the first C smallest eigenvalues. Similar to [10,11], the final partition result is obtained by discretizing $\mathbf{Y}$ via the method in [11] or by k-means as in [10].

3 Feature Selection for Local Learning Based Clustering

It can be seen that the key part of LLC is to learn a local regression model and then predict the cluster assignment for each point by (2). Thus one may easily observe that the

[1] The k-mutual neighbors are adopted in order to well describe the local structure, i.e. $\mathbf{x}_j$ is considered as a neighbor of $\mathbf{x}_i$ only if $\mathbf{x}_i$ is also one of the k-nearest neighbors of $\mathbf{x}_j$.

inclusion of noisy features in the input vector might ruin the inner product in (2). Consequently, the prediction error may not be able to truly reflect the purity measurement of cluster assignment in each neighborhood. Hence, we need to exclude those irrelevant features from the prediction.

To this end, we introduce a binary feature selection vector $\boldsymbol{\tau} = [\tau_1, \tau_2, \ldots, \tau_d], \tau_l \in \{0, 1\}$ to the local discriminant function as follows:

$$f_i^c(\mathbf{x}) = \mathbf{x}^T diag(\sqrt{\boldsymbol{\tau}})\boldsymbol{\theta}_i^c + b_i^c = \sum_{l=1}^{d} x_l \sqrt{\tau_l}(\boldsymbol{\theta}_i^c)_l + b_i^c, \tag{6}$$

where $diag(\sqrt{\boldsymbol{\tau}}) \in \mathbb{R}^{d\times d}$ is a diagonal matrix with $\sqrt{\boldsymbol{\tau}} \in \mathbb{R}^d$ on the diagonal. The reason for using the square root on $\boldsymbol{\tau}$ will become clear later, but now it does not change anything because $\tau_l \in \{0, 1\}$. $b_i^c \in \mathbb{R}$ is the bias term. Then in (6), the entries of the input vector $\mathbf{x}$ could be turned on and off depending on the value of τ_l. To avoid a combinatorial search for $\boldsymbol{\tau}$ later, we relax the constraint $\tau_l \in \{0, 1\}$ to $\tau_l \geq 0$ and further restrict its scale by $\sum_{l=1}^{d} \tau_l = 1$, giving them the flavor of probabilities. Consequently the local discriminant function can be solved by:

$$\min_{\boldsymbol{\theta}_i^c, b_i^c} \sum_{c=1}^{C}\sum_{i=1}^{n}\Big[\sum_{\mathbf{x}_j\in\mathcal{N}_i} \beta(y_{jc} - \mathbf{x}_j^T diag(\sqrt{\boldsymbol{\tau}})\boldsymbol{\theta}_i^c - b_i^c)^2 + \boldsymbol{\theta}_i^{cT}\boldsymbol{\theta}_i^c\Big], \tag{7}$$

or equivalently, the following problem:

$$\min_{\mathbf{w}_i^c, b_i^c} \sum_{c=1}^{C}\sum_{i=1}^{n}\Big[\sum_{\mathbf{x}_j\in\mathcal{N}_i} \beta(y_{jc} - \mathbf{x}_j^T \mathbf{w}_i^c - b_i^c)^2 + \mathbf{w}_i^{cT} diag(\boldsymbol{\tau}^{-1})\mathbf{w}_i^c\Big], \tag{8}$$

which is obtained by applying a change of variables $diag(\sqrt{\boldsymbol{\tau}})\boldsymbol{\theta}_i^c \rightarrow \mathbf{w}_i^c$. The local model is now tantamount to be of the following form:

$$f_i^c(\mathbf{x}) = \mathbf{x}^T \mathbf{w}_i^c + b_i^c, \tag{9}$$

and $\mathbf{w}_i^c$ is now regularized with a weighted l_2 norm: $\mathbf{w}_i^{cT} diag(\boldsymbol{\tau}^{-1})\mathbf{w}_i^c = \sum_l \frac{(\mathbf{w}_i^c)_l^2}{\tau_l}$, i.e., the second term in the square bracket of (8). Hence a small value for τ_l, which is expected to be associated with an irrelevant feature, will result in a large penalization on $(\mathbf{w}_i^c)_l$ by this weighted norm. In particular, when $\tau_l = 0$, we will prove later that it will lead to $(\mathbf{w}_i^c)_l = 0 \ \ \forall i, c$ [2]. Under such circumstance, the l-th feature will be completely eliminated from the prediction. Subsequently, an improved clustering result can be expected. To perform feature selection in LLC, we develop an alternating update algorithm to estimate the clustering captured in $\mathbf{Y}$ and the feature weight $\boldsymbol{\tau}$.

3.1 Update Y for a Given $\boldsymbol{\tau}$

First, the nearest neighbors $\mathcal{N}_i$ should be re-found according to the $\boldsymbol{\tau}$-weighted square Euclidean distance, i.e.:

$$d_\tau(\mathbf{x}_1, \mathbf{x}_2) = \|\mathbf{x}_1 - \mathbf{x}_2\|_\tau^2 = \sum_{l=1}^{d} \tau_l (\mathbf{x}_1^{(l)} - \mathbf{x}_2^{(l)})^2. \tag{10}$$

[2] In this paper, we will use the convention that $\frac{z}{0} = 0$ if $z = 0$ and ∞ otherwise.

With fixed feature weight $\boldsymbol{\tau}$, analytic solutions for problem (8) can be then easily obtained by setting the derivatives to zero. The solution are:

$$\mathbf{w}_i^c = \beta[\beta \mathbf{X}_i \mathbf{\Pi}_i \mathbf{X}_i^T + diag(\boldsymbol{\tau}^{-1})]^{-1} \mathbf{X}_i \mathbf{\Pi}_i \mathbf{y}_i^c, \tag{11}$$

$$b_i^c = \frac{1}{n_i} \mathbf{e}_i^T (\mathbf{y}_i^c - \mathbf{X}_i^T \mathbf{w}_i^c), \tag{12}$$

where $\mathbf{e}_i = [1\ 1 \cdots 1]^T \in \mathbb{R}^{n_i}$, $\mathbf{\Pi}_i = \mathbf{I}_i - \frac{1}{n_i}\mathbf{e_i}\mathbf{e_i^T}$ is a centering projection matrix, satisfying $\mathbf{\Pi}_i \mathbf{\Pi}_i = \mathbf{\Pi}_i$, and $\mathbf{I}_i$ is an n_i-by-n_i unit matrix.

For high dimensional data, the matrix inversion in (11) will be very computational inefficient whose time complexity is $O(d^3)$. Fortunately, by applying the Woodbury's matrix inversion lemma, we could get:

$$\begin{aligned}\mathbf{w}_i^c = {} & \beta diag(\boldsymbol{\tau}) \mathbf{X}_i \mathbf{\Pi}_i \big[\mathbf{I}_i - \\ & (\beta^{-1}\mathbf{I}_i + \mathbf{\Pi}_i \mathbf{X}_i^T diag(\boldsymbol{\tau}) \mathbf{X}_i \mathbf{\Pi}_i)^{-1} \mathbf{\Pi}_i \mathbf{X}_i^T diag(\boldsymbol{\tau}) \mathbf{X}_i \mathbf{\Pi}_i \big] \mathbf{y}_i^c.\end{aligned} \tag{13}$$

Then the time complexity of the matrix inversion in (13) is only $O(n_i^3)$. In general $n_i \ll d$, thus the computational cost can be considerably reduced. Besides, from (13), it can be seen that $(\mathbf{w}_i^c)_l$ $(\forall i, c)$ goes to 0 as the feature weight τ_l vanishes.

Subsequently the predicted cluster assignment confidence for $\mathbf{x}_i$ will be obtained as follows: $\widehat{y}_{ic} = \mathbf{x}_i^T \mathbf{w}_i^c + b_i^c = \boldsymbol{\alpha}_i^T \mathbf{y}_i^c$, with

$$\boldsymbol{\alpha}_i^T = \beta(\mathbf{k}_i^\tau - \frac{1}{n_i}\mathbf{e}_i^T \mathbf{K}_i^\tau)\mathbf{\Pi}_i \big[\mathbf{I}_i - (\beta^{-1}\mathbf{I}_i + \mathbf{\Pi}_i \mathbf{K}_i^\tau \mathbf{\Pi}_i)^{-1} \mathbf{\Pi}_i \mathbf{K}_i^\tau \mathbf{\Pi}_i\big] + \frac{1}{n_i}\mathbf{e}_i^T, \tag{14}$$

where we define $\mathbf{k}_i^\tau = \mathbf{x}_i^T diag(\boldsymbol{\tau}) \mathbf{X}_i$, and $\mathbf{K}_i^\tau = \mathbf{X}_i^T diag(\boldsymbol{\tau}) \mathbf{X}_i$.

As in LLC, we construct the key matrix $\mathbf{T}$ by (14) and (4). To solve the same optimization problem in (5), the columns of $\mathbf{Y}$ are simply set to the first C eigenvectors of $\mathbf{T}$ corresponding to the smallest C eigenvalues.

3.2 Update $\boldsymbol{\tau}$ for a Given $\mathbf{Y}$

With fixed $\mathbf{Y}$ and neighborhood determined at each point, a reasonable $\boldsymbol{\tau}$ is the one that can lead to a better local regression model, which is characterized by a lower objective value at the minimum in (8). We will apply this criterion to re-estimate $\boldsymbol{\tau}$. Firstly, we remove the bias term by plugging (12) into (8), then it is rewritten as follows:

$$\min_{\mathbf{w}_i^c} F(\mathbf{w}_i^c, \boldsymbol{\tau}) = \sum_{c=1}^{C} \sum_{i=1}^{n} \Big[\beta \|\mathbf{\Pi}_i \mathbf{y}_i^c - (\mathbf{X}_i \mathbf{\Pi}_i)^T \mathbf{w}_i^c\|^2 + \mathbf{w}_i^{cT} diag(\boldsymbol{\tau}^{-1}) \mathbf{w}_i^c\Big]. \tag{15}$$

Subsequently, the estimation of $\boldsymbol{\tau}$ is reformulated as follows:

$$\min_{\boldsymbol{\tau}} \mathcal{P}(\boldsymbol{\tau}), \ \ s.t. \sum_{l=1}^{d} \tau_l = 1, \tau_l \geq 0, \forall l, \tag{16}$$

where $\mathcal{P}(\boldsymbol{\tau}) = F(\mathbf{w}_i^{c*}, \boldsymbol{\tau})$, with $\mathbf{w}_i^{c*} = \arg\min_{\mathbf{w}_i^c} F(\mathbf{w}_i^c, \boldsymbol{\tau})$ given in (13). Then the Lagrange of (16) is:

$$\mathcal{L}(\boldsymbol{\tau}, \lambda, \boldsymbol{\varepsilon}) = \mathcal{P}(\boldsymbol{\tau}) + \lambda(\sum_{l=1}^{d} \tau_l - 1) - \sum_{l=1}^{d} \varepsilon_l \tau_l, \tag{17}$$

where λ and $\boldsymbol{\varepsilon}$ are Lagrangian multipliers, and $\lambda \geq 0$ is a scalar, $\boldsymbol{\varepsilon} \geq \mathbf{0}$ is a vector. The derivative of $\mathcal{L}$ with respect to τ_l, $(l = 1, \ldots, d)$ is computed as:

$$\frac{\partial \mathcal{L}}{\partial \tau_l} = \frac{\partial \mathcal{P}}{\partial \tau_l} + \lambda - \varepsilon_l, \tag{18}$$

where

$$\begin{aligned} \frac{\partial \mathcal{P}}{\partial \tau_l} &= \frac{\partial F(\mathbf{w}_i^c, \tau)}{\partial \tau_l}\Big|_{\mathbf{w}_i^c = \mathbf{w}_i^{c*}} + \sum_l \frac{\partial (\mathbf{w}_i^{c*})_l}{\partial \tau_l} \underbrace{\frac{\partial F(\mathbf{w}_i^c, \tau)}{\partial (\mathbf{w}_i^c)_l}\Big|_{\mathbf{w}_i^c = \mathbf{w}_i^{c*}}}_{0} \\ &= -\frac{\sum_{c=1}^{C} \sum_{i=1}^{n} (\mathbf{w}_i^{c*})_l^2}{\tau_l^2}. \end{aligned} \tag{19}$$

At the optimality, we therefore have:

$$\tau_l^2 = \frac{\sum_{c=1}^{C} \sum_{i=1}^{n} (\mathbf{w}_i^{c*})_l^2}{\lambda - \varepsilon_l}, \quad \forall l, \tag{20}$$

$$\lambda \geq 0, \varepsilon_l \geq 0, \tau_l \geq 0, \quad \forall l, \tag{21}$$

$$\sum_{l}^{d} \tau_l = 1, \tag{22}$$

$$\varepsilon_l \tau_l = 0, \quad \forall l. \tag{23}$$

By using the Karush-Kuhn-Tucker (KKT) condition, i.e., (23), it is easy to verify the following two cases: Case 1: $\sum_{c=1}^{C} \sum_{i=1}^{n} (\mathbf{w}_i^{c*})_l^2 = 0 \Rightarrow \tau_l = 0$; Case 2: $\sum_{c=1}^{C} \sum_{i=1}^{n} (\mathbf{w}_i^{c*})_l^2 \neq 0 \Rightarrow \varepsilon_l = 0$ and $\tau_l = \sqrt{\sum_{c=1}^{C} \sum_{i=1}^{n} (\mathbf{w}_i^{c*})_l^2} / \sqrt{\lambda}$. Together with (22), it follows that the optimal solution of $\boldsymbol{\tau}$ can be calculated in a closed form:

$$\tau_l = \frac{\sqrt{\sum_{c=1}^{C} \sum_{i=1}^{n} (\mathbf{w}_i^{c*})_l^2}}{\sum_{m=1}^{d} \sqrt{\sum_{c=1}^{C} \sum_{i=1}^{n} (\mathbf{w}_i^{c*})_m^2}}. \tag{24}$$

It can be seen from (24) that a feature is unimportant if the corresponding element in the regression coefficients has neglectable magnitude for all the clusters at each point.

3.3 The Complete Algorithm

The complete local learning based clustering algorithm with feature selection (denoted as LLC-fs) is described in Algorithm 1. The loop stops when the relative variation of the trace value in (5) between two consecutive iterations is below a threshold (we set it at 10^{-2} in this paper), indicating the partitioning has almost stabilized. Then $\mathbf{Y}$ is discretized to obtain the final clustering result with k-means as in [10].

input : $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$, size of the neighborhood k, trade-off parameter β
output: $\mathbf{Y}, \boldsymbol{\tau}$

Initialize $\tau_l = \frac{1}{d}$, for $l = 1, \ldots, d$;
while *not converge* **do**
    Find k-mutual neighbors for $\{\mathbf{x}_i\}_{i=1}^n$, using the metric defined in (10);
    Construct the matrix $\mathbf{T}$ in (4) with $\boldsymbol{\alpha}_i$ given in (14), and solve (5) to obtain $\mathbf{Y}$;
    Compute $\mathbf{w}_i^{c*}, \forall i, c$ by (13) and update $\boldsymbol{\tau}$ with (24);
end

Algorithm 1: Feature selection for local learning based clustering algorithm

4 Sparse Norm Equivalence

In this section, it will be shown that the weighted l_2 norm regularization associated with a l_1 norm constraint on these weights, is equivalent to a well-known sparse-promoting regularization penalty. We address this equivalence based upon the fact that the infimum of the weighted l_2 norm, with the weights defined on the standard simplex, is equal to a squared special l_1 norm regularization.

Theorem 1

$$\min_{\sum_l \tau_l = 1, \tau_l \geq 0} \sum_l \frac{\|\widetilde{\mathbf{W}}_l\|^2}{\tau_l} = \left(\sum_l \|\widetilde{\mathbf{W}}_l\| \right)^2,$$

where we define $\|\widetilde{\mathbf{W}}_l\| = \sqrt{\sum_{c=1}^C \sum_{i=1}^n (\mathbf{w}_i^c)_l^2}$.

Proof. From the Cauchy-Schwarz inequality we have $\sum_l \|\widetilde{\mathbf{W}}_l\| = \sum_l \tau_l^{\frac{1}{2}} \tau_l^{-\frac{1}{2}} \|\widetilde{\mathbf{W}}_l\| \leq (\sum_l \tau_l)^{\frac{1}{2}} (\sum_l \tau_l^{-1} \|\widetilde{\mathbf{W}}_l\|^2)^{\frac{1}{2}} \leq (\sum_l \tau_l^{-1} \|\widetilde{\mathbf{W}}_l\|^2)^{\frac{1}{2}}$, where equality is obtained when $\sum_l \tau_l = 1$ and $\frac{\|\widetilde{\mathbf{W}}_\mu\|}{\tau_\mu} = \frac{\|\widetilde{\mathbf{W}}_\nu\|}{\tau_\nu}, \forall \mu, \nu$. It is equivalent to requiring $\tau_l = \|\widetilde{\mathbf{W}}_l\| / \sum_\mu \|\widetilde{\mathbf{W}}_\mu\|$, which is always satisfied by (24).

In fact, $\sum_l \|\widetilde{\mathbf{W}}_l\|$ can be viewed as a combination of the l_1-norm regularization on the feature level and the l_2-norm regularization on the cluster level for all local models. In a simplified case, where the regression model is not built locally, i.e., omitting the subscript i, it will be $\sum_l \sqrt{\sum_c (\mathbf{w}^c)_l^2}$. It is just the so-called l_1/l_2 norm [12], which results in sparse solution at the feature level, i.e., $(\mathbf{w}^c)_\mu$ is close to $0, \forall c$ on some dimension μ. Hence, according to Theorem 1, the weighted l_2 norm regularization should be able to produce at least as sparse as that of the squared $\sum_l \|\widetilde{\mathbf{W}}_l\|$ penalty. Consequently, the sparsity of $\boldsymbol{\tau}$ follows from (24).

5 Experimental Results

Five benchmark datasets were used, whose characteristics are summarized in Table 1. In all the experiments, we evaluated the performance with the clustering accuracy (ACC) index [3]. On each dataset, we investigated whether the proposed LLC-fs could

Table 1. Characteristics of the datasets used in feature selection experiments

Dataset	#Dimension (d)	#Sample (n)	#Class (C)
USPS 4 vs.9	256	1673	2
USPS 0 vs.8	256	2261	2
colon cancer	2000	62	2
SRBCT	2308	63	4
breast cancer	3303	44	2

improve the LLC algorithm and the baseline k-means clustering, both of which assume all features are equally important. Furthermore, LLC-fs was compared with the state-of-the-art unsupervised feature selection method, Q-α algorithm[3] [5], which is also a *wrapper* approach with iterative eigen-decomposition and feature weight estimation, but is global learning based. Because the selection of the optimal number of clusters is beyond the scope of this paper, we simply set the number of clusters equal to the number of classes in each dataset for all the algorithms.

Both the LLC and the LLC-fs algorithms have two parameters: the size of the mutual neighbors k, and the trade-off parameter β. For each datasets, k and β were chosen from pre-specified candidate intervals, respectively. Then both LLC and LLC-fs were performed with each combination of k and β, and the whole process was repeated 10 times. For LLC, we only report the performance with the best parameter combinations (denoted as LLC-best$k\beta$). Q-α and *k-means* have no parameters, the mean and the standard deviation of the ACC index over 10 runs are presented. The parameter sensitivity study for the proposed LLC-fs algorithm will be given at the end of this section.

Handwritten Digit Datasets. In this case study, we focused on the task of clustering on the USPS ZIP code[4] handwritten digits, which are 16×16 grayscale images. Each image is thus represented by a 256-dimensional vector. In particular, we considered two pairs of digits: "4 vs. 9" and "0 vs. 8", which are known difficult to differentiate.

For the USPS datasets, k and β were chosen from $30 \sim 60$ and $[0.1, 10]$, respectively. k has been chosen in such an interval because there are on average more than 800 samples per cluster for both datasets, and there are known heavy overlappings within each pair, a larger neighborhood may help obtain a more accurate local prediction. For simplicity, only the mean and the standard deviation of the ACC index for LLC-fs, with the combination $k = 30, \beta = 1$ over 10 runs, are presented. The results are summarized in Table 2.

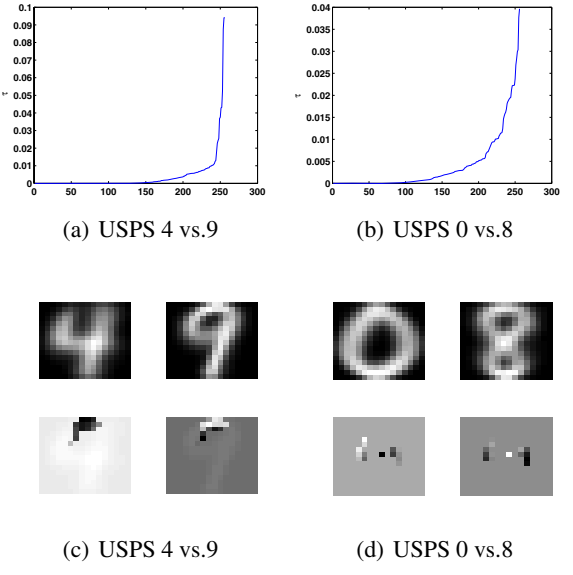
It can be seen from Table 2 that LLC-fs significantly improves the performance of other methods on both USPS datasets, whereas the global approach Q-α performs even worse than the baseline k-means clustering on the "4 vs.9" dataset that overlaps heavily. A plausible reason is that, when the points resemble to each other in a large scale due to the high dimensionality or too many features of similar values, a local search strategy

[3] Its MATLAB source code was obtained from the authors of [5].

[4] http://www-stat-class.stanford.edu/∼tibs/ElemStatLearn/data.html

Table 2. Accuracies of various methods on the 10 benchmark datasets

Dataset	k-means	LLC-best$k\beta$	Q-α	LLC-fs
USPS 4 vs.9	0.7764±0	0.7992	0.5686±0.0796	**0.9871±0.0003**
USPS 0 vs.8	0.8262±0	0.8859	0.8755±0.0008	**0.9858±0.0034**
colon cancer	0.6000±0.0847	0.6129	0.5306±0.0092	**0.7419±0**
SRBCT	0.5047±0.0380	0.5079	0.4174±0.0130	**0.6302±0.0107**
breast cancer	0.5454±0	0.5682	0.5273±0.0096	**0.6134±0**



(a) USPS 4 vs.9 (b) USPS 0 vs.8

(c) USPS 4 vs.9 (d) USPS 0 vs.8

Fig. 1. Unsupervised feature selection by LLC-fs with $k = 30$, $\beta = 1$ on the USPS digits. (a),(b): the (sorted) $\boldsymbol{\tau}$ values on the "4 vs. 9" and "0 vs. 8" datasets, respectively. In (c) and (d): the first row plots the class mean images; the second row shows the top 15 features in each mean image ranked by the $\boldsymbol{\tau}$ weight vector.

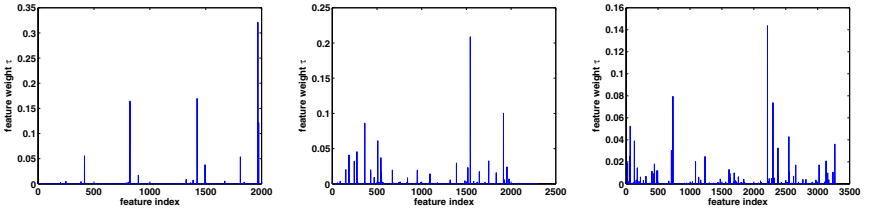
is expected to produce a more accurate intermediate clustering result than the global based counterpart, for aiding the feature selection.

To get a better understanding of what features have been ranked top by our weighting scheme, Figure 1 shows in the top features in the image domain. Firstly, the sorted $\boldsymbol{\tau}$'s in typical runs on the two datasets are presented in Figure 1(a),1(b) respectively. It can be seen that the both $\boldsymbol{\tau}$ vectors are sparse, and only few of the feature weights are above a very clear threshold. One can find that the 15 top-ranked features have covered almost all the strongly discriminative regions for each digit pairs (see Figure 1(c) and 1(d)), thus resulting in more accurate partitions.

Genomic Datasets. We studied the clustering on public gene expression datasets: colon cancer [13], SRBCT [14], and breast cancer [15].

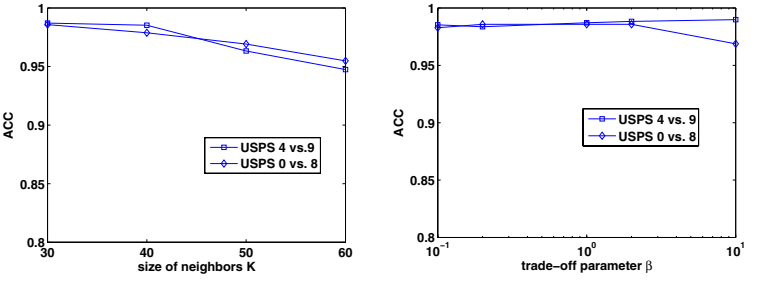
For these genomic data, the size of the mutual nearest neighbors k should be neither too small (it would be less accurate with deficient local training sets of very high-dimensionality) nor too large (we have limited samples). We chose the k from $20 \sim 40$ for all the data. β was still selected in $[0.1, 10]$. For clarity, we only report the results for LLC-fs with the combination $k = 30, \beta = 1$ over 10 runs.

It can be seen from Table 2 that the superiority over the compared algorithms is remarkable on these high dimensional datasets with very scarce samples. The typical feature weighting results in the 10 runs are also plotted in Figure 2, all with $k = 30, \beta =$

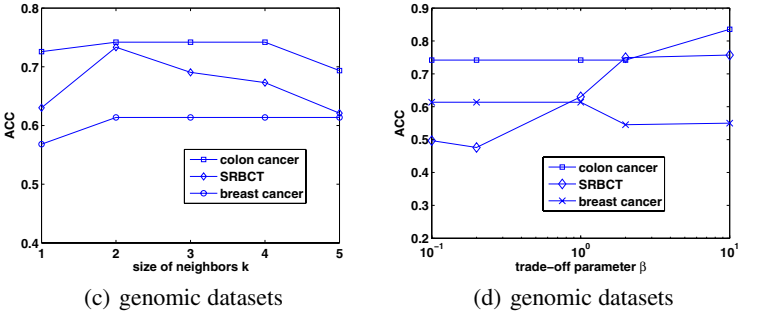


(a) colon cancer (b) SRBCT (c) breast cancer

Fig. 2. Feature weight vector τ's learned by LLS-fs with $k = 30, \beta = 1$ on the genomic datasets



(a) USPS datasets (b) USPS datasets



(c) genomic datasets (d) genomic datasets

Fig. 3. The parameter sensitivity studies for LLC-fs on the USPS datasets and genomic datasets. (a) (c) varying the size of neighborhood k with fixed $\beta = 1$; (b) (d) varying the trade-off parameter β with fixed $k = 30$. The values on each line represent the average ACC over 10 runs.

1. For each dataset, $\boldsymbol{\tau}$ is quite sparse, only few of them have significant magnitudes while most feature weights are close to zero. The reason why the LLC-fs significantly improves the performance of LLC on these data is that it can jointly identify these most relevant genes for clustering.

Parameter Sensitivity Study. The effects of parameter k and β in LLC-fs algorithm are presented in Figure 3, where each parameter is varied while the others are fixed.

In Figure 3(a) and 3(b), on these two USPS datasets which have more than 800 samples per cluster and of high dimensionality, the performance of our method does not vary much when the k and β are chosen from $30 \sim 60$ and $[0.1, 10]$, respectively. Figure 3(c) justifies our setting on of k on these genomic datasets, i.e., the performance is stable and satisfactory when k is neither too large nor too small. From Figure 3(d), it is observed that a large β generally would lead to better performance on genomic datasets. The explaination is that the weighted l_2 norm penalty will be very large in (8) because there are many irrelevant features, a large trade-off parameter β could balance the fitting error and the penalty.

6 Conclusion

In this paper, a novel feature selection method has been proposed for the local learning based clustering, where an optimal subset of features is jointly learned with the clustering. The proposed approach is developed within a regularization framework. The resulting feature weights are very sparse. Experimental results demonstrate that the proposed feature selection method is able to improve the accuracy of the basic local learning clustering. Furthermore, it generally outperforms the state-of-the-art counterpart, especially when the similarities among samples become less discriminable as in the high dimensional space.

Acknowledgement

This work was supported by the Faculty Research Grant of HKBU under Projects: FRG/07-08/II-54, and the Research Grant Council of Hong Kong SAR under Grant: HKBU210306.

References

1. Wu, M., Schölkopf, B.: A local learning approach for clustering. Advances in Neural Information Processing Systems 19, 1529–1536 (2007)
2. Dash, M., Choi, K., Scheuermann, P., Liu, H.: Feature selection for clustering-a filter solution. In: Proceedings of IEEE International Conference on Data Mining, pp. 115–122 (2002)
3. He, X., Cai, D., Niyogi, P.: Laplacian score for feature selection. Advances in Neural Information Processing Systems 18, 507–514 (2005)
4. Cheung, Y.M., Zeng, H.: Local kernel regression score for feature selection. IEEE Transactions on Knowledge and Data Engineering (in press, 2009)

5. Wolf, L., Shashua, A.: Feature selection for unsupervised and supervised inference: The emergence of sparsity in a weight-based approach. Journal of Machine Learning Research 6, 1855–1887 (2005)
6. Dy, J., Brodley, C.: Feature selection for unsupervised learning. Journal of Machine Learning Research 5, 845–889 (2004)
7. Law, M.H.C., Jain, A.K., Figueiredo, M.A.T.: Feature selection in mixture-based clustering. Advances in Neural Information Processing Systems 15, 609–616 (2003)
8. Roth, V., Lange, T.: Feature selection in clustering problems. Advances in Neural Information Processing Systems 16, 473–480 (2004)
9. Zeng, H., Cheung, Y.M.: Feature selection for clustering on high dimensional data. In: Proceedings of the Pacific Rim International Conference on Artificial Intelligence, pp. 913–922 (2008)
10. Ng, A., Jordan, M., Weiss, Y.: On spectral clustering: Analysis and an algorithm. Advances in Neural Information Processing Systems 14, 849–856 (2002)
11. Yu, S., Shi, J.: Multiclass spectral clustering. In: Proceedings of IEEE International Conference on Computer Vision, pp. 313–319 (2003)
12. Obozinski, G., Taskar, B., Jordan, M.: Multi-task feature selection. Technical Report (2006)
13. Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., Levine, A.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proceedings of the National Academy of Sciences 96, 6745–6750 (1999)
14. Khan, J., Wei, J., Ringnér, M., Saal, L., Ladanyi, M., Westermann, F., Berthold, F., Schwab, M., Antonescu, C., Peterson, C., et al.: Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. Nature Medicine 7, 673–679 (2001)
15. West, M., Blanchette, C., Dressman, H., Huang, E., Ishida, S., Spang, R., Zuzan, H., Olson Jr., J., Marks, J., Nevins, J.: Predicting the clinical status of human breast cancer by using gene expression profiles. In: Proceedings of the National Academy of Sciences, vol. 98, pp. 11462–11467 (2001)

RV-SVM: An Efficient Method for Learning Ranking SVM*

Hwanjo Yu, Youngdae Kim, and Seungwon Hwang

Department of Computer Science and Engineering,
Pohang University of Science and Technology (POSTECH),
Pohang, South Korea
{hwanjoyu,prayer,swhwang}@postech.ac.kr

Abstract. Learning *ranking (or preference)* functions has become an important data mining task in recent years, as various applications have been found in information retrieval. Among rank learning methods, ranking SVM has been favorably applied to various applications, e.g., optimizing search engines, improving data retrieval quality. In this paper, we first develop a 1-norm *ranking* SVM that is faster in *testing* than the standard ranking SVM, and propose Ranking Vector SVM (RV-SVM) that revises the 1-norm ranking SVM for faster *training*. The number of variables in the RV-SVM is significantly smaller, thus the RV-SVM trains much faster than the other ranking SVMs. We experimentally compared the RV-SVM with the state-of-the-art rank learning method provided in SVM-light. The RV-SVM uses much less support vectors and trains much faster for nonlinear kernels than the SVM-light. The accuracies of RV-SVM and SVM-light are comparable on relatively large data sets. Our implementation of RV-SVM is posted at http://iis.postech.ac.kr/rv-svm.

1 Introduction

Learning ranking functions has been a major issue in the data mining and machine learning communities and produced many applications in information retrieval [13, 10, 15,28,23]. Task of learning ranking is different from regression or classification in terms that the training data in ranking is relative ordering or *partial orders*. For instance, let "A is preferred to B" be denoted as "A $\succ$ B". For a dataset $D = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_m\}$, an example of partial orders is $\{(\boldsymbol{x}_4 \succ \boldsymbol{x}_2 \succ \boldsymbol{x}_3 \succ \boldsymbol{x}_5), (\boldsymbol{x}_2 \succ \boldsymbol{x}_1 \succ \boldsymbol{x}_6), ...\}$. Partial orders do not contain a class label nor real value assigned to each object, thus neither classification or regression methods can learn from partial orders.

Among rank learning methods (e.g., ranking SVM and RankBoost [13, 10]), ranking SVM has been favorably applied to various applications in information retrieval [13, 15, 23, 28, 29]. Ranking SVM learns a ranking function F from partial orders such that $F(\boldsymbol{x}_i) > F(\boldsymbol{x}_j)$ for any $\boldsymbol{x}_i \succ \boldsymbol{x}_j$. (We briefly discuss literatures related to ranking SVM in Section 2.)

This paper develops new methods for learning ranking functions. We first develop a 1-norm ranking SVM, a ranking SVM that is based on *1-norm* objective function.

* This work was supported by the Korea Research Foundation Grant funded by the Korean Government (KRF-2008-331-D00528).

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 426–438, 2009.
© Springer-Verlag Berlin Heidelberg 2009

(Standard ranking SVM is based on 2-norm objective function.) The 1-norm ranking SVM learns a function with much less support vectors than the standard SVM. Thereby, its testing time is much faster than 2-norm SVMs and provides better feature selection properties. (The function of 1-norm SVM is likely to utilize a less number of features by using a less number of support vectors [11].) Feature selection is also important in ranking. Ranking functions are relevance or preference functions in document or data retrieval. Identifying key features increases the interpretability of the function. Feature selection for nonlinear kernel is especially challenging, and the fewer the number of support vectors are, the more efficiently feature selection can be done [12, 20, 6, 31, 8].

Next, we propose another ranking machine, *Ranking Vector SVM (RV-SVM)*, that revises the 1-norm ranking SVM for fast training. The RV-SVM trains much faster than standard SVMs while not compromising the accuracy when the training set is relatively large. *The key idea of RV-SVM is to express the ranking function with "ranking vectors" instead of support vectors.* The support vectors in ranking SVMs are the pairwise difference vectors of the closest pairs [28, 29]. Thus, the training requires investigating every *data pair* as potential candidates of support vectors, and the number of data pairs are quadratic to the size of training set. On the other hand, the ranking function of the RV-SVM utilizes each training data object instead of data pairs. Thus, the number of variables for optimization is substantially reduced in the RV-SVM.

We experimentally compared the performance of RV-SVM with the state-of-the-art rank learning method provided in SVM-light [2]. RV-SVM trains much faster than SVM-light for nonlinear funtions. RV-SVM uses a substantially less number of support vectors than SVM-light. Their accuracies are comparable when the size of dataset is relatively large. Our implementation of RV-SVM is posted at http://iis.postech.ac.kr/rv-svm.

This paper is organized as follows. We first review related work and ranking SVM in Section 2 and 3. We present our methods, 1-norm ranking SVM and RV-SVM, in Section 4 and 5 respectively. We discuss our experiments in Section 6 and conclude our study.

2 Related Work

SVMs have been actively researched for classification and regression. Conventional classifying SVMs are based on 2-norm objective functions where the objective function minimizes 2-norm of the weight vector [25, 21, 5, 9, 14]. By minimizing 2-norm of the weight vector, SVMs maximize the margin, which corresponds to improving generalization of classification functions [25]. Training in the SVMs becomes a quadratic programming (QP) problem, and fast training algorithms have been developed for the 2-norm SVMs such as sequential minimal optimization (SMO) [21], reduced SVM [17], and cutting plane algorithm [16].

Learning ranking functions (relevance or preference functions) has recently gained much attention, and many methodologies have been proposed especially based on the SVM [13, 7] and the Ensemble approaches [10, 26, 22]. Ranking SVM is one of the earliest methodologies showing reliable performance, and it has been applied to many practical applications for information retrieval [15, 23, 28, 29, 7]. Ranking SVM is now typically used as the baseline method for rank learning research.

The formulation of ranking SVM is extended from that of the classifying SVM by replacing the data vectors with pairwise difference vectors [13]. This "simple" extension changes the meaning of the margin maximization into maximizing closest data pairs in ranking, which corresponds to improving the generalization of ranking [28].

This extension also makes it possible to implement the ranking SVM using the same decomposition algorithms (e.g., SMO [21]) of classifying SVMs [15]. However, due to the use of pairwise difference vectors in training, *the training complexity of ranking SVMs is inherently more expensive by two asymptotic orders of magnitude (with respect to data size) than that of classifying SVMs.* This critical disadvantage is unavoidable because the number of data pairs are quadratic to the number of data objects. Joachims published cutting plane algorithms for classifying and ranking SVMs but his methods are limited to linear (classifying or ranking) functions [16].

This is the first paper that introduces 1-norm *ranking* SVM and its extension that are not inherently confined by the disadvantage of the "double order complexity problem" discussed above.

3 Ranking SVM Overview

We overview ranking SVM in this section. The training set in the ranking SVM is denoted as $R = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_m, y_m)\}$ where $\boldsymbol{x}_i$ is a data point and y_i is an integer indicating the ranking of $\boldsymbol{x}_i$, such that $\boldsymbol{x}_i \succ_R \boldsymbol{x}_j$ when $y_i < y_j$. We say $\boldsymbol{x}_i \succ_R \boldsymbol{x}_j$ if a vector $\boldsymbol{x}_i$ is ranked higher than $\boldsymbol{x}_j$ in an order R. We assume for simplicity that R is strict ordering, which means that for all pairs $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in R, either $\boldsymbol{x}_i \succ \boldsymbol{x}_j$ or $\boldsymbol{x}_i \prec \boldsymbol{x}_j$. However, it can be straightforwardly generalized to weak ordering.

The goal of ranking SVMs is to learn function F that satisfies $F(\boldsymbol{x}_i) > F(\boldsymbol{x}_j)$ for all pairs of $\{(\boldsymbol{x}_i, \boldsymbol{x}_j) : y_i < y_j \in R\}$ where R is the training set, and also generalize beyond the training set. $F(\boldsymbol{x}_i)$ is a scoring function that returns a ranking score of $\boldsymbol{x}_i$. When F is a linear function ($= \boldsymbol{w} \cdot \boldsymbol{x}$), learning F becomes computing a weight vector, $\boldsymbol{w}$, such that, for most pairs $\{(\boldsymbol{x}_i, \boldsymbol{x}_j) : y_i < y_j \in R\}$,

$$F(\boldsymbol{x}_i) > F(\boldsymbol{x}_j) \implies \boldsymbol{w} \cdot \boldsymbol{x}_i > \boldsymbol{w} \cdot \boldsymbol{x}_j \quad (1)$$

$$\implies \boldsymbol{w}(\boldsymbol{x}_i - \boldsymbol{x}_j) > 0 \quad (2)$$

$$\implies F(\boldsymbol{x}_i - \boldsymbol{x}_j) > 0. \quad (3)$$

Note that ranking SVMs train a function from pairwise orderings that are generated from R. When the number of training vectors is m in R, the total number of *pairwise* orderings that can be generated is $\frac{m(m-1)}{2} = O(m^2)$.

OPT 1 (ranking SVM, *primal*).

$$\textit{minimize:} \quad L_p(w, \xi) = \tfrac{1}{2}||\boldsymbol{w}||^2 + C \sum_{ij}^{P} \xi_{ij} \quad (4)$$

$$\textit{s.t.:} \quad \boldsymbol{w}(\boldsymbol{x}_i - \boldsymbol{x}_j) \geq 1 - \xi_{ij}, \qquad \forall \{(\boldsymbol{x}_i, \boldsymbol{x}_j) : y_i < y_j \in R\} \quad (5)$$

$$\xi \geq 0 \quad (6)$$

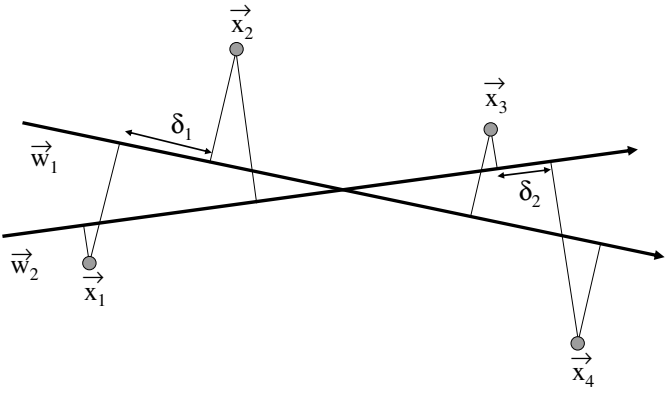


Fig. 1. Linear projection of four data points

This goal of ranking SVMs is expressed in OPT 1 (primal form) [13, 15]. OPT 1 suppresses $||\boldsymbol{w}||^2$, which corresponds to improve the generalization by maximizing the margin. P $(= \frac{m(m-1)}{2})$ is the number of the pairwise orderings in R, and there are P constraints for Eq.(5). Eq.(5) are to force Eq.(2). ξ_{ij} is a slack variable and C is a parameter controlling the tradeoff between the margin size and the amount of error. Note that this primal form is similar to that of classifying SVM except that the left side of Eq.(5) is $\boldsymbol{w} \cdot \boldsymbol{x}_i$ in the classifying SVM.

Since the support vectors in the classifying SVM are the vectors $\boldsymbol{x}_i$ that are closest to the decision boundary, the support vectors in the ranking SVM become the pairwise difference vectors $(\boldsymbol{x}_i - \boldsymbol{x}_j)$ that satisfy the constraint (5) with equality sign, that are *the closest pairwise difference vectors in ranking according to* F [28, 29]. By minimizing the norm of $\boldsymbol{w}$, OPT 1 maximizes the ranking of the closest data pairs, that corresponds to improving the generalization of the function.

To illustrate, see Figure 1 showing two different functions $F_{\boldsymbol{w}_1}$ and $F_{\boldsymbol{w}_2}$ that project four data vectors $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4\}$ onto $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ respectively in a two-dimensional space. Both $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ make the same ordering R for the four vectors, that is, $\boldsymbol{x}_1 \succ \boldsymbol{x}_2 \succ \boldsymbol{x}_3 \succ \boldsymbol{x}_4$. The distance between the closest two projections onto $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are respectively δ_1 and δ_2, which are formulated as $\frac{1}{||\boldsymbol{w}_1||}$ and $\frac{1}{||\boldsymbol{w}_2||}$.

Although the two weight vectors $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ make the same ordering, intuitivelly $\boldsymbol{w}_1$ is likely to generalize better than $\boldsymbol{w}_2$ because the distance of the closest vectors in $\boldsymbol{w}_1$ $(= \delta_1)$ is larger than that in $\boldsymbol{w}_2$ $(= \delta_2)$. OPT 1, by minimizing the norm of $\boldsymbol{w}$, computes the function that maximize the distance between the closest two projections [28].

OPT 2 (ranking SVM, *dual*).

$$\textit{maximize:}\quad L_d(\alpha) = \sum_{ij}^{P} \alpha_{ij} - \sum_{ij}^{P}\sum_{uv}^{P} \alpha_{ij}\alpha_{uv} K(\boldsymbol{x}_i - \boldsymbol{x}_j, \boldsymbol{x}_u - \boldsymbol{x}_v) \tag{7}$$

$$\textit{s.t.:}\quad C \geq \alpha \geq 0 \tag{8}$$

Similarily to the classifying SVM, this primal form can be tranformed to the dual form (OPT 2), on which the kernel trick can be applied in order to support nonlinear ranking functions [13]. α_{ij} in OPT 2 is a coefficient for a pairwise difference vector $(\boldsymbol{x}_i - \boldsymbol{x}_j)$. $K(\cdot)$ is a kernel function. Note that the kernel function is computed P^2 ($\sim m^4$) times, thus solving OPT 2 takes $O(m^4)$ at least. Once α is computed, $\boldsymbol{w}$ can be written in terms of the pairwise difference vectors and their coefficients such that:

$$\boldsymbol{w} = \sum_{ij}^{P} \alpha_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j) \tag{9}$$

The ranking function F on a new vector $\boldsymbol{z}$ can be computed using the kernel function replacing the dot product as follows:

$$F(\boldsymbol{z}) = \boldsymbol{w} \cdot \boldsymbol{z} \tag{10}$$

$$= \sum_{ij}^{P} \alpha_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j) \cdot \boldsymbol{z} \tag{11}$$

$$= \sum_{ij}^{P} \alpha_{ij} K(\boldsymbol{x}_i - \boldsymbol{x}_j, \boldsymbol{z}). \tag{12}$$

4 1-Norm Ranking SVM

The goal of 1-norm ranking SVM is the same as that of the standard ranking SVM, that is, to learn F that satisfies Eq.(1) for most $\{(\boldsymbol{x}_i, \boldsymbol{x}_j) : y_i < y_j \in R\}$ and generalize well beyond the training set. In the 1-norm ranking SVM, we express Eq.(1) using the F of Eq.(11) as follows.

$$F(\boldsymbol{x}_u) > F(\boldsymbol{x}_v) \Longrightarrow \sum_{ij}^{P} \alpha_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j) \cdot \boldsymbol{x}_u > \sum_{ij}^{P} \alpha_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j) \cdot \boldsymbol{x}_v \tag{13}$$

$$\Longrightarrow \sum_{ij}^{P} \alpha_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j) \cdot (\boldsymbol{x}_u - \boldsymbol{x}_v) > 0 \tag{14}$$

Thus, the constraint of the 1-norm ranking SVM becomes Eq.(16).

The standard ranking SVM suppresses the weight $\boldsymbol{w}$ to improve the generalization performance. (The objective function (4) minimizes the norm of $\boldsymbol{w}$.) The weight $\boldsymbol{w}$ can be also suppressed by suppressing the coefficient α since the weight is expressed by the sum of the coefficient times pairwise ranking difference vectors (Eq.(9)). (Mangasarian proves in [18] that suppressing the coefficient α corresponds to suppressing the weight $\boldsymbol{w}$ in the standard SVM.)

Thus, we have OPT 3 for the 1-norm ranking which suppresses α instead of $\boldsymbol{w}$ in the objective function (Eq.(15)). C is a user parameter controlling the tradeoff between the margin size and the amount of error, ξ, and K is the kernel function, e.g., $K(a, b) = a \cdot b$ for linear kernel and $K(a, b) = exp(-\gamma||a - b||^2)$ for RBF kernel.

OPT 3 (1-norm ranking SVM).

$$minimize: \qquad L(\alpha, \xi) = \sum_{ij}^{P} \alpha_{ij} + C \sum_{ij}^{P} \xi_{ij} \tag{15}$$

$$s.t.: \sum_{ij}^{P} \alpha_{ij} K(\boldsymbol{x}_i - \boldsymbol{x}_j, \boldsymbol{x}_u - \boldsymbol{x}_v) \geq 1 - \xi_{uv}, \forall \{(u,v) : y_u < y_v \in R\} \tag{16}$$

$$\alpha \geq 0,\ \xi \geq 0 \tag{17}$$

Training in the 1-norm ranking SVM becomes a linear programming (LP) problem thus solvable by LP algorithms such as the Simplex and Interior Point method [18, 11, 19]. Just as the standard ranking SVM (OPT 2), K needs to be computed P^2 ($\sim m^4$) times, and there are P number of constraints (16) and α to compute. Once α is computed, F is computed using the same ranking function as the standard ranking SVM, i.e., Eq.(12).

The accuracies of 1-norm ranking SVM and standard ranking SVM are comparable, and both methods need to compute the kernel function $O(m^4)$ times. In practice, the training of the standard SVM is more efficient because fast decomposition algorithms have been developed such as sequential minimal optimization (SMO) [21] while the 1-norm ranking SVM uses common LP solvers.

It is shown that *1-norm SVMs use much less support vectors that standard 2-norm SVMs*, that is, the number of positive coefficients (i.e., $\alpha > 0$) after training is much less in 1-norm SVMs [19, 11]. It is because, unlike the standard 2-norm SVM, the support vectors in the 1-norm SVM are not bounded to those close to the boundary in classification or the minimal ranking difference vectors in ranking. Thus, the testing involves much less kernel evaluations, and it is more robust when the training set contains noisy features [32]. Our experiments show that the 1-norm *ranking* SVM uses much less support vectors in the function than the standard ranking SVM.

5 Ranking Vector SVM (RV-SVM)

Although the 1-norm ranking SVM has merits over the standard ranking SVM in terms of the testing efficiency and feature selection, its training complexity is very high w.r.t. the number of data points. In this section, we present Ranking Vector SVM (RV-SVM), which revises our 1-norm ranking SVM to reduce the training time substantially. The RV-SVM significantly reduces the number of variables in the optimization problem while not compromizing the accuracy. *The key idea of RV-SVM is to express the ranking function with "ranking vectors" instead of support vectors.* The support vectors in ranking SVMs are chosen from pairwise difference vectors, and the number of pairwise difference vectors are quadratic to the size of training set. On the other hand, the ranking vectors are chosen from the training vectors, thus the number of variables to optimize is substantially reduced.

To theoretically justify our approach, we first present the Representer Theorem.

Theorem 1 (Representer Theorem [24])
Denote by Ω: $[0, \infty) \rightarrow \mathcal{R}$ a strictly monotonic increasing function, by $\mathcal{X}$ a set, and by $c : (\mathcal{X} \times \mathcal{R}^2)^m \rightarrow \mathcal{R} \cup \{\infty\}$ an arbitrary loss function. Then each minimizer $F \in \mathcal{H}$ of the regularized risk

$$c((x_1, y_1, F(x_1)), ..., (x_m, y_m, F(x_m))) + \Omega(||F||_{\mathcal{H}}) \tag{18}$$

admits a representation of the form

$$F(x) = \sum_{i=1}^{m} \alpha_i K(x_i, x) \tag{19}$$

We omit the proof due to the space limit, but the proof of the theorem is presented in [24].

Note that, in the theorem, the loss function c is *arbitrary* allowing *coupling* between data points $(\boldsymbol{x}_i, y_i)$, and the regularizer Ω has to be monotonic.

Given such a loss function and regularizer, the representer theorem states that although we might be trying to solve the optimization problem in an infinite-dimensional space $\mathcal{H}$, containing linear combinations of kernels centered on *arbitrary* points of $\mathcal{X}$, the solution lies in the span of m particular kernels – those centered on the *training* points [24].

Based on the theorem, we define our ranking function F as Eq.(19), which is based on the *training points* rather than arbitrary points (or pairwise difference vectors). Function (19) is similar to function (12) except that, unlike the latter using pairwise difference vectors $(\boldsymbol{x}_i - \boldsymbol{x}_j)$ and their coeifficients (α_{ij}), the former utilizes the training vectors $(\boldsymbol{x}_i)$ and their coefficients (α_i). With this function, Eq.(13) becomes the following.

$$F(\boldsymbol{x}_u) > F(\boldsymbol{x}_v) \Longrightarrow \sum_{i}^{m} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}_u) > \sum_{i}^{m} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}_v) \tag{20}$$

$$\Longrightarrow \sum_{i}^{m} \alpha_i (K(\boldsymbol{x}_i, \boldsymbol{x}_u) - K(\boldsymbol{x}_i, \boldsymbol{x}_v)) > 0. \tag{21}$$

Thus, we set our loss function c as follows.

$$c = \sum_{\forall\{(u,v): y_u < y_v \in R\}} (1 - \sum_{i}^{m} \alpha_i (K(\boldsymbol{x}_i, \boldsymbol{x}_u) - K(\boldsymbol{x}_i, \boldsymbol{x}_v))) \tag{22}$$

The loss function utilizes couples of data points penalizing misranked pairs, that is, it returns higher values as the number of misranked pairs increases. Thus, the loss function is order sensitive, and it is an instance of the function class c in Eq.(18).

We set the regularizer $\Omega(||f||_{\mathcal{H}}) = \sum_{i}^{m} \alpha_i$ $(\alpha_i \geq 0)$, which is strictly monotonically increasing. Let P is the number of pairs $(u, v) \in R$ such that $y_u < y_v$, and let $\xi_{uv} = 1 - \sum_{i}^{m} \alpha_i (K(\boldsymbol{x}_i, \boldsymbol{x}_u) - K(\boldsymbol{x}_i, \boldsymbol{x}_v))$.

Then, our RV-SVM is formulated as the LP of OPT 4, and the solution of the optimization problem lies in the span of kernels centered on the training points (i.e., Eq.(19)) as suggested in the representer theorem.

Just as the 1-norm ranking SVM, the RV-SVM suppresses α to improve the generalization, and forces Eq.(21) by constraint (24). Note that there are only m number of α_i in the RV-SVM. Thus, the kernel function is evaluated $O(m^3)$ times while the standard ranking SVM computes it $O(m^4)$ times.

OPT 4 (Ranking Vector SVM (RV-SVM)).

$$\textit{minimize:} \quad L(\alpha, \xi) = \sum_{i}^{m} \alpha_i + C \sum_{ij}^{P} \xi_{ij} \tag{23}$$

$$\textit{s.t.:} \sum_{i}^{m} \alpha_i (K(\boldsymbol{x}_i, \boldsymbol{x}_u) - K(\boldsymbol{x}_i, \boldsymbol{x}_v)) \geq 1 - \xi_{uv}, \forall \{(u, v) : y_u < y_v \in R\} \tag{24}$$

$$\alpha, \xi \geq 0 \tag{25}$$

---

6 Experiment

We evaluate the RV-SVM on synthetic datasets (Section 6.1) and a real-world dataset (Secton 6.2). We compare the RV-SVM with the state-of-the-art ranking SVM provided in SVM-light. Our experiments show that the RV-SVM trains substantially faster than the SVM-light for nonlinear kernels while their accuracies are comparable. More importantly, the number of ranking vectors in the RV-SVM is multiple orders of magnitudes smaller than the number of support vectors in the SVM-light. We performed our experiments on a Windows XP Professional machine with a Pentium IV 2.8GHz and 1GB of RAM. We implemented the RV-SVM using C and used CPLEX [1] for the LP solver. Our source codes can be found at "http://iis.postech.ac.kr/rv-svm" [30].

Evaluation metric: MAP (mean average precision) is used to measure the ranking quality when there are only two classes of ranking [27], and NDCG is used to evaluate ranking performance for IR applications when there are multiple levels of ranking [3,4, 7,26]. Kendall's τ is used when there is a global ordering of data and the training data is a subset of it. Ranking SVMs as well as the RV-SVM minimize the amount of error or mis-ranking, which is corresponding to optimizing the Kendall's τ [15,28]. Thus, we use the Kendall's τ to compare their accuracy.

6.1 Experiments on Synthetic Dataset

Below is the description of our experiments on synthetic datasets.

1. We randomly generated a training and a testing dataset D_{train} and D_{test} respectively, where D_{train} contains m_{train} (= 40, 80, 120, 160, 200) data points of n

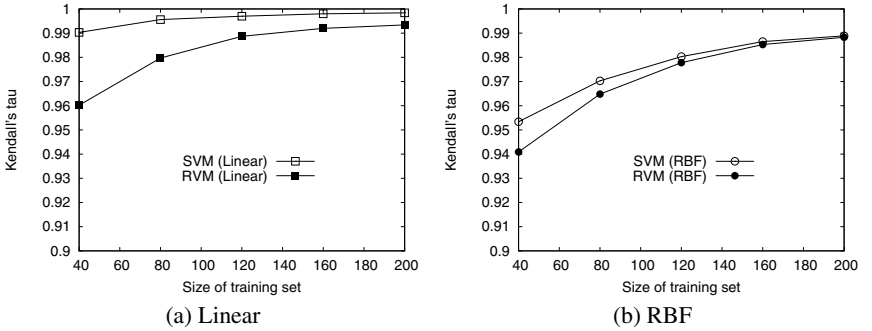
[1] http://www.ilog.com/products/cplex/

(e.g., 5) dimensions (i.e., m_{train}-by-n matrix), and D_{test} contains m_{test} (= 50) data points of n dimensions (i.e., m_{test}-by-n matrix). Each element in the matrices is a random number between zero and one. (We only did experiments on the data set of up to 200 objects due to performance reason. Ranking SVMs run intolerably slow on data sets larger than 200.)

2. We randomly generate a global ranking function F^*, by randomly generating the weight vector $\boldsymbol{w}$ in $F^*(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$ for linear, and in $F^*(\boldsymbol{x}) = exp(-||\boldsymbol{w} - \boldsymbol{x}||)^2$ for RBF function.
3. We rank D_{train} and D_{test} according to F^*, which forms the global ordering R^*_{train} and R^*_{test} on the training and testing data.
4. We train a function F from R^*_{train}, and test the accuracy of F on R^*_{test}.

We tuned the soft margin parameter C by trying $C = 10^{-5}, 10^{-5}, ..., 10^5$, and used the highest accuracy for comparison. For the linear and RBF functions, we used linear and RBF kernels accordingly. We repeat this entire process 30 times to get the mean accuracy.

Accuracy: Figure 2 compares the accuracies of the RV-SVM and the ranking SVM from the SVM-light. The ranking SVM outperforms RV-SVM when the size of data set is small, but their difference becomes trivial as the size of data set increases. This phenomenon can be explained by the fact that when the training size is too small, the number of potential ranking vectors becomes too small to draw an accurate ranking function whereas the number of potential support vectors is still large. However, as the size of training set increases, RV-SVM becomes as accurate as the ranking SVM because the number of potential ranking vectors becomes large as well.

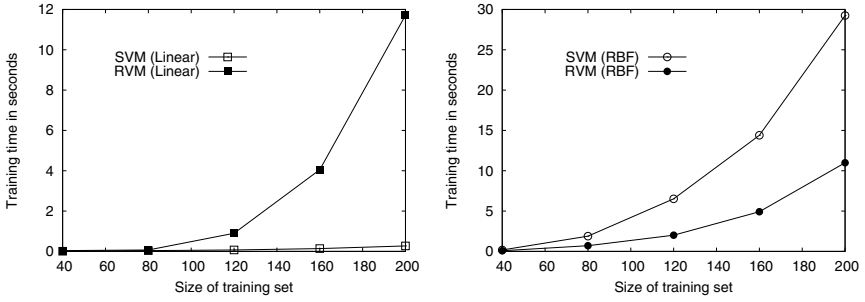


(a) Linear (b) RBF

Fig. 2. Accuracy

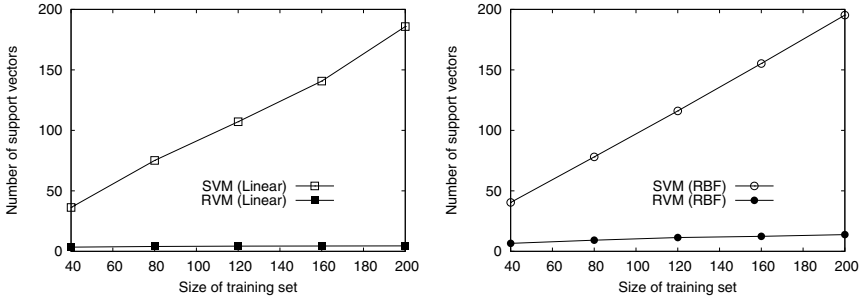
Training Time: Figure 3 compares the training time of the RV-SVM and the SVM-light. While the SVM light trains much faster than RV-SVM for linear kernel (SVM light is specially optimized for linear kernel.), *the RV-SVM trains significantly faster than the SVM light for RBF kernel.*

Number of Support (or Ranking) Vectors: Figure 4 compares the number of support (or ranking) vectors used in the function of RV-SVM and the SVM-light. RV-SVM's model uses a significantly smaller number of support vectors than the SVM-light.



(a) Linear Kernel (b) RBF Kernel

Fig. 3. Training time



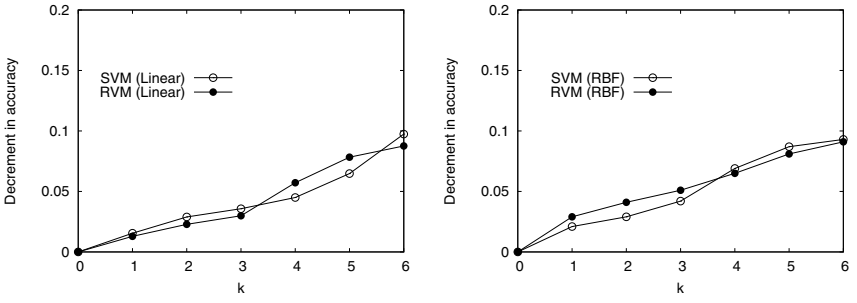
(a) Linear Kernel (b) RBF Kernel

Fig. 4. Number of support (or ranking) vectors

Sensitivity to noise: In this experiment, we compare the sensitivity of each method to noise. We insert noise by switching the orders of some data pairs in R^*_{train}. We set the size of training set $m_{train} = 100$ and the dimension $n = 5$. After we make R^*_{train} from a random function F^*, we randomly picked k vectors from the R^*_{train} and switched it with its adjacent vector in the ordering to implant noise in the training set. Figure 5 shows the decrements of the accuracies as the number of misorderings increases in the training set. Their accuracies are moderately decreasing as the noise increases in the training set, and their sensitivities to noise are comparable.

6.2 Experiment on Real Dataset

In this section, we experiment using the OHSUMED dataset obtained from the LETOR, the site containing benchmark datasets for ranking [1]. OHSUMED is a collection of documents and queries on medicine, consisting of 348,566 references and 106 queries. There are in total 16,140 query-document pairs upon which relevance judgements are made. In this dataset the relevance judgements have three levels: “definitely relevant”, “partially relevant”, and “irrelevant”. The OHSUMED dataset in the LETOR extracts 25 features. We report our experiments on the first three queries and their documents. We



(a) Linear (b) RBF

Fig. 5. Sensitivity to noise ($m_{train} = 100$)

compare the performance of RV-SVM and SVM-light on them. We tuned the parameters 3-fold cross validation with trying C and $\gamma = 10^{-6}, 10^{-5}, ..., 10^{6}$ for the linear and RBF kernels and compared the highest performance. The training time is measured for training the model with the tuned parameters. We repeated the whole process three times and reported the mean values.

Table 1. Experiment results: Accuracy (Acc), Training Time (Time), and Number of Support or Ranking Vectors (#SV or #RV)

		query 1 $\lvert D\rvert = 134$			query 2 $\lvert D\rvert = 128$			query 3 $\lvert D\rvert = 182$		
		Acc	Time	#SV or #RV	Acc	Time	#SV or #RV	Acc	Time	#SV or #RV
RV-SVM	linear	**.5484**	**.23**	**1.4**	**.6730**	**.41**	**3.83**	.6611	1.94	1.99
	RBF	.5055	.85	4.3	.6637	.41	2.83	**.6723**	**4.71**	**1**
SVM	linear	**.5634**	**1.83**	**92**	.6723	1.03	101.66	.6588	4.24	156.55
	RBF	.5490	3.05	92	**.6762**	**3.50**	**102**	**.6710**	**55.08**	**156.66**

Table 1 shows the results. The accuracies of the SVM and RV-SVM are comparable overall; SVM shows a little high accuracy than RV-SVM for query 1, but for the other queries, their accuracy differences are not statistically significant. More importantly, *the number of ranking vectors in RV-SVM is significantly smaller than that of support vectors in SVM.* For example, for query 3, the RV-SVM having just one ranking vector outperformed the SVM with over 150 support vectors. *The training time of RV-SVM is significantly shorter than that of SVM-light.*

7 Conclusion

This paper first develops the 1-norm ranking SVM which uses a much less support vectors than the standard 2-norm ranking SVM. We then develop the Ranking Vector SVM (RV-SVM) which uses as less support vectors as the 1-norm SVM and trains

much faster than the standard 2-norm SVM. Our experiments show that, compared to the SVM-light, our RV-SVM shows comparable accuracy on relatively large datasets, trains much faster, and uses a substantially less support vectors.

References

1. Letor: Learning to rank for information retrieval, http://research.microsoft.com/users/LETOR/
2. SVM-light, http://svmlight.joachims.org/
3. Baeza-Yates, R., Ribeiro-Neto, B. (eds.): Modern Information Retrieval. ACM Press, New York (1999)
4. Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hullender, G.: Learning to rank using gradient descent. In: Proc. Int. Conf. Machine Learning (ICML 2004) (2004)
5. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2, 121–167 (1998)
6. Cao, B., Shen, D., Sun, J.-T., Yang, Q., Chen, Z.: Feature selection in a kernel space. In: Proc. Int. Conf. Machine Learning (ICML 2007) (2007)
7. Cao, Y., Xu, J., Liu, T.-Y., Li, H., Huang, Y., Hon, H.-W.: Adapting ranking svm to document retrieval. In: Proc. ACM SIGIR Int. Conf. Information Retrieval (SIGIR 2006) (2006)
8. Cho, B., Yu, H., Lee, J., Chee, Y., Kim, I.: Nonlinear support vector machine visualization for risk factor analysis using nomograms and localized radial basis function kernels. IEEE Transactions on Information Technology in Biomedicine (accepted)
9. Christianini, N., Shawe-Taylor, J.: An Introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge (2000)
10. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research (2003)
11. Fung, G., Mangasarian, O.L.: A feature selection newton method for support vector machine classification. Computational Optimization and Applications (2004)
12. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research (2003)
13. Herbrich, R., Graepel, T., Obermayer, K. (eds.): Large margin rank boundaries for ordinal regression. MIT Press, Cambridge (2000)
14. Jiang, X., Yu, H.: SVM-JAVA: A Java implementation of the SMO (sequential minimal optimization) for training SVM (2008), http://iis.hwanjoyu.org/svm-java
15. Joachims, T.: Optimizing search engines using clickthrough data. In: Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD 2002) (2002)
16. Joachims, T.: Training linear svms in linear time. In: Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD 2006) (2006)
17. Lee, Y.-J., Mangasarian, O.L.: RSVM: Reduced support vector machines. In: SIAM Int. Conf. Data Mining (2001)
18. Mangasarian, O.L.: Generalized Support Vector Machines. MIT Press, Cambridge (2000)
19. Mangasarian, O.L.: Exact 1-norm support vector machines via unconstrained convex differentiable minimization. Journal of Machine Learning Research (2006)
20. Mangasarian, O.L., Wild, E.W.: Feature selection for nonlinear kernel support vector machines. Technical report, University of Wisconsin, Madison (1998)
21. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In: Scholkopf, A.S.B., Burges, C. (eds.) Advances in Kernel Methods: Support Vector Machines. MIT Press, Cambridge (1998)

22. Qin, T., Liu, T.-Y., Lai, W., Zhang, X.-D., Wang, D.-S., Li, H.: Ranking with multiple hyperplanes. In: Proc. ACM SIGIR Int. Conf. Information Retrieval (SIGIR 2007) (2007)
23. Radlinski, F., Joachims, T.: Query chains: Learning to rank from implicit feedback. In: Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD 2005) (2005)
24. Scholkopf, B., Herbrich, R., Smola, A.J., Williamson, R.C.: A generalized representer theorem. In: Helmbold, D.P., Williamson, B. (eds.) COLT 2001 and EuroCOLT 2001. LNCS, vol. 2111, p. 416. Springer, Heidelberg (2001)
25. Vapnik, V.: Statistical Learning Theory. John Wiley and Sons, Chichester (1998)
26. Xu, J., Li, H.: Adarank: A boosting algorithm for information retrieval. In: Proc. ACM SIGIR Int. Conf. Information Retrieval (SIGIR 2007) (2007)
27. Yan, L., Dodier, R., Mozer, M.C., Wolniewicz, R.: Optimizing classifier performance via the wilcoxon-mann-whitney statistics. In: Proc. Int. Conf. Machine Learning (ICML 2003) (2003)
28. Yu, H.: SVM selective sampling for ranking with application to data retrieval. In: Proc. Int. Conf. Knowledge Discovery and Data Mining (KDD 2005) (2005)
29. Yu, H., Hwang, S.-W., Chang, K.C.-C.: Enabling soft queries for data retrieval. Information Systems (2007)
30. Yu, H., Kim, Y., Hwang, S.-W.: RV-SVM: An efficient method for learning ranking SVM. Technical report, Department of Computer Science and Engineering, Pohang University of Science and Technology (POSTECH), Pohang, Korea (2008), `http://iis.hwanjoyu.org/rv-svm`
31. Yu, H., Yang, J., Wang, W., Han, J.: Discovering compact and highly discriminative features or feature combinations of drug activities using support vector machines. In: IEEE Computer Society Bioinformatics Conf. (CSB 2003), pp. 220–228 (2003)
32. Zhu, J., Rosset, S., Hastie, T., Tibshriani, R.: 1-norm support vector machines. In: Proc. Advances in Neural Information Processing Systems (NIPS 2000) (2003)

A Kernel Framework for Protein Residue Annotation

Huzefa Rangwala[1], Christopher Kauffman[2], and George Karypis[2]

[1] George Mason University, Fairfax, VA 22030, USA
rangwala@cs.gmu.edu
[2] University of Minnesota, Minneapolis, MN 55414, USA

Abstract. Over the last decade several prediction methods have been developed for determining structural and functional properties of individual protein residues using sequence and sequence-derived information. Most of these methods are based on support vector machines as they provide accurate and generalizable prediction models. We developed a general purpose protein residue annotation toolkit (*Pro*SAT) to allow biologists to formulate residue-wise prediction problems. *Pro*SAT formulates annotation problem as a classification or regression problem using support vector machines. For every residue *Pro*SAT captures local information (any sequence-derived information) around the reside to create fixed length feature vectors. *Pro*SAT implements accurate and fast kernel functions, and also introduces a flexible window-based encoding scheme that allows better capture of signals for certain prediction problems. In this work we evaluate the performance of *Pro*SAT on the disorder prediction and contact order estimation problems, studying the effect of the different kernels introduced here. *Pro*SAT shows better or at least comparable performance to state-of-the-art prediction systems. In particular *Pro*SAT has proven to be the best performing transmembrane-helix predictor on an independent blind benchmark. **Availability:** http://bio.dtc.umn.edu/prosat

1 Introduction

Experimental methods to determine the structure and function of proteins have been out-paced with the abundance of available sequence data. As such, over the past decade several computational methods have been developed to characterize the structural and functional aspects of proteins from sequence information [26].

Support vector machines (SVMs) [28] along with other machine learning tools have been extensively used to successfully predict the residue-wise structural or functional properties of proteins [20, 4, 15]. The task of assigning every residue with a discrete class label or continuous value is defined as a *residue annotation* problem. Examples of structural annotation problems include the secondary structure prediction [22,15,11], local structure prediction [14,5], and contact order prediction [27,18]. Examples of function property annotation include prediction of interacting residues [20] (e.g., DNA-binding residues, and ligand-binding residues), solvent accessible surface area estimation [21,25], and disorder prediction [9, 4].

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 439–451, 2009.
© Springer-Verlag Berlin Heidelberg 2009

We have developed a general purpose protein residue annotation toolkit called *Pro*SAT. This toolkit uses a support vector machine framework and is capable of predicting both a discrete label or a continuous value. *Pro*SAT allows use of any type of sequence information with residues for annotation. For every residue, *Pro*SAT encodes the input information from the residue and its neighbors. We introduce a new flexible encoding scheme that differentially weighs information extracted from neighboring residues, based on the distance to the central residue. *Pro*SAT also uses an exponential second-order kernel function shown to be effective in capturing pairwise interactions between residues, and hence improve the classification and regression performance for the annotation problems [15].

To the best of our knowledge, *Pro*SAT is the first tool that is designed to allow life science researchers to quickly and efficiently train SVM-based models for annotating protein residues with any desired property. The kernel functions implemented are also optimized for speed, by utilizing fast vector-based operation routines within the CBLAS library [29]. *Pro*SAT[1] is made available as a pre-compiled binary on several different architectures and environments.

In this paper we report our evaluation studies highlighting the different features of *Pro*SAT on the disorder prediction [4] and contact order estimation [27] problem. *Pro*SAT shows a statistically significant improvement on both the disorder prediction (1%) and contact order estimation problems (20%) in comparison to previously established methods. We have also tested *Pro*SAT on the DNA-binding [20], and local structure prediction problem (results not reported here). *Pro*SAT improves over state-of-the-art transmembrane helix prediction methods [12], as evaluated by an independent benchmark [17]. Recently, *Pro*SAT was used to develop the best performing transmembrane-helix segment identification and orientation system called TOPTMH [1], and improve the comparative modeling ligand-binding regions of proteins [16]. The models trained by *Pro*SAT are also used to generate predictions for a webserver developed by us called MONSTER (Minnesota prOteiN Sequence annoTation servER) available at `http://bio.dtc.umn.edu/monster`.

2 Problem Definition and Notations

In this paper, we will refer to protein sequences by X and Y, and an arbitrary residue by x. Given a sequence X of length n, with it are associated derived features F, a $n \times d$ matrix where d is the dimensionality of the feature space. The features associated with the ith residue x_i are the ith row of the matrix F denoted as F_i. When multiple types of features are considered, the lth feature matrix is specified by F^l. In Figure 1 (a) we show the PSI-BLAST derived position specific scoring matrix of dimensions $n \times 20$ (discussed in Section 3.2).

In order to encode information for a residue *Pro*SAT uses the information from neighboring residues as well. *Pro*SAT uses a w`mer`-based encoding to capture sequence information for residue x_i to perform the residue-wise prediction. *Pro*SAT uses the $(2w+1)$ rows of the matrix F, $F_{i-w} \ldots F_{i+w}$ to encapsulate the feature information associated with the w`mer` centered at residue x_i. This submatrix is

[1] http://bio.dtc.umn.edu/prosat

denoted by $w\texttt{mer}(F_i)$ and is linearized to generate a vector of length $(2w+1)d$, where d is the dimension of the matrix F.

As seen in Figure 1(b) for the circled residue, three residues above and below are also selected and the corresponding information from the feature matrix is extracted. Further Figure 1(c) represents the linearized submatrix as a vector which encodes the information for the problem.

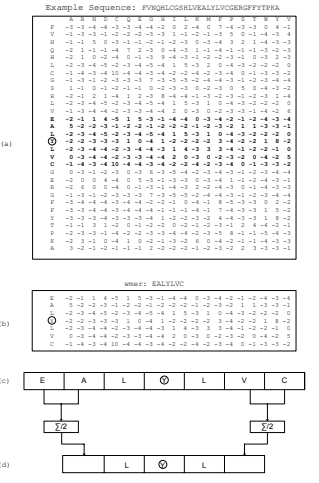


Fig. 1. (a) Input example sequence along with PSI-BLAST profile matrix of dimensions $n \times 20$, with a residue circled to show the encoding steps. (b) Example $w\texttt{mer}$ of $w = 3$ and length seven, with extracted submatrix from the original PSI-BLAST matrix. (c) Encoded vector of length 7×20 formed by linearizing the submatrix (d) Flexible encoding showing three residues in the center using the finer representation, and two residues flanking the central residues on both sides using a coarser representation as an averaging statistic. Length of this vector equals 5×20.

3 Methods

We approach the protein residue annotation problem by utilizing local sequence information around each residue in a supervised machine learning framework. We use support vector machines (SVM) [28] in both classification and regression formulations to address the problem of annotating residues with discrete labels and continuous values respectively. We use the publicly available SVM^{light} program [10] for the discriminatory learning.

3.1 Support Vector Classification and Regression

The task of assigning a label to the residue x from one of the K possible annotation labels is a typical multiclass classification problem. The general strategy is to build K one-versus-rest binary SVM classification models that assign a residue to be in a particular class or not. Let $\mathcal{A}^+$ refer to the residues with on particular label, the positive class, and $\mathcal{A}^-$ refer to the remaining residues, the negative class. In its dual formulation, a support vector machine learns a classification function $f(x)$ of the form

$$f(x) = \sum_{x_i \in \mathcal{A}^+} \lambda_i^+ \mathcal{K}(x, x_i) - \sum_{x_i \in \mathcal{A}^-} \lambda_i^- \mathcal{K}(x, x_i), \tag{1}$$

where λ_i^+ and λ_i^- are non-negative weights that are computed during training to provide the best possible prediction, and $\mathcal{K}(.,.)$ is a *kernel* function designed

to capture the similarity between pairs of residues. Having learned the function $f(x)$, a new residue x is predicted to be positive or negative depending on whether $f(x)$ is positive or negative. The value of $f(x)$ also signifies the tendency of x to be a member of the positive or negative class and can be used to obtain a meaningful ranking of a set of the residues.

We use the error insensitive support vector regression ϵ-SVR [28] for learning a function $f(x)$ for estimation in case of determining a quantity, as in the case of solvent accessibility prediction problem. Given a set of training instances (x_i, y_i), where y_i is the continuous value to be estimated for residue x_i, the ϵ-SVR aims to learns a function of the form

$$f(x) = \sum_{x_i \in \Delta^+} \alpha_i^+ \mathcal{K}(x, x_i) - \sum_{x_i \in \Delta^-} \alpha_i^- \mathcal{K}(x, x_i), \tag{2}$$

where Δ^+ contains the residues for which $y_i - f(x_i) > \epsilon$, Δ^- contains the residues for which $y_i - f(x_i) < -\epsilon$, and α_i^+ and α_i^- are non-negative weights that are computed during training by maximizing a quadratic objective function. The objective of the maximization is to determine the flattest $f(x)$ in the feature space and minimize the estimation errors for instances in $\Delta^+ \cup \Delta^-$. Hence, instances that have an estimation error satisfying $|f(x_i) - y_i| < \epsilon$ are neglected. The parameter ϵ controls the width of the regression deviation or tube.

3.2 Sequence-Based Information

Pro`SAT` can use any general user-supplied features. In our empirical evaluation for a given protein X of length n we encode the sequence information using PSI-BLAST position specific scoring matrices, predicted secondary structure, and position independent scoring matrices like BLOSUM62. These feature matrices are referred to as $\mathcal{P}$, $\mathcal{S}$, and $\mathcal{B}$, respectively and are described below.

Position Specific Scoring Matrices. The profile of a protein is derived by computing a multiple sequence alignment of it with a set of sequences that have a statistically significant sequence similarity, i.e., they are sequence homologs as ascertained by PSI-BLAST [2]. In Figure 1 (a) we show the PSI-BLAST derived position specific scoring matrix for a sequence of length n. The dimensions of this matrix $n \times 20$. For every residue the PSI-BLAST matrix captures evolutionary conservation information by providing a score for each of the twenty amino acids.

The profiles in this study were generated using the latest version of the PSI-BLAST [2] (available in NCBI's blast release 2.2.10 using `blastpgp -j 5 -e 0.01 -h 0.01`) searched against NCBI's NR database that was downloaded in November of 2004 and contains 2,171,938 sequences.

Predicted Secondary Structure Information. We use YASSPP [15] to predict secondary structure and generate a position-specific secondary structure matrices. For a length n sequence, the result is $\mathcal{S}$, a $n \times 3$ feature matrix. The (i, j)th entry of this matrix represents the propensity for residue i to be in state j, where $j \in \{1, 2, 3\}$ corresponds to the three secondary structure elements: alpha helices, beta sheets, and coil regions.

Position Independent Scoring Matrices. A less computationally expensive feature of protein sequences may be obtained from a position independent scoring matrix such as the BLOSUM62 substitution matrix. The primary motivation for using BLOSUM62-derived feature vectors is to improve the classification accuracy in cases where a sequence does not have a sufficiently large number of homologous sequences in NR. In these cases PSI-BLAST fails to compute a correct alignment for some segments of the sequence giving a misleading PSSM [9, 15]. To make effective use of *Pro*SAT's capabilities we create a $n \times 20$ feature matrix, referred to as $\mathcal{B}$, where each row of the matrix is a copy of the BLOSUM62 row corresponding to the amino acid at that position in the sequence.

By using both PSSM- and BLOSUM62-based information, the SVM learner can construct a model that is partially based on non-position specific information. Such a model will remain valid in cases where PSI-BLAST could not generate correct alignments due to lack of homology to sequences in the nr database [15].

3.3 Kernel Functions

A kernel function computes a similarity between two objects and selection of an appropriate kernel function for a problem is key to the effectiveness of support vector machine learning. We consider several individual kernels of interest and then proceed to describe combinations of kernels used in this study. Throughout this section we use F and G be the feature matrix for sequences X and Y respectively. A specific residue of X is denoted x_i and its associated vector of features is F_i.

Window Kernel. Our contribution in this work is a two-parameter linear window-kernel, denoted by $\mathcal{W}_{w,f}$ which computes the similarity between two w`mers`, $w\texttt{mer}(x_i)$ and $w\texttt{mer}(y_j)$ according to their features $w\texttt{mer}(F_i)$ and $w\texttt{mer}(G_j)$, respectively. The kernel function is defined as

$$\begin{aligned}\mathcal{W}_{w,f}(x_i, y_j) = & \sum_{k=-f}^{f} \langle F_{i+k}, G_{j+k} \rangle + \\ & \langle \sum_{k=f+1}^{w} F_{i+k}, \sum_{k=f+1}^{w} G_{j+k} \rangle + \\ & \langle \sum_{k=f+1}^{w} F_{i-k}, \sum_{k=f+1}^{w} G_{i-k} \rangle .\end{aligned} \tag{3}$$

The parameter w governs the size of the w`mer` considered in computing the kernel while f offers control over the fine-grained versus coarse-grained sections of the window. Rows within $\pm f$ contribute an individual dot product to the total similarity while rows outside this range are first summed and then their dot product is taken. In all cases $f \leq w$ and as f approaches w, the window kernel becomes simply a sum of the dot products, the most fine-grained similarity measure considered. This window encoding is shown in Figure 1(d) where the positions away from the central residue are averaged to provide a coarser representation, whereas the positions closer to the central residue provide a finer representation. The rationale behind this kernel design is that some problems may require only approximate information for sequence neighbors which are far away from the central residue while nearby sequence neighbors are more important. Specifying $f \ll w$ merges these distant neighbors into only a coarse

contribution to the overall similarity, as it only accounts for compositional information and not the specific positions where these features occur. The window kernel is defined as a dot-product, which makes it equivalent to linear kernel with a feature encoding scheme that takes into account the two variable parameters, w and f. Hence, we can embed the dot-product based $\mathcal{W}$ within other complex kernel functions.

Exponential Kernels. Another individual kernel we use extensively is the second order exponential kernel, $\mathcal{K}^{soe}$, developed in our earlier works for secondary structure and local structure information prediction [15, 23]. Given any base kernel function $\mathcal{K}$, we define $\mathcal{K}^2$ as

$$\mathcal{K}^2(x, y) = \mathcal{K}(x, y) + (\mathcal{K}(x, y))^2. \tag{4}$$

which is a second-order kernel in that it computes pairwise interactions between the elements x and y. We then define $\mathcal{K}^{soe}$ as

$$\mathcal{K}^{soe}(x, y) = \exp\left(1 + \frac{\mathcal{K}^2(x, y)}{\sqrt{\mathcal{K}^2(x, x)\,\mathcal{K}^2(y, y)}}\right) \tag{5}$$

which normalizes $\mathcal{K}^2$ and embeds it into an exponential space.

We also use the standard radial basis kernel function (*rbf*), defined for some parameter γ by $\mathcal{K}^{rbf}(x, y) = \exp(-\gamma||x-y||^2)$. By setting a specific γ parameter and using normalized unit length vectors the standard *rbf* kernel can be shown equivalent (upto a scaling factor) to a first order exponential kernel obtained by removing the $\mathcal{K}^2(x, y)$ term in Equation 4, and plugging the modified kernel in Equation 5.

In this paper, we denote the *soe* to be the kernel $\mathcal{K}^{soe}$ using the $\mathcal{W}_{w,f}$ as the base, *rbf* to be the kernel $\mathcal{K}^{rbf}$ using the normalized form with $\mathcal{W}_{w,f}$ as the base, and *lin* to be the base linear kernel $\mathcal{W}_{w,f}$.

3.4 Integrating Information

To integrate the different information, we use a linear combination of the kernels derived for different feature matrices. Consider two sequences with features F^l and G^l for $l = 1, \ldots, k$, our fusion kernel using the is defined

$$\mathcal{K}^{fusion}(x_i, y_j) = \sum_{l=1}^{k} \omega_l\, \mathcal{K}^{soe}(F_i^l, G_j^l) \tag{6}$$

where the weights ω_l are supplied by the user. Note the *soe* kernel in Equation 6 can be replaced by the *lin*, and *rbf* kernels.

In the future we intend to explore the possibility of automatically learning the weights ω_l. This can be done by using some of the recent multiple kernel integration work that combines heterogeneous information using semidefinite programming [19], second order cone programming [3], and semi-infinite linear programming [24].

4 Case Studies

*Pro*SAT was tested on a wide variety of local structure and function prediction problems. Here we present a case study on the disorder prediction, contact order estimation and transmembrane-helix prediction problems. We review the methods used for solving the problems, and provide comparative results by using standard benchmarks which are described below.

*Pro*SAT was also tested on the DNA-binding prediction problem [20], ligand-binding prediction problem, solvent accessibility surface area estimation [25, 21], and local structure alphabet prediction problem [5]. The results of these experiments are not reported here for sake of brevity. *Pro*SAT showed comparable to the state-of-the-art prediction systems for the different problems.

4.1 Experimental Protocol

The general protocol we used for evaluating the different parameters, and features, as well as comparing to previously established studies remained fairly consistent across the different problems. In particular we used a n-fold cross validation methodology, where $1/\mathrm{n}th$ of the database in consideration was used for testing and the remaining dataset was used for training, with the experiment being repeated n times.

4.2 Evaluation Metrics

We measure the quality of the methods using the standard receiver operating characteristic (ROC) scores. The ROC score is the normalized area under the curve that plots the true positives against the false positives for different thresholds for classification [8]. We also compute other standard statistics, and report the $F1$ score which takes into account both the precision and recall for the prediction problem.

The regression performance is assessed by computing the standard Pearson correlation coefficient (CC) between the predicted and observed true values for every protein in the datasets. We also compute the root mean square error $rmse$ between the predicted and observed values for every proteins. The results reported are averaged across the different proteins and cross validation steps. For the $rmse$ metric, a lower score implies a better quality prediction.

We also compute a statistical significance test, $errsig$ to differentiate between the different methods. $errsig$ is the significant difference margin for each score and is defined as the standard deviation divided by the square root of the number of proteins.

4.3 Disorder Prediction

Some proteins contain regions which are intrinsically disordered in that their backbone shape may vary greatly over time and external conditions. A disordered region of a protein may have multiple binding partners and hence can

Table 1. Classification Performance on the Disorder Dataset

	w	$f=1$		$f=3$		$f=5$		$f=7$		$f=9$		$f=11$	
		ROC	F1	ROC	F1	ROC	F1	ROC	F1	ROC	F1	ROC	F1
$\mathcal{P}^{lin}$	3	0.775	0.312	**0.800**	0.350	-	-	-	-	-	-	-	-
	7	0.815	0.366	**0.817**	0.380	0.816	0.384	0.816	0.383	-	-	-	-
	11	0.821	0.378	0.826	0.391	**0.828**	0.396	0.826	0.400	0.824	0.404	0.823	0.403
	13	0.823	0.384	0.829	0.398	0.832*	0.405	0.830	0.404	0.828	0.407	0.826	0.409
$\mathcal{P}^{rbf}$	3	**0.811**	0.370	0.811	0.369	-	-	-	-	-	-	-	-
	7	0.845	0.442	**0.849**	0.450	0.848	0.445	0.845	0.442	-	-	-	-
	11	0.848	0.464	0.855	0.478	0.858	0.482	**0.858**	0.480	0.855	0.470	0.853	0.468
	13	0.848	0.473	0.855	0.484	0.859	0.490	0.861*	0.492	0.860	0.487	0.857	0.478
$\mathcal{P}^{soe}$	3	0.815	0.377	**0.816**	0.379	-	-	-	-	-	-	-	-
	7	0.847	0.446	**0.852**	0.461	0.852	0.454	0.851	0.454	-	-	-	-
	11	0.848	0.469	0.856	0.482	0.860	0.491	**0.862**	0.491	0.861	0.485	0.862	0.485
	13	0.847	0.473	0.856	0.485	0.861	0.491	0.864	0.495	0.865*	0.494	0.864	0.492
$\mathcal{P}\,\mathcal{S}^{soe}$	3	0.836	0.418	**0.838**	0.423	-	-	-	-	-	-	-	-
	7	0.860	0.472	**0.862**	0.476	0.860	0.473	0.859	0.468	-	-	-	-
	11	0.861	0.490	0.867	0.496	**0.868**	0.498	0.868	0.495	0.866	0.488	0.865	0.485
	13	0.860	0.497	0.867	0.503	0.870	0.503	0.871*	0.503	0.870	0.498	0.868	0.492
$\mathcal{P}\,\mathcal{S}\,\mathcal{B}^{soe}$	3	**0.842**	0.428	0.841	0.428	-	-	-	-	-	-	-	-
	7	0.869	0.497	**0.870**	0.499	0.869	0.494	0.867	0.489	-	-	-	-
	11	0.871	0.516	0.875	0.518	**0.877**	0.517	0.877	0.512	0.874	0.508	0.873	0.507
	13	0.869	0.519	0.875	0.522	0.878	0.521	0.879**	0.519	0.879	0.518	0.876	0.514

DISPro [4] reports a *ROC* score of 0.878. The numbers in bold show the best models for a fixed w parameter, as measured by *ROC*. $\mathcal{P}$, $\mathcal{B}$, and $\mathcal{S}$ represent the PSI-BLAST profile, BLOSUM62, and YASSPP scoring matrices, respectively. *soe*, *rbf*, and *lin* represent the three different kernels studied using the $\mathcal{W}_{w,f}$ as the base kernel. * denotes the best classification results in the sub-tables, and ** denotes the best classification results achieved on this dataset. For the best model we report a Q_2 accuracy of 84.60% with an *errsig* rate of 0.33.

take part in multiple biochemical processes in the cell which make them critical in performing various functions [7]. Disorder region prediction methods like IUPred [6], Poodle [9], and DISPro [4] mainly use physiochemical properties of the amino acids or evolutionary information within a machine learning tool like bi-recurrent neural network or SVMs.

*Pro*SAT was evaluated on the disorder prediction problem by training binary classification model to discriminate between residues that belong to part of disordered region or not. For evaluating the disorder prediction problem we used the DisPro [4] dataset which consisted of 723 sequences (215612 residues), with the maximum sequence identity between sequence pairs being 30%.

Table 2. Runtime Performance of *Pro*SAT on the Disorder Dataset (in seconds)

	w=f=11				w=f=13				w=f=15			
	#KER	NO	YES	SP	#KER	NO	YES	SP	#KER	NO	YES	SP
$\mathcal{P}^{lin}$	1.93e+10	83993	45025	1.86	1.92e+10	95098	53377	1.78	1.91e+10	106565	54994	1.93
$\mathcal{P}^{rbf}$	1.91e+10	79623	36933	2.15	1.88e+10	90715	39237	2.31	1.87e+10	91809	39368	2.33
$\mathcal{P}^{soe}$	2.01e+10	99501	56894	1.75	2.05e+10	112863	65035	1.73	2.04e+10	125563	69919	1.75

The runtime performance of *Pro*SAT was benchmarked for learning a classification model on a 64-bit Intel Xeon CPU 2.33 GHz processor. #KER denotes the number of kernel evaluations for training the SVM model. NO denotes runtime in seconds when the cblas library was not used, YES denotes the runtime in seconds when the cblas library was used, and SP denotes the speedup achieved using the cblas library.

We used the PSI-BLAST profile matrix denoted by $\mathcal{P}$, a BLOSUM62 derived scoring matrix denoted by $\mathcal{B}$, and predicted secondary structure matrix denoted by $\mathcal{S}$ feature matrices both independently, and in combinations . We varied the w, and f parameters for the $\mathcal{W}$, and also compared the *lin*, *rbf*, and *soe* kernels. Table 1 shows the binary classification performance measured using the *ROC* and F_1 scores achieved on the disorder dataset after a ten fold cross validation experiment, previously used to evaluate the DISPro prediction method.

Comparing the *ROC* performance of the $\mathcal{P}^{soe}$, $\mathcal{P}^{rbf}$, and $\mathcal{P}^{lin}$ models across different values of w and f used for parameterization of the base kernel ($\mathcal{W}$), we observe that the *soe* kernel shows superior performance to the *lin* kernel and slightly better performance compared to the normalized *rbf* kernel used in this study. This verifies results of our previous studies for predicting secondary structure [15] and predicting RMSD between subsequence pairs [23], where the *soe* kernel outperformed the *rbf* kernel.

The performance *Pro*SAT on the disorder prediction problem was shown to improve when using the $\mathcal{P}$, $\mathcal{B}$, and $\mathcal{S}$ feature matrices in combination rather than individually. We show results for the $\mathcal{P}\,\mathcal{S}$ and $\mathcal{P}\,\mathcal{S}\,\mathcal{B}$ features in Table 1. The flexible encoding introduced by *Pro*SAT shows a slight merit for the disorder prediction problem. These improvements are statistically significant as evaluated by the *errsig* measure.

The best performing fusion kernel improves the accuracy by 1% in comparison to DisPro [4] that encapsulates profile, secondary structure and relative solvent accessibility information within a bi-recurrent neural network.

4.4 Contact Order Estimation

Pairs of residues are considered to be in contact if their C_β atoms are within a threshold radius, generally 12 Å. Residue-wise contact order [27] is an average of the distance separation between contacting residues within a sphere of set threshold. Previously, a support vector regression method [27] has used a combination of local sequence-derived information in the form of PSI-BLAST profiles [2] and predicted secondary structure information [11], and global information based on amino acid composition and molecular weight for good quality estimates of the residue-wise contact order value. Amongst other techniques, critical random networks [18] use PSI-BLAST profiles as a global descriptor for this estimation problem.

*Pro*SAT was used to train ϵ-SVR regression models for estimating the residue-wise contact order on a previously used dataset [27] using the fusion of $\mathcal{P}$ and $\mathcal{S}$ features, with a *soe* kernel. This dataset consisted of 680 sequences (120421 residues), and the maximum pairwise sequence identity for this dataset was 40%.

In Table 3 we present the regression performance for estimating the residue wise contact order by performing 15-fold cross validation. These results are evaluated by computing the correlation coefficient and rmse values averaged across the different proteins in the dataset.

Analyzing the effect of the w and f parameters for estimating the residue-wise contact order values, we observe that a model trained with $f < w$ generally shows better CC and *rmse* values. The best models as measured by the CC scores are

Table 3. Residue-wise Contact Order Estimation Performance

	w	$f=1$		$f=3$		$f=5$		$f=7$		$f=9$		$f=11$	
		CC	rmse	CC	rmse	CC	rmse	CC	rmse	CC	rmse	CC	rmse
$\mathcal{P}\,\mathcal{S}^{soe}$	3	0.704	0.696	**0.708**	0.692	-	-	-	-	-	-	-	-
	7	0.712	0.683	0.719	0.677	**0.723**	0.672	.722	0.672	-	-	-	-
	11	0.711	0.681	0.720	0.673	**0.725**	0.667	0.725	0.666	0.724	0.666	0.722	0.667
	15	0.709	0.680	0.719	0.672	0.726**	0.665	0.726	0.664	0.725	0.664	0.723	0.664

CC and $rmse$ denotes the average correlation coefficient and rmse values. The numbers in bold show the best models as measured by CC for a fixed w parameter. $\mathcal{P}$, and $\mathcal{S}$ represent the PSI-BLAST profile and YASSPP scoring matrices, respectively. soe, rbf, and lin represent the three different kernels studied using the $\mathcal{W}_{w,f}$ as the base kernel. * denotes the best regression results in the sub-tables, and ** denotes the best regression results achieved on this dataset. For the best results the $errsig$ rate for the CC values is 0.003. The published results [27] uses the default rbf kernel to give $CC = 0.600$ and $rmse = 0.78$.

highlighted in Table 3. A model with equivalent CC values but having a lower f value is considered better because of the reduced dimensionality achieved by such models.

The best estimation performance achieved by our ϵ-SVR based learner uses a fusion of the $\mathcal{P}$ and $\mathcal{S}$ feature matrices and improves CC by 21%, and $rmse$ value by 17% over the ϵ-SVR technique of Song and Barrage [27]. Their method uses the standard *rbf* kernel with similar local sequence-derived amino acid and predicted secondary structure features. The major improvement of our method can be attributed to our fusion-based kernel setting with efficient encoding, and the normalization introduced in Equation 5.

4.5 Transmembrane-Helix Prediction

Proteins which span the cell membrane have proven difficult to crystallize in most cases and are generally too large for NMR studies. Computational methods to elucidate transmembrane protein structure are a quick means to obtain approximate topology. Many of these proteins are composed of a inter-cellular, extra-cellular, transition, and membrane portions where the membrane portion contains primarily hydrophobic residues in helices (a multi-class classification problem). Accurately labeling these four types of residues allows helix segments allows them to be excluded from function studies as they are usually not involved in the activity of the protein. MEMSAT [12] in its most recent incarnation uses

Table 4. Performance of *Pro*SAT and TOPTMH on the trans-membrane helix prediction problem

Method	$\mathcal{P}^{soe}$	*TOPTMH*	MEMSAT3	TMHMM1	PHDpsihtm08	HMMTOP2	PHDhtm08
Q_2	84	84	83	80	80	80	78
REC	81	75	78	68	76	69	76
PRE	87	90	88	81	83	89	82

Q_2, REC, and PRE denote the per-residue accuracy, recall and precision respectively. Results for MEMSAT3 [12], TOPTMH [1] and $\mathcal{P}^{soe}$ were obtained by evaluating it on the TMH static benchmark [17] and submitting the results of prediction to the server. We use the $\mathcal{P}^{soe}$ kernel with $w = f = 7$. All the other results were obtained from the TMH static benchmark evaluation web-site. Note, TOPTMH [1] uses *Pro*SAT for performing per-residue annotation, and then uses a set of hidden markov models to improve the per-segment accuracy.

profile inputs to a neural network to predict whether residues in a transmembrane protein are part of a transmembrane helical region or not.

Kernytsky and Rost have benchmarked a number of methods and maintain a server to compare the performance of new methods which we employ in our evaluation [17]. We evaluate *Pro*SAT using this independent static benchmark. Firstly, we perform model selection on a set of 247 sequences used previously by the Phobius algorithm [13]. We use the $\mathcal{P}^{soe}$ kernel with w and f parameters set to 7 to train a four-way classification model for predicting the residue to be in either the helical region, non-helical region, inter-cellular region, and extracellular region. Using the trained model we annotate each of the 2247 sequences in the static benchmark (no true labels known to us)[2]. The performance of *Pro*SAT is shown in Table 4, which is better in comparison to state-of-the-art methods. The predictions from *Pro*SAT were further smoother using a second-level model to build the best performing transmembrane helix identification system called TOPTMH [1]. The reader is encouraged to find more details about experimental results in the TOPTMH [1] study.

4.6 Runtime Performance of Optimized Kernels

We also benchmark the learning phase of *Pro*SAT on the disordered dataset comparing the runtime performance of the program compiled with and without the CBLAS subroutines. These results are reported in Table 2 and were computed on a 64-bit Intel Xeon CPU 2.33 GHz processor for the $\mathcal{P}^{lin}$, $\mathcal{P}^{rbf}$, and $\mathcal{P}^{soe}$ kernels varying the $w$mer size from 11 to 15. Table 2 also shows the number of kernel evaluations for the different models. We see speedups ranging from 1.7 to 2.3 with use of the CBLAS library. Similar experiments were performed on other environments and other prediction problems, and similar trends were seen.

5 Conclusions and Future Directions

In this work we have developed a general purpose support vector machine based toolkit for easily developing predictive models to annotate protein residue with structural and functional properties. *Pro*SAT was tested with different sets of features on several annotation problems. Besides the problems illustrated here *Pro*SAT was used for developing a webserver called MONSTER[3] that predicts several local structure and functional properties using PSI-BLAST profiles only. *Pro*SAT also showed success in predicting and modeling ligand-binding site regions from sequence information only [16].

The empirical results presented here showed the capability of *Pro*SAT to accept information in the form of PSI-BLAST profiles, BLOSUM62 profiles, and predicted secondary structure. *Pro*SAT was tested with the *soe*, *rbf*, and *lin* kernel function. In addition, the results showed that for some problems (contact

[2] Static Benchmark for testing Transmembrane helix prediction at http://cubic.bioc.columbia.edu/services/tmh_benchmark

[3] http://bio.dtc.umn.edu/monster

order estimation), by incorporating local information at different levels of granularity with the flexible encoding, *Pro*SAT was able to achieve better performance when compared to the traditional fine-grain approach.

Presently we are studying different multiple kernel integration methods that would automatically weight the contribution of different information in Equation 6. An optimal set of weights can be learned using semi-definite programming [19], and semi-infinite linear programming [24]. Currently, *Pro*SAT automatically performs a grid search over the different parameters for selecting the best model. The multiple kernel integration work can also be used to select the best model. This would allow the biologist to use *Pro*SAT effectively. Further like the TOPTMH [1] system, we would like to smooth the predictions obtained from the residue-level predictors. This can be done by training a second level model or incorporating domain specific rules. A second level SVM-based model [15] has been implemented in *Pro*SAT already, and preliminary results show good promise.

We believe that *Pro*SAT provides to the practitioners an efficient and easy-to-use tool for a wide variety of annotation problems. The results of some of these predictions can be used to assist in solving the overarching 3D structure prediction problem. In the future, we intend to use this annotation framework to predict various 1D features of a protein and effectively integrate them to provide valuable supplementary information for determining the 3D structure of proteins.

References

1. Ahmed, R., Rangwala, H., Karypis, G.: Toptmh: Topology predictor for transmembrane alpha-helices. In: European Conference in Machine Learning (in press, 2008), www.cs.umn.edu/~karypis
2. Altschul, S.F., Madden, L.T., Schäffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped blast and psi-blast: a new generation of protein database search programs. Nucleic Acids Research 25(17), 3389–3402 (1997)
3. Bach, F., Lanckriet, G.R.G., Jordan, M.I.: Multiple kernel learning, conic duality, and the smo algorithm. In: Proceedings of the 2004 International Conference on Machine Learning (2004)
4. Cheng, J., Sweredoski, M.J., Baldi, P.: Accurate prediction of protein disordered regions by mining protein structure data. Data Mining and Knowledge Discovery 11(3), 213–222 (2005)
5. de Brevern, A.G., Etchebest, C., Hazout, S.: Bayesian probabilistic approach for predicting backbone structures in terms of protein blocks. Proteins 41(3), 271–287 (2000)
6. Dosztányi, Z., Csizmok, V., Tompa, P., Simon, I.: Iupred: web server for the prediction of intrinsically unstructured regions of proteins based on estimated energy content. Bioinformatics 21(16), 3433–3434 (2005)
7. Dunker, A.K., Brown, C.J., Lawson, J.D., Iakoucheva, L.M., Obradovic, Z.: Intrinsic disorder and protein function. Biochemistry 41(21), 6573–6582 (2002)
8. Gribskov, M., Robinson, N.: Use of receiver operating characteristic (roc) analysis to evaluate sequence matching. Computational Chemistry 20, 25–33 (1996)
9. Hirose, S., Shimizu, K., Kanai, S., Kuroda, Y., Noguchi, T.: Poodle-l: a two-level svm prediction system for reliably predicting long disordered regions. Bioinformatics 23(16), 2046–2053 (2007)

10. Joachims, T.: Making large-Scale SVM Learning Practical. In: Advances in Kernel Methods: Support Vector Learning. MIT-Press, Cambridge (1999)
11. Jones, D.T.: Protein secondary structure prediction based on position-specific scoring matricies. J. Mol. Biol. 292, 195–202 (1999)
12. Jones, D.T.: Improving the accuracy of transmembrane protein topology prediction using evolutionary information. Bioinformatics 23(5), 538–544 (2007)
13. Kall, L., Krogh, A., Sonnhammer, E.L.L.: A combined transmembrane topology and signal peptide prediction method. Journal of Molecular Biology 338, 1027–1036 (2004)
14. Karchin, R., Cline, M., Mandel-Gutfreund, Y., Karplus, K.: Hidden markov models that use predicted local structure for fold recognition: alphabets of backbone geometry. Proteins 51(4), 504–514 (2003)
15. Karypis, G.: Yasspp: better kernels and coding schemes lead to improvements in protein secondary structure prediction. Proteins 64(3), 575–586 (2006)
16. Kauffman, C., Rangwala, H., Karypis, G.: Improving homology models for protein-ligand binding sites. In: LSS Comput. Syst. Bioinformatics Conference, number 08-012, San Francisco, CA (in press, 2008), `www.cs.umn.edu/~karypis`
17. Kernytsky, A., Rost, B.: Static benchmarking of membrane helix predictions. Nucleic. Acids Res. 31(13), 3642–3644 (2003)
18. Kinjo, A.R., Nishikawa, K.: Crnpred: highly accurate prediction of one-dimensional protein structures by large-scale critical random networks. BMC Bioinformatics 7(401) (2006)
19. Lanckriet, G.R.G., Deng, M., Cristianini, N., Jordan, M.I., Noble, W.S.: Kernel-based data fusion and its application to protein function prediction in yeast. In: Proceedings of the 2004 Pacific Symposium on Biocomputing (2004)
20. Ofran, Y., Mysore, V., Rost, B.: Prediction of dna-binding residues from sequence. Bioinformatics 23(13), i347–i353 (2007)
21. Pollastri, G., Baldi, P., Farselli, P., Casadio, R.: Prediction of coordination number and relative solvent accessibility in proteins. Proteins: Structure, Function, and Genetics 47, 142–153 (2002)
22. Pollastri, G., Przybylski, D., Rost, B., Baldi, P.: Improving the prediction of protein secondary structure in three and eight classes using recurrent neural network and profiles. Proteins: Structure, Function, and Bioinformatics 47, 228–235 (2002)
23. Rangwala, H., Karypis, G.: frmsdpred: Predicting local rmsd between structural fragments using sequence information. Proteins (Feburary 2008)
24. Ratsch, G., Sonnenburg, S., Schafer, C.: Learning interpretable svms for biological sequence classification. BMC Bioinformatics 7(S9) (2006)
25. Rost, B.: Phd: predicting 1d protein structure by profile based neural networks. Meth. in Enzym. 266, 525–539 (1996)
26. Schwede, T., Kopp, J., Guex, N., Peltsch, M.C.: Swiss-model: An automated protein homology-modeling server. Nucleic Acids Research 31(13), 3381–3385 (2003)
27. Song, J., Burrage, K.: Predicting residue-wise contact orders in proteins by support vector regression. BMC Bioinformatics 7(425) (2006)
28. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, Heidelberg (1995)
29. Whaley, R.C., Dongarra, J.: Automatically Tuned Linear Algebra Software. In: Ninth SIAM Conference on Parallel Processing for Scientific Computing (1999) CD-ROM Proceedings

Dynamic Exponential Family Matrix Factorization

Kohei Hayashi[1], Jun-ichiro Hirayama[2], and Shin Ishii[2]

[1] Grad. School of Information Science, Nara Inst. of Science and Technology, Japan
[2] Graduate School of Informatics, Kyoto University, Japan
{kohei-h,junich-h,ishii}@sys.i.kyoto-u.ac.jp

Abstract. We propose a new approach to modeling time-varying relational data such as e-mail transactions based on a dynamic extension of matrix factorization. To estimate effectively the true relationships behind a sequence of noise-corrupted relational matrices, their dynamic evolutions are modeled in a space of low-rank matrices. The observed matrices are assumed as to be sampled from an exponential family distribution that has the low-rank matrix as natural parameters. We apply the sequential Bayesian framework to track the variations of true parameters. In the experiments using both artificial and real-world datasets, we demonstrate our method can appropriately estimate time-varying true relations based on noisy observations, more effectively than existing methods.

1 Introduction

Methods for modeling and analyzing relational data, i.e., collections of quantified relationships between (usually two) objects, are recently attracting much attention in the fields of data mining and machine learning. Examples of such data are link structure of WEB, social networks within human groups, patterns of coauthorship of academic papers, or gene-gene interactions in biological networks. Among various research topics in this field (often called "link mining"; see [1] for an extensive review), this study addressed the issue of identifying underlying relationships based on stochastic observations, i.e., the "link prediction" problem.

Conventionally, pairwise relationships appeared in various contexts are commonly represented as graphs, networks, or equivalently weighted adjacency matrices. For example, consider a social relationship among people which is measured by their e-mail transactions with each other. We regard the frequency of e-mails from user i to user n as an observed link (or non-link if zero frequency) x_{in}. Then we assume that the matrix $\mathbf{X} = [x_{in}]$ is a noise-corrupted observation of underlying true relationship, $\widetilde{\mathbf{X}}$, of these users. The link prediction problem is to estimate the true matrix $\widetilde{\mathbf{X}}$ from the observation $\mathbf{X}$, where $\mathbf{X}$ may contain missing values[1].

[1] The usual definition of link prediction is to predict a missing x_{in} by estimating $\widetilde{x}_{in}$, while we generalize this to smoothing (denoising) an observed (but noise-corrupted) x_{in}. We also assume here such situations that no additional features about the objects (e.g., gender or occupation of people) are available.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 452–462, 2009.
© Springer-Verlag Berlin Heidelberg 2009

For such problems to recover the true matrix $\widetilde{\mathbf{X}}$ from observed $\mathbf{X}$, matrix factorization techniques [2] – [7] have been actively studied in the machine learning field. This is a general technique of approximating an observed matrix $\mathbf{X}$ by a low-rank matrix $\widetilde{\mathbf{X}} = \mathbf{U}^{\mathrm{T}}\mathbf{V}$ under various loss criteria, in order to effectively reduce the degree of freedom of $\mathbf{X}$. Since the true relation is likely to have structures of a low degree of freedom and the rank-reduced representation is usually good for generalization, the low-rank matrix representation is advantageous for prediction. The framework is general and widely applicable, especially in such situations that one does not have enough knowledge of the true relational structure.

Matrix factorization techniques, however, only consider the cases that target data are sampled from a stationary environment. Many real-world relational data, on the other hand, have such characters that their statistical properties are dynamically changing over time [8]. In the case of e-mail transactions, the true social links are inherently dynamic over long time scales. Observed links at a certain time point (e.g., e-mail counts at a day or a week) are generated from a temporary state of latent relationship at the time, and thus they have distinct statistical characteristics at different time points. Existing matrix factorization techniques have only addressed a single $\mathbf{X}$ but not a temporal series $\mathbf{X}^1, \mathbf{X}^2, \ldots$.

To address this issue, we propose a dynamic extension of matrix factorization in this study. Our model utilizes the framework of generalized linear models (GLMs) [9] to deal with various kinds of data, employing the original idea of [2] in the context of matrix factorization. The high flexibility of this framework is especially useful when dealing with binary or count data such as in e-mail transaction data.

The rest of this article is organized as follows. In Sec. 2, we introduce the original GLM-based matrix factorization (called "exponential family matrix factorization (EMF)") of [2]. In Sec. 3, we derive the Bayesian inference for this model by using the Laplace approximation, and then extend the model into incorporating a state-space model. A Kalman filter style estimate for this model is also described. In Sec. 4, the results with synthetic and real datasets are shown. In Sec. 5, we summarize our current study and discuss future works.

2 Exponential Family Matrix Factorization

Let $\mathcal{U}$ and $\mathcal{V}$ be two sets of R and C objects under consideration, respectively. The pairwise relationship between $\mathcal{U}$ and $\mathcal{V}$ can be stochastically observed as an $R \times C$ matrix $\mathbf{X} = [x_{in}]$, where x_{in} is the quantity representing the relation between the i-th element of $\mathcal{U}$ and the j-th element of $\mathcal{V}$. In this article, we assume $\mathbf{X}$ is asymmetric to allow "directional" relation.

Now we assume that the observed matrix $\mathbf{X}$ is generated stochastically around its expectation $\widetilde{\mathbf{X}}$ (the "true" matrix). In a probabilistic formulation of matrix factorization (e.g., [5], [10]) , $\widetilde{\mathbf{X}}$ is often modeled as

$$\widetilde{\mathbf{X}} = \mathbf{U}^{\mathrm{T}}\mathbf{V},$$

with a $K \times R$ matrix $\mathbf{U}$ and a $K \times C$ matrix $\mathbf{V}$. 'T' denotes the transpose. Usually we assume $K < \min(R, C)$ so that $\widetilde{\mathbf{X}}$ has a degenerated rank. If we assume each x_{in} is independently generated from Gaussian of mean $\tilde{x}_{in}$ and variance σ^2, the generative model is simply written as

$$\mathbf{X} = \mathbf{U}^{\mathrm{T}}\mathbf{V} + \mathbf{E}, \qquad \mathbf{E} \sim \prod_{i}^{R} \prod_{n}^{C} N(\varepsilon_{in} \mid 0, \sigma^2). \tag{1}$$

In this case, the maximum likelihood (i.e., least square) estimation of $\mathbf{U}$ and $\mathbf{V}$ correspond to the singular value decomposition (SVD). However, the Gaussian assumption is not suitable for some types of data, such as binary, count, and non-negative ones.

Gordon [2] employed the GLM framework to extend matrix factorization techniques to be applicable to non-Gaussian situations. A typical GLM (with the "canonical link") assumes that a target variable is distributed as an exponential family that has a linear regression function of the natural parameter; the expectation $\widetilde{\mathbf{X}}$ under the model then becomes a nonlinear function of input variables. As the same manner, Gordon's model assumed that the expectation $\widetilde{\mathbf{X}}$ is given by a nonlinear function both of $\mathbf{U}$ and $\mathbf{V}$, that is,

$$\widetilde{\mathbf{X}} = \mathbf{f}(\mathbf{U}^{\mathrm{T}}\mathbf{V})$$

where $\mathbf{f}(\cdot)$ is a nonlinear function defined below. More concretely, $\mathbf{X}$ is assumed to be distributed as the following generative model:

$$p(\mathbf{X} \mid \mathbf{U}, \mathbf{V}) = \prod_{i}^{R} \prod_{n}^{C} \mathrm{Exp}(x_{in} \mid \theta_{in}), \tag{2}$$

$$= \exp\left[\mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{\Theta}) - \sum_{in} F(\theta_{in}) + \sum_{in} G(x_{in}) \right], \tag{3}$$

where $\mathbf{\Theta} = \mathbf{U}^{\mathrm{T}}\mathbf{V}$, and $\mathbf{u}_i$ and $\mathbf{v}_n$ are the i-th and n-th column vectors of $\mathbf{U}$ and $\mathbf{V}$, respectively. Here, we have defined

$$\mathrm{Exp}(x \mid \theta) \equiv \exp[x\theta - F(\theta) + G(x)], \tag{4}$$

where θ is a natural parameter, and $F(\theta) = \ln \int \exp[x\theta + G(x)]\mathrm{d}x$ is the log-normalization term. In this case, since the sufficient statistics is $\mathbf{X}$, the mean parameter of this distribution is equal to $\widetilde{\mathbf{X}}$ by definition. Thus we have $\widetilde{x}_{in} = f(\theta_{in}) = F'(\theta_{in})$ where F' is the first derivative of F, from the well-known property of exponential family. We note that many important distributions are represented by the form of Eq. (4), such as Bernoulli (binary), Poisson (count) and exponential (non-negative) distributions. In this article, we call this model as exponential family matrix factorization (EMF). In [2], an efficient Newton's method-based procedure to estimate $\mathbf{U}$ and $\mathbf{V}$ with quadratic regularization (i.e., an MAP estimation under Gaussian priors) was presented.

3 Dynamic EMF

3.1 Model

Let $\mathbf{X}^t$ denote an observation at a discretized time step t. Now we have a time-series $\mathcal{D}^T \equiv \{\mathbf{X}^1, \ldots, \mathbf{X}^T\}$ and our goal is to obtain a sequence of low-rank matrices $\{\widetilde{\mathbf{X}}^t\}$ which approximate $\{\mathbf{X}^t\}$. To naively perform this has a very large number of parameters. In order to reduce the number by utilizing the character of the time-series, one good idea is directly to model the dynamics of the underlying features of the data. To this end, we construct a dynamical model that extends the original EMF model to be suitable for time-series relational data.

We assume that the parameter $\mathbf{\Theta} = \mathbf{U}^{\mathrm{T}}\mathbf{V}$ in Eq. (3) now changes with time. Let $\mathbf{\Theta}^t$ be the parameter at time step t. We also denote $\mathbf{U}^t = [\mathbf{u}_1^t, \ldots, \mathbf{u}_R^t]$ and $\mathbf{V}^t = [\mathbf{v}_1^t, \ldots, \mathbf{v}_C^t]$ as the two matrices such that $\mathbf{\Theta}^t = (\mathbf{U}^t)^{\mathrm{T}}\mathbf{V}^t$. To model the dynamics of the low-rank matrix $\mathbf{\Theta}^t$, we assume random walks for $\mathbf{U}^t$ and $\mathbf{V}^t$:

$$\mathbf{U}^t = \mathbf{U}^{t-1} + \mathbf{\Gamma}^t, \qquad \Gamma_{ki}^t \sim N(0, \gamma^2), \tag{5a}$$

$$\mathbf{V}^t = \mathbf{V}^{t-1} + \mathbf{\Xi}^t, \qquad \Xi_{kn}^t \sim N(0, \xi^2). \tag{5b}$$

The initial values $\mathbf{u}_i^1$ and $\mathbf{v}_n^1$ are assumed to be sampled from isotropic Gaussian $N(\mathbf{u}_i^1|0, \sigma^2\mathbf{I})$ and $N(\mathbf{v}_n^1|0, \omega^2\mathbf{I})$, respectively. Given $\mathbf{U}^t$ and $\mathbf{V}^t$, the observation $\mathbf{X}^t$ is generated from the EMF model, that is,

$$p(\mathbf{X}^t \mid \mathbf{U}^t, \mathbf{V}^t) = \exp\left[\mathrm{tr}\left((\mathbf{X}^t)^{\mathrm{T}}\mathbf{\Theta}^t\right) - F(\mathbf{\Theta}^t) + G(\mathbf{X}^t)\right] \tag{6}$$

where $\mathbf{\Theta}^t = (\mathbf{U}^t)^{\mathrm{T}}\mathbf{V}^t$. The graphical model is depicted in Figure 1.

One may concern about the scale indeterminacy in our parametrization $\mathbf{\Theta} = \mathbf{U}^T\mathbf{V}$, that is, for an arbitrary constant a, replacing $\mathbf{U}$ and $\mathbf{V}$ respectively as $a\mathbf{U}$ and $(1/a)\mathbf{V}$ does not change the likelihood (6). In [2], this indeterminacy was resolved by introducing quadratic regularization terms which controlled their *a priori* scales. Similarly, in our model, by appropriately setting the *a priori* scale

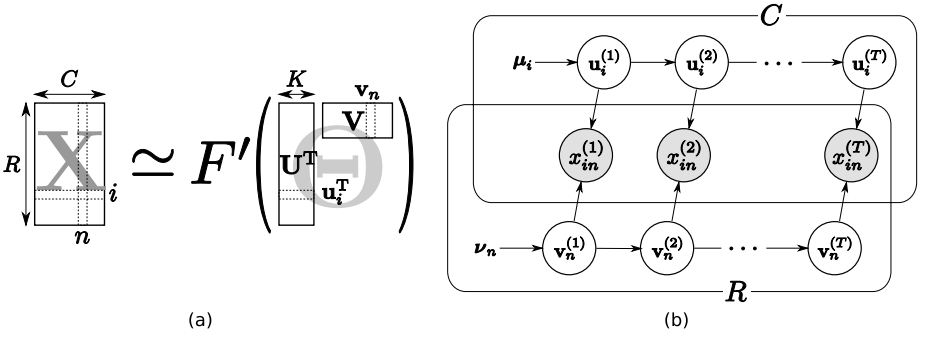


Fig. 1. (a) An overview of exponential family matrix factorization, which decomposes an observation $\mathbf{X}$ into $\mathbf{U}$ and $\mathbf{V}$ in a space of natural parameter $\mathbf{\Theta}$ by using $\mathbf{f}$, the inverse of the canonical link function. (b) Graphical model of our model.

(variance) parameters, σ, ω, γ, and ξ, either by estimating or by hand with a further regularization, the indeterminacy is expected to be resolved. Furthermore, our Bayesian scheme rather than the point estimates would also be suitable for such situations.

3.2 Laplace Approximation

Now we assume the (conditional) prior distributions of $\mathbf{U}^t$ and $\mathbf{V}^t$ based on the past sequence $\mathcal{D}^{t-1}$ are obtained as Gaussian forms,

$$p(\mathbf{U}^t \mid \mathcal{D}^{t-1}) = \prod_i^R N(\mathbf{u}_i^t \mid \boldsymbol{\mu}_i^t, \boldsymbol{\Sigma}_i^t), \quad p(\mathbf{V}^t \mid \mathcal{D}^{t-1}) = \prod_n^C N(\mathbf{v}_n^t \mid \boldsymbol{\nu}_n^t, \boldsymbol{\Omega}_n^t). \quad (7)$$

When a new datum $\mathbf{X}^t$ is observed, we estimate posterior distributions of $\mathbf{U}^t$ and $\mathbf{V}^t$ in a Bayesian manner. However, the calculation of the posteriors with these Gaussian priors (7) and EMF likelihood (2) in closed forms is difficult. It should be noted that this is also true even when the likelihood is also a Gaussian; the Gaussian prior is conjugate for $\boldsymbol{\Theta}$, but not for $\mathbf{U}$ or $\mathbf{V}$. If we assume a Gaussian prior directly on $\boldsymbol{\Theta}$, the posterior of $\boldsymbol{\Theta}$ is also Gaussian; however, the rank constraint on $\boldsymbol{\Theta}$ is hard to be applied.

In this study, we approximate them by the Laplace approximation. This makes the posteriors also of Gaussian form, which is useful to derive sequential Bayesian filters. According to the Laplace approximation in a naive way, however, we need to compute the full covariance matrix of size $(RK + CK)^2$ so as to handle all the combinations between $\mathbf{U}$ and $\mathbf{V}$. To reduce the computational cost, we further assume the column-wise posterior independence for $\mathbf{U}$ and $\mathbf{V}$ (a similar assumption is also used in [3]). Then, the posterior after observing the new datum $\mathbf{X}^t$ is given by

$$p(\mathbf{U}^t, \mathbf{V}^t \mid \mathcal{D}^t) \simeq \prod_i^R N(\mathbf{u}_i^t \mid \widehat{\mathbf{u}}_i^t, \mathbf{P}_i^t) \prod_n^C N(\mathbf{v}_n^t \mid \widehat{\mathbf{v}}_n^t, \mathbf{Q}_n^t), \quad (8)$$

where $\widehat{\mathbf{u}}_i^t$ and $\widehat{\mathbf{v}}_n^t$ are the MAP estimates of $\mathbf{u}_i^t$ and $\mathbf{v}_n^t$, respectively. $\mathbf{P}_i^t$ and $\mathbf{Q}_n^t$ are the inverse Hessian of the log-posterior around the MAP points with respect to $\mathbf{u}_i^t$ and $\mathbf{v}_n^t$, respectively. This approximation allows us to maintain only $(R + C)$ covariance matrices of size K^2.

The posterior means, $\widehat{\mathbf{U}}^t$ and $\widehat{\mathbf{V}}^t$, can be obtained in the same manner as in the original EMF [2]. The inverse of covariance matrices $\mathbf{P}_i^t$ and $\mathbf{Q}_n^t$ are given by

$$\left(\mathbf{P}_i^t\right)^{-1} = \widehat{\mathbf{V}}^t \widehat{\mathbf{D}}_i^t (\widehat{\mathbf{V}}^t)^{\mathrm{T}} + (\boldsymbol{\Sigma}_i^t)^{-1}, \ \left(\mathbf{Q}_n^t\right)^{-1} = \widehat{\mathbf{U}}^t \widehat{\mathbf{D}}_n^t (\widehat{\mathbf{U}}^t)^{\mathrm{T}} + (\boldsymbol{\Omega}_n^t)^{-1}, \quad (9)$$

where $\widehat{\mathbf{D}}_i^t = \mathrm{diag}(f'(\widehat{\theta}_{i,1}^t), \ldots, f'(\widehat{\theta}_{i,C}^t))$, $\widehat{\mathbf{D}}_n^t = \mathrm{diag}(f'(\widehat{\theta}_{1,n}^t), \ldots, f'(\widehat{\theta}_{R,n}^t))$, $\widehat{\theta}_{in}^t = (\widehat{\mathbf{u}}_i^t)^{\mathrm{T}} \widehat{\mathbf{v}}_n^t$ and f' is the second-order derivative of F. From the property of F, which is the cumulant-generating function, diagonal elements of $\widehat{\mathbf{D}}_i^t$ and $\widehat{\mathbf{D}}_n^t$ are

Algorithm 1. Estimation procedure of dynamic EMF

Input: $\{\boldsymbol{\mu}_i^1, \boldsymbol{\Sigma}_i^1 | i = 1, \ldots, R\}, \gamma^2, \{\boldsymbol{\nu}_n^1, \boldsymbol{\Omega}_n^1 | n = 1, \ldots, C\}, \xi^2$
  for $t = 1, \ldots, T$ **do**
    if $t > 1$ **then**
      // transition
      for $i = 1, \ldots, R$ **do**
        $\boldsymbol{\mu}_i^t \leftarrow \widehat{\mathbf{u}}_i^{t-1}$
        $\boldsymbol{\Sigma}_i^t \leftarrow \gamma^2 \mathbf{I} + \mathbf{P}_i^{t-1}$
      end for
      for $n = 1, \ldots, C$ **do**
        $\boldsymbol{\nu}_n^t \leftarrow \widehat{\mathbf{v}}_n^{t-1}$
        $\boldsymbol{\Omega}_n^t \leftarrow \xi^2 \mathbf{I} + \mathbf{Q}_n^{t-1}$
      end for
    end if
    // MAP estimation with the Newton's method [2]. $\mathbf{x}_i$ *and* $\mathbf{x}_n$ *represent* i*-th row and* n*-th column vectors of* $\mathbf{X}$ *(omitting the subscript* t*)*
    repeat
      solve $(\mathbf{V}\mathbf{D}_i\mathbf{V}^{\mathrm{T}} + \boldsymbol{\Sigma}_i)\mathbf{u}_i^{\text{new}} = \left(\mathbf{x}_{i,}^{\mathrm{T}} - \mathbf{f}(\mathbf{u}_i^{\mathrm{T}}\mathbf{V}) + \mathbf{u}_i^{\mathrm{T}}\mathbf{V}\mathbf{D}_i\right)\mathbf{V}^{\mathrm{T}} + \boldsymbol{\Sigma}_i\boldsymbol{\mu}_i$ for $i = 1, \ldots, R$
      solve $(\mathbf{U}\mathbf{D}_n\mathbf{U}^{\mathrm{T}} + \boldsymbol{\Omega}_n)\mathbf{v}_n^{\text{new}} = \mathbf{U}(\mathbf{x}_n - \mathbf{f}(\mathbf{U}^{\mathrm{T}}\mathbf{v}_n) + \mathbf{D}_n\mathbf{U}^{\mathrm{T}}\mathbf{v}_n) + \boldsymbol{\Omega}_n\boldsymbol{\nu}_n$ for $n = 1, \ldots, C$
    until convergence
    // Laplace approximation
    $(\mathbf{P}_i^t)^{-1} \leftarrow \widehat{\mathbf{V}}^t\widehat{\mathbf{D}}_i^t(\widehat{\mathbf{V}}^t)^{\mathrm{T}} + (\boldsymbol{\Sigma}_i^t)^{-1}$ for $i = 1, \ldots, R$
    $(\mathbf{Q}_n^t)^{-1} \leftarrow \widehat{\mathbf{U}}^t\widehat{\mathbf{D}}_n^t(\widehat{\mathbf{U}}^t)^{\mathrm{T}} + (\boldsymbol{\Omega}_n^t)^{-1}$ for $n = 1, \ldots, C$
  end for
  return $\{\widehat{\mathbf{U}}^t, \widehat{\mathbf{V}}^t | t = 1 \ldots, T\}$ for the target estimates.

the variances of x_{in}'s under the EMF model with parameter $\widehat{\boldsymbol{\Theta}}^t$. We note that, although we have assumed conditional independences in the posterior (8), the estimations of $\mathbf{u}_i^t$ and $\mathbf{v}_n^t$ are *not* independently done; they actually interact with each other, through the estimation of mean and variance of x_{in} under the model, where calculation requires all the parameters in $\mathbf{U}^t$ and $\mathbf{V}^t$.

3.3 Sequential Bayesian Inference

Due to the Gaussian form of the posterior and the independence assumption of Eq. (8), we can utilize the Kalman filter-like sequential Bayesian inference for $\mathbf{u}_i$ and $\mathbf{v}_n$. When we have the $(t-1)$th posterior $p(\mathbf{u}_i^{t-1}|\mathcal{D}^{t-1})$, the prior distribution at the next step t can be obtained by a marginalization: $p(\mathbf{u}_i^t|\mathcal{D}^{t-1}) = \int p(\mathbf{u}_i^t|\mathbf{u}_i^{t-1})p(\mathbf{u}_i^{t-1}|\mathcal{D}^{t-1})\mathrm{d}\mathbf{u}_i^{t-1}$. Due to the Gaussian assumption of state transition, they are given by

$$p(\mathbf{u}_i^t \mid \mathcal{D}^{t-1}) = N(\boldsymbol{\mu}_i^t, \boldsymbol{\Sigma}_i^t), \qquad \boldsymbol{\mu}_i^t = \widehat{\mathbf{u}}_i^{t-1}, \qquad \boldsymbol{\Sigma}_i^t = \gamma^2\mathbf{I} + \mathbf{P}_i^{t-1} \tag{10a}$$

$$p(\mathbf{v}_n^t \mid \mathcal{D}^{t-1}) = N(\boldsymbol{\nu}_n^t, \boldsymbol{\Omega}_n^t), \qquad \boldsymbol{\nu}_n^t = \widehat{\mathbf{v}}_n^{t-1}, \qquad \boldsymbol{\Omega}_n^t = \xi^2\mathbf{I} + \mathbf{Q}_n^{t-1}. \tag{10b}$$

After observing $\mathbf{X}^t$, the posterior distributions are updated by Eqs. (8)-(9).

In this study, we only employ the forward inference (filtering) but not the backward one (smoothing), because we primarily assume that our algorithm is applied in an online manner. When considering computationally heavier but more accurate prediction, the backward inference, at least back to the recent part, would be necessary. However, our current algorithm still shows high performances as seen in simulation experiments in the next section. Finally, with the posterior of Eq. (8), we obtain a prediction of $\mathbf{X}^t$ as $\widetilde{\mathbf{X}}^t = \mathbf{f}(\widehat{\boldsymbol{\Theta}}^t)$, where $\widehat{\boldsymbol{\Theta}}^t = (\widehat{\mathbf{U}}^t)^{\mathrm{T}}\widehat{\mathbf{V}}^t$. The overall procedure is summarized in Algorithm 1.

4 Experiments

4.1 Synthetic Data

To evaluate the basic performance of our method, we conducted a simulation experiment with a synthesized dataset. We first prepared a sequence of rank-4 natural parameters $\{\boldsymbol{\Theta}^1, \ldots, \boldsymbol{\Theta}^{100}\}$, such that the expectation $\widetilde{\mathbf{X}}$ varied as illustrated in Fig. 2(a). In these figures, each matrix is aligned as a single column. Then, each observation $\mathbf{X}^t$ was generated from the Poisson distribution of natural parameter $\boldsymbol{\Theta}^t$ (Figure 2 (b)).

For comparison, we employed SVD and the original (static) EMF both using sliding windows. That is, each estimate of a low-rank matrix at time step t was based on the previous $w+1$ step's observations at $t-w, t-w+1, \ldots, t$. In our Poisson case, this is equivalent to the use of a compound data $\bar{\mathbf{X}}^t = \frac{1}{w+1}\sum_{i=0}^{w}\mathbf{X}^{t-i}$ as the observation at t. We evaluated root mean squared errors (RMSE) between the true sequence $\widetilde{\mathbf{X}}^t$ and the parameter sequence estimated by

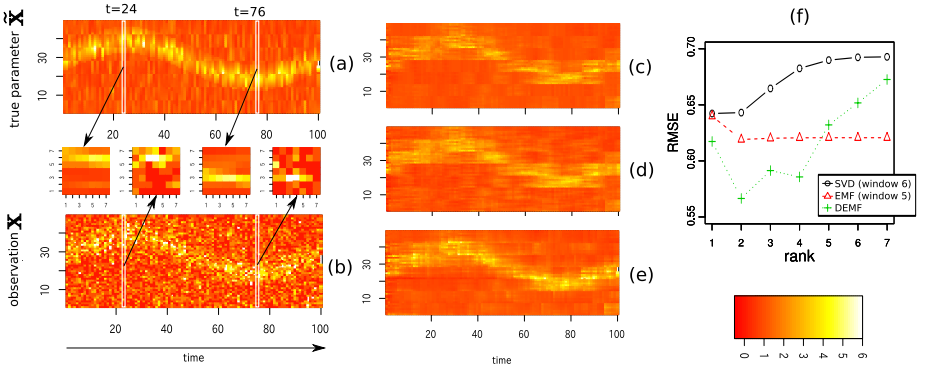


Fig. 2. The experiment result of synthetic data. Each panel in (a)-(e) represents the matrix in which t-th column vector corresponds to vectorization of matrix $\mathbf{X}^t$. (a) True expectation parameters $\{\widetilde{\mathbf{X}}^t\}$. (b) Noisy observation $\{\mathbf{X}^t\}$ with Poisson noise. (c)-(e) represent the best results of estimated expectation parameter by (c) SVD with window size $w = 6$, (d) original EMF with $w = 5$, and (e) Dynamic EMF. (f) Plotting the RMSE between the true parameter and the parameter estimated by three methods.

each of these three methods. In this experiment, we fixed the transition variances γ^2, ξ^2 in (5) at the true value ($= 10^{-4}$) and did not estimate them.

Figure 2(c)-(e) visualize typical results by SVD, EMF, and our method, respectively. From these panels, a slight improvement by our method can be seen, that is, the result (Figure 2(e)) is sharper and clearer than the others. Although the difference is not very clear in these figures, we can confirm the improvement qualitatively by comparing the RMSE by the three methods (Figure 2(f)). For SVD and EMF, the best results obtained with the optimal ws are presented[2]. We can see that, by regularizing the rank to be low, our method outperforms the other methods in terms of RMSE.

4.2 E-Mail Data

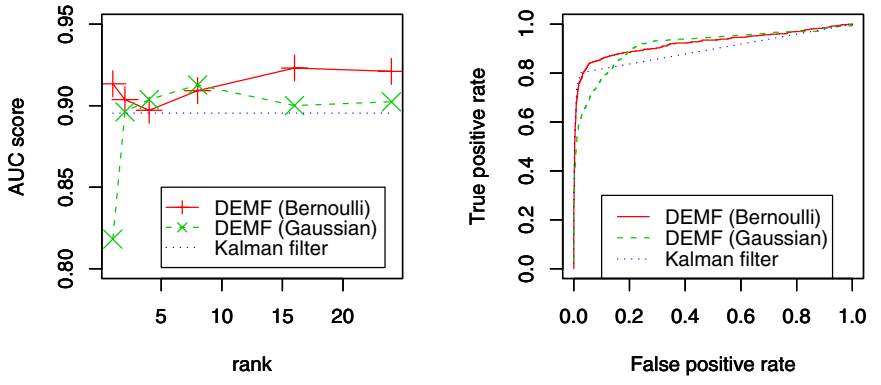
Next we tested the effectiveness of our method for link prediction with a real-world dataset. In this experiment, we used the Enron e-mail dataset [11], the collection of 252,759 e-mails of 151 Enron employees in years from 1998 to 2003. In our experiment, we only used the data from Jan 1st 2001 to Jun 30th 2001 (for 181 days). Then we had a time-series of 181 observations $\{\mathbf{X}^1, \ldots, \mathbf{X}^{181}\}$ each of which corresponds to a single day, where we set $x^t_{in} = 1$ if user i send any e-mail to user n on the t-th day, otherwise set $x^t_{in} = 0$. We omitted the e-mails from/to the outside of the Enron group and only used those exchanged internally within the group.

In this experiment, we cannot know the "true" matrix $\widetilde{\mathbf{X}}$ and therefore the evaluation of RMSE is not possible. To measure the estimation performance, we then explicitly treated a part of observed matrices as missing values and used them as the targets of link prediction. We randomly chose $1,000$ and $4,000$ elements as missing values from those of positive (i.e., $x^t_{in} = 1$) and negative ($x^t_{in} = 0$), respectively. For these test data, we evaluated the area under an ROC curve (AUC) score. The AUC score [12] is widely used as an evaluation criterion in link prediction problems. AUC takes its value from 0.5 to 1: a perfect classifier gets 1 and random one gets 0.5.

For comparison, we applied the Kalman filter algorithm for each element-wise sequence $\{x^t_{in}\}_{t=1}^T$ separately. Our model used both Gaussian of unit variance and Bernoulli models in DEMF at their rank $= 1, 2, 4, 8, 16, 24$. The other settings of our model were the same as those in the previous experiment.

The experimental results are shown in Figure 3, in which we can see the superiority of our method. Except rank-1 Gaussian DEMF, all the AUC scores by our DEMF were higher than those by the Kalman filter. Our model effectively used the correlation between elements and then was successful in the link prediction task. Furthermore, DEMF in the Bernoulli model took the best score (0.923), suggesting an advantage of using the GLM framework when dealing with non-Gaussian data. Figure 3(b) shows the ROC curves by the three methods at their individual optimal scores. Although the Kalman filter could successfully predict

[2] We found that $w = 6$ and $w = 5$ are the optimal values for SVD and the original EMF, respectively.



(a) AUC scores. (b) ROC curves.

Fig. 3. Results of the missing value prediction on Enron dataset

a missing value when the true one is negative (= 0), keeping the false positive rate $\simeq 0$ as well as Bernoulli DEMF, the prediction often failed when it was positive (= 1), so that the true positive rate was low. The Kalman filter was thus not appropriate for our current problem, since it was probably over-fitted to the real observation sequence which contained only few positives. On the other hand, our DEMF avoided such an over-fitting, enjoying the restricted parametrization through the low-rank matrix factorization.

5 Conclusions and Future Works

In this study, we proposed a new probabilistic dynamic model based on the matrix factorization in order to deal with time-varying relational data. We used the Laplace approximation for deriving the sequential Bayesian estimation to track the temporal change of latent low-rank relationship effectively. We demonstrated that our model works well in experiments using both synthetic and real datasets.

Several probabilistic models of dynamic relational data have recently been proposed in the literature. Sarkar *et al.* [13] introduced a latent space model of dynamic social networks. This model estimates the distance between nodes of a social network in the Bernoulli natural parameter space and was applied to the link prediction. The idea of matrix factorization was also used in searching for the optimal initial parameter. Dynamic topic model (DTM) [14], a dynamic extension of latent Dirichlet allocation (LDA) [15], is a model of temporal evolution of relationships between documents and their words. Because LDA has an interpretation as matrix factorization [16], DTM can also be seen as a dynamic extension of matrix factorization.

In this work, the prediction of $\mathbf{X}$ was simply done by the model where parameter was set at the MAP estimate. In the Bayesian paradigm, however, it

is usual to use the predictive distribution by marginalizing the model parameters **U** and **V** with their posterior distributions. Although such an integration requires further approximation, it may be beneficial for better prediction. We are currently investigating the use of sequential Monte Carlo methods [17], also known as the particle filter, to approximate the integral as well as to improve the dynamic filtering. Furthermore, the decomposed parameters **U** and **V** can be regarded as extracted features of underlying relationship structure. These features can be useful for dynamic clustering of sequential relational data, such as social networks. This issue is also remained for our future study.

References

1. Getoor, L., Diehl, C.P.: Link mining: a survey. SIGKDD Explor. Newsl. 7(2), 3–12 (2005)
2. Gordon, G.J.: Generalized 2 linear 2 models. In: Advances in Neural Information Processing Systems, vol. 15 (2003)
3. Yu, K., Chu, W., Yu, S., Tresp, V., Xu, Z.: Stochastic relational models for discriminative link prediction. In: Schölkopf, B., Platt, J., Hoffman, T. (eds.) Advances in Neural Information Processing Systems 19, pp. 1553–1560. MIT Press, Cambridge (2007)
4. Srebro, N., Rennie, J.D.M., Jaakkola, T.S.: Maximum-margin matrix factorization. Advances in Neural Information Processing Systems 17 (2005)
5. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: Platt, J.C., Koller, D., Singer, Y., Roweis, S. (eds.) Advances in Neural Information Processing Systems 20. MIT Press, Cambridge (2008)
6. Sun, J., Tao, D., Papadimitriou, S., Yu, P.S., Faloutsos, C.: Incremental tensor analysis: theory and applications. ACM Transactions on Knowledge Discovery from Data (2008)
7. Tao, D., Song, M., Li, X., Shen, J., Sun, J., Wu, X., Faloutsos, C., Maybank, S.J.: Bayesian Tensor Approach for 3-D Face Modeling. IEEE Transactions on Circuits and Systems for Video Technology 18(10), 1397–1410 (2008)
8. Carley, K.M.: Dynamic network analysis. In: Breiger, R., Carley, K.M., Pattison, P. (eds.) Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers, Washington, DC, pp. 133–145 (2003)
9. Mccullagh, P., Nelder, J.: Generalized Linear Models, Second Edition. Chapman & Hall/CRC (August 1989)
10. Hoff, P.D.: Model averaging and dimension selection for the singular value decomposition. Journal of the American Statistical Association 102(478), 674–685 (2007)
11. Shetty, J., Adibi, J.: The Enron email dataset database schema and brief statistical report. Information Sciences Institute Technical Report, University of Southern California (2004)
12. Bradley, A.P.: The use of the area under the roc curve in the evaluation of machine learning algorithms. Pattern Recognition 30(7), 1145–1159 (1997)
13. Sarkar, P., Moore, A.W.: Dynamic social network analysis using latent space models. SIGKDD Explor. Newsl. 7(2), 31–40 (2005)

14. Blei, D.M., Lafferty, J.D.: Dynamic topic models. In: ICML 2006: Proceedings of the 23rd international conference on Machine learning, pp. 113–120. ACM Press, New York (2006)
15. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
16. Buntine, W.L.: Variational extensions to em and multinomial pca. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) ECML 2002. LNCS, vol. 2430, pp. 23–34. Springer, Heidelberg (2002)
17. Doucet, A., Defreitas, N., Gordon, N.: Sequential Monte Carlo Methods in Practice. Springer, Heidelberg (2001)

A Nonparametric Bayesian Learning Model: Application to Text and Image Categorization

Nizar Bouguila[1] and Djemel Ziou[2]

[1]Concordia Institute for Information Systems Engineering, Concordia University, Montreal, Canada, Qc, H3G 2W1
bouguila@ciise.concordia.ca
[2] Département d'Informatique, Université de Sherbrooke, Canada, Qc, J1K2R1
djemel.ziou@usherbrooke.ca

Abstract. In this paper a nonparametric Bayesian infinite mixture model is introduced. The adoption of this model is motivated by its flexibility. Indeed, it does not require the specification of the number of mixture components to be given in advance and estimates it in a principled manner. Our approach relies on the estimation of the posterior distribution of clusterings using Gibbs sampler. Through applications involving text and image categorization, we show that categorization via infinite mixture models offers a more powerful and robust performance than classic finite mixtures.

1 Introduction

Categorization is an important problem in the case of natural language texts and images processing due to the increased generation of digital documents (images or text). Learning techniques are frequently used for this task by building and training classifiers from a set of preclassified examples. Finite mixture models are one of the most used learning techniques and have several features that make them attractive for categorization and clustering problems [1]. One of the most challenging aspects, when using finite mixture models, is usually to estimate the number of clusters which best describes the data without over- or under-fitting it. For this purpose, many approaches have been suggested [2].

In this paper, we are interested in Bayesian nonparametric approaches for modeling and selection using mixture of Dirichlet processes [3] which has been shown to be a powerful alternative to select the number of clusters [4,5,6]. In contrast with classic Bayesian approaches which suppose an unknown finite number of mixture components, nonparametric Bayesian approaches assume infinitely complex models (i.e an infinite number of components). Indeed, nonparametric Bayesian approaches allow the increasing of the number of mixture components to infinity, which removes the problems underlying the selection of the number of clusters which can increase or decrease as new data arrive. Because of their simplicity and thanks to the development of MCMC techniques, infinite mixture models based on Dirichlet processes are now widely used in different domains

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 463–474, 2009.
© Springer-Verlag Berlin Heidelberg 2009

and variety of applications. The majority of the work with infinite mixture models makes the Gaussian assumption. However, we have shown in previous works that other distributions such as the Dirichlet [7] and the generalized Dirichlet [8,9,10] can give better results in some applications and are more appropriate especially when modeling proportion vectors. We propose then a Bayesian hierarchical infinite mixture model based on generalized Dirichlet distributions for text and image categorization.

The remainder of this paper is structured as follows. In section II, we present the formulation of the infinite generalized Dirichlet mixture by specifying the priors and the conditional posterior distributions. Section III gives the complete Gibbs sampling algorithm. Experimental results are illustrated in Section IV. Finally, Section V concludes the paper.

2 A Dirichlet Process Mixture of Generalized Dirichlet Distributions

2.1 The Infinite Generalized Dirichlet Mixture Model

Let $\mathcal{X} = (\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N)$ be a set of independent vectors representing N texts or images. By considering a Dirichlet process $\mathcal{X}$ can be modeled using a set of latent parameters $\{\theta_1, \ldots, \theta_N\}$ where each $\boldsymbol{X}_i$ has distribution $F(\theta_i)$ and each θ_i is drawn independently and identically from a mixing distribution G on which a Dirichlet process prior is placed:

$$\begin{aligned} \boldsymbol{X}_i|\theta_i &\sim F(\theta_i) \\ \theta_i|G &\sim G \\ G &\sim DP(G_0, \eta) \end{aligned} \tag{1}$$

where G_0 and η define a baseline distribution for the Dirichlet process prior and the concentration parameter, respectively. The Dirichlet process mixture can be viewed as the limit of a finite mixture model with a uniform Dirichlet prior (i.e a Dirichlet with hyperparameters η/M, where M is the number of clusters) [11]. Let Z_i be an integer denoting the unknown component from which $\boldsymbol{X}_i$ is drawn. The unobserved (or missing) vector $Z = (Z_1, \ldots, Z_N)$ is generally called the "membership vector" of the mixture model and the joint model over $\mathcal{X}$, Z is given by $p(\mathcal{X}, Z) = p(\mathcal{X}|Z)p(Z)$ where

$$p(\mathcal{X}|Z) = \int p(\mathcal{X}|Z, \Theta)p(\Theta)d\Theta \tag{2}$$

$$p(Z) = p(Z_1)\prod_{i=2}^{N} p(Z_i|Z_1, \ldots, Z_{N-1}) \tag{3}$$

and Θ is the set of all the parameters defining the mixture model. In addition, by taking the Dirichlet prior, we can easily show that [11]

$$p(Z_i = j|Z_1, \ldots, Z_{N-1}) = \frac{\eta/M + n_j}{\eta + N - 1} \tag{4}$$

where n_j is the number of vectors previously affected to cluster j. By taking the limit of the previous equation, as $M \to \infty$, we can show that

$$p(Z_i = j|Z_1, \ldots, Z_{N-1}) = \begin{cases} \frac{n_j}{N-1+\eta} \text{ if } n_j > 0 \text{ (cluster } j \in \mathcal{R}) \\ \frac{\eta}{N-1+\eta} \text{ if } n_j = 0 \text{ (cluster } j \in \mathcal{U}) \end{cases} \quad (5)$$

Where $\mathcal{R}$ and $\mathcal{U}$ are the sets of represented and unrepresented clusters, respectively. Note that despite the fact that the number of clusters is supposed to be infinite, the number of represented (i.e nonempty) clusters is finite and should be between 1 and N. From Eq. 5, we can note also that if a cluster is represented, its conditional prior will depend on the number of observations assigned to it. Conditional prior for unrepresented clusters depends on η and N. Indeed, according to Eq. 5 a new cluster may appear with a probability $\frac{\eta}{N-1+\eta}$. Thus, the average number of clusters M is given by [12,13] $\sum_{i=1}^{N} \frac{\eta}{\eta+i-1} \in \mathcal{O}(\eta \log N)$. which shows that the number of clusters increases only in a logarithmic manner in the number of observations.

The Dirichlet process mixture approach for clustering is based on the MCMC technique of Gibbs sampling [6] by generating the assignments of vectors according to the posterior distribution

$$p(Z_i = j|Z_{-i}, \mathcal{X}) \propto p(Z_i = j|Z_{-i}) \int p(\boldsymbol{X}_i|Z_i = j, \theta_j) p(\theta_j|Z_{-i}, \mathcal{X}_{-i}) d\theta_j \quad (6)$$

where Z_{-i} represents all the vectors assignments except Z_i and $\mathcal{X}_{-i}$ represents all the vectors except $\boldsymbol{X}_i$. An important problem now is the choice of the distributions $p(\boldsymbol{X}_i|\theta_j)$ representing the different components in our mixture model. The majority of the work with infinite mixture models makes the Gaussian assumption. In this paper, however, we introduce an infinite mixture model based on generalized Dirichlet distributions which have been shown to be able to give better results than the Gaussian [8,9,10].

2.2 Generalized Dirichlet Distribution

If the random vector $\boldsymbol{X}_i = (X_{i1}, \ldots, X_{id})$ follows a generalized Dirichlet distribution, the joint density function is given by [8]:

$$p(\boldsymbol{X}_i|\alpha_1, \beta_1, \ldots, \alpha_d, \beta_d) = \prod_{l=1}^{d} \frac{\Gamma(\alpha_l + \beta_l)}{\Gamma(\alpha_l)\Gamma(\beta_l)} X_{il}^{\alpha_l - 1} (1 - \sum_{t=1}^{l} X_{it})^{\gamma_l} \quad (7)$$

for $\sum_{l=1}^{d} X_{il} < 1$ and $0 < X_{il} < 1$ for $l = 1, \ldots, d$, where $\gamma_l = \beta_l - \alpha_{l+1} - \beta_{l+1}$ for $l = 1, \ldots, d-1$ and $\gamma_l = \beta_l - 1$. Note that the generalized Dirichlet distribution is reduced to a Dirichlet distribution [7] when $\beta_l = \alpha_{l+1} + \beta_{l+1}$. Let us introduce an interesting property of the generalized Dirichlet distribution previously used in [8] and that we shall exploit in this paper. Indeed, if a vector $\boldsymbol{X}_i$ has a generalized Dirichlet distribution with parameters $(\alpha_1, \beta_1, \ldots, \alpha_d, \beta_d)$, then we can construct

a vector $\boldsymbol{Y}_i = (Y_{i1}, \ldots, Y_{id})$ using the following geometric transformation $Y_{il} = T(X_{il})$:

$$T(X_{il}) = \begin{cases} X_{il} & \text{if } l = 1 \\ \frac{X_{il}}{(1-X_{i1}-\ldots-X_{il-1})} & \text{for } l = 2, 3, \ldots, d \end{cases} \tag{8}$$

In this vector $\boldsymbol{Y}_i$, each Y_{il}, $l = 1, \ldots, d$ has a Beta distribution with parameters α_l and β_l and then a given generalized Dirichlet distribution associated with a given cluster j can be defined as follows

$$p(\boldsymbol{Y}_i|\theta_j) = \prod_{l=1}^{d} \frac{\Gamma(s_{jl})}{\Gamma(s_{jl}m_{jl})\Gamma(s_{jl}(1-m_{jl}))} Y_{il}^{s_{jl}m_{jl}-1}(1-Y_{il})^{s_{jl}(1-m_{jl})-1} \tag{9}$$

where $s_{jl} = \alpha_{jl} + \beta_{jl}$ and $m_{jl} = \frac{\alpha_{jl}}{s_{jl}}$ and can be viewed as scales and locations, respectively. The symbol $\theta_j = (\boldsymbol{s}_j, \boldsymbol{m}_j)$ refers to the entire set of parameters to be estimated, where $\boldsymbol{s}_j = (s_{j1}, \ldots, s_{jd})$ and $\boldsymbol{m}_j = (m_{j1}, \ldots, m_{jd})$. In the following subsections, we focus on the development of the priors and posteriors of our infinite generalized Dirichlet mixture model.

2.3 Priors and Conditional Posteriors

We know that each location m_{jl} is defined in the compact support [0,1], then an appealing flexible choice as a prior is the Beta distribution, with location ε and scale ζ common to all components, which was found flexible in real applications. Thus, $\boldsymbol{m}_j$ for each cluster is given the following prior:

$$p(\boldsymbol{m}_j|\zeta, \varepsilon) \sim \prod_{l=1}^{d} \frac{\Gamma(\zeta)}{\Gamma(\zeta\varepsilon)\Gamma(\zeta(1-\varepsilon))} m_{jl}^{\zeta\varepsilon}(1-m_{jl})^{\zeta(1-\varepsilon)} \tag{10}$$

where $\boldsymbol{m}_j = (m_{j1}, \ldots, m_{jd})$. The conditional posterior distribution for $\boldsymbol{m}_j$ is obtained by multiplying the prior $p(\boldsymbol{m}_j|\zeta, \varepsilon)$ by the likelihood conditioned on Z:

$$\begin{aligned} p(\boldsymbol{m}_j|Z, \mathcal{Y}; \varepsilon, \zeta, s_j) \propto & \prod_{l=1}^{d} \left[\frac{\Gamma(s_{jl})}{\Gamma(s_{jl}m_{jl})\Gamma(s_{jl}(1-m_{jl}))} \right]^{n_j} \\ & \times \left(\prod_{Z_i=j} Y_{il} \right)^{s_{jl}m_{jl}-1} \left(\prod_{Z_i=j} (1-Y_{il}) \right)^{s_{jl}(1-m_{jl})-1} \\ & \times \prod_{l=1}^{d} \frac{\Gamma(\zeta)}{\Gamma(\zeta\varepsilon)\Gamma(\zeta(1-\varepsilon))} m_{jl}^{\zeta\varepsilon}(1-m_{jl})^{\zeta(1-\varepsilon)} \end{aligned} \tag{11}$$

where $n_j = \sum_{i=1}^{N} \mathbb{I}_{Z_i=j}$ and represents the number of vectors belonging to cluster j. The hyperparameters, ε and ζ, associated with the m_{jl} are given uniform (we have started by testing a Beta prior for ε, and the best experimental results were obtained with location equal to 1 and scale fixed to 2, which corresponds actually to a uniform distribution) and inverse Gamma priors, respectively:

$$p(\varepsilon) \sim \mathcal{U}_{[0,1]} \tag{12}$$

$$p(\zeta|\varphi,\varrho) \sim \frac{\varrho^{\varphi}\exp(-\varrho/\zeta)}{\Gamma(\varphi)\zeta^{\varphi+1}} \tag{13}$$

Note that although inverse Gamma priors have been used in many applications as priors, they can cause serious problems if the scale is too close to zero [14]. However, we did not observe this problem in our case. Besides, we made this specific choice because as a lighter-tailed distribution, the inverse Gamma prevents to place too much mass on large scale values [15]. For the hyperparameters ε and ζ, Eq. 10 is considered as the likelihood which gives with Eqs. 12 and 13 the following:

$$p(\varepsilon|\boldsymbol{m}_1,\ldots,\boldsymbol{m}_M;\zeta) \propto \frac{\Gamma(\zeta)^{Md}}{\Gamma(\zeta\varepsilon)^{Md}\Gamma(\zeta(1-\varepsilon))^{Md}} \prod_{j=1}^{M}\prod_{l=1}^{d} m_{jl}^{\zeta\varepsilon}(1-m_{jl})^{\zeta(1-\varepsilon)} \tag{14}$$

$$p(\zeta|\boldsymbol{m}_1,\ldots,\boldsymbol{m}_M;\varepsilon,\varphi,\varrho) \propto \frac{\varrho^{\varphi}\exp(-\varrho/\zeta)\Gamma(\zeta)^{Md}}{\Gamma(\varphi)\zeta^{\varphi+1}\Gamma(\zeta\varepsilon)^{Md}\Gamma(\zeta(1-\varepsilon))^{Md}} \prod_{j=1}^{M}\prod_{l=1}^{d} m_{jl}^{\zeta\varepsilon}(1-m_{jl})^{\zeta(1-\varepsilon)} \tag{15}$$

Since the scale s_{jl} control the dispersion of the distributions, a common choice as a prior is an inverse gamma with shape σ and scale ϖ common to all components [15], then

$$p(\boldsymbol{s}_j|\sigma,\varpi) \sim \prod_{l=1}^{d} \frac{\varpi^{\sigma}\exp(-\varpi/s_{jl})}{\Gamma(\sigma)s_{jl}^{\sigma+1}} \tag{16}$$

Having this prior, the conditional posterior for s_j is

$$\begin{aligned} p(\boldsymbol{s}_j|Z,\mathcal{Y};\sigma,\varpi,\boldsymbol{m}_j) \propto & \left[\prod_{l=1}^{d}\frac{\Gamma(s_{jl})}{\Gamma(s_{jl}m_{jl})\Gamma(s_{jl}(1-m_{jl}))}\right]^{n_j} \\ & \times \left(\prod_{Z_i=j} Y_{il}\right)^{s_{jl}m_{jl}-1}\left(\prod_{Z_i=j}(1-Y_{il})\right)^{s_{jl}(1-m_{jl})-1} \\ & \times \prod_{l=1}^{d}\frac{\varpi^{\sigma}\exp(-\varpi/s_{jl})}{\Gamma(\sigma)s_{jl}^{\sigma+1}} \end{aligned} \tag{17}$$

The hyperparameters, σ and ϖ, associated with the s_{jl} are given inverse Gamma and exponential priors, respectively:

$$p(\sigma|\lambda,\mu) \sim \frac{\mu^{\lambda}\exp(-\mu/\sigma)}{\Gamma(\lambda)\sigma^{\lambda+1}} \tag{18}$$

$$p(\varpi|\phi) \sim \phi\exp(-\phi\varpi) \tag{19}$$

For the hyperparameters σ and ϖ, Eq. 16 is considered as the likelihood which gives with Eqs. 18 and 19 the following:

$$p(\sigma|\boldsymbol{s}_1,\ldots,\boldsymbol{s}_M;\lambda,\mu,\varpi) \propto \frac{\varpi^{Md\sigma}\mu^{\lambda}\exp(-\mu/\sigma)}{\Gamma(\sigma)^{Md}\Gamma(\lambda)\sigma^{\lambda+1}} \prod_{j=1}^{M}\prod_{l=1}^{d}\frac{\exp(-\varpi/s_{jl})}{s_{jl}^{\sigma+1}} \tag{20}$$

$$p(\varpi|\boldsymbol{s}_1,\ldots,\boldsymbol{s}_M;\phi,\sigma) \propto \frac{\varpi^{Md\sigma}\phi\exp(-\phi\varpi)}{\Gamma(\sigma)^{Md}}\prod_{j=1}^{M}\prod_{l=1}^{d}\frac{\exp(-\varpi/s_{jl})}{s_{jl}^{\sigma+1}} \tag{21}$$

The choice of η is crucial in our model. In fact, the number of clusters is directly related to η which controls the generation frequency of new clusters. Then, we have chosen an inverse gamma prior for the concentration parameter η

$$p(\eta|\chi,\kappa) \sim \frac{\kappa^{\chi}\exp(-\kappa/\eta)}{\Gamma(\chi)\eta^{\chi+1}} \tag{22}$$

which gives with Eq. 5 the following posterior (for more details, see [11])

$$p(\eta|M,N;\chi,\kappa) \propto \frac{\kappa^{\chi}\exp(-\kappa/\eta)}{\Gamma(\chi)\eta^{\chi+1}}\eta^{M}\prod_{i=1}^{N}\frac{1}{i-1+\eta} = \frac{\kappa^{\chi}\exp(-\kappa/\eta)}{\Gamma(\chi)\eta^{\chi+1}}\frac{\eta^{M}\Gamma(\eta)}{\Gamma(N+\eta)} \tag{23}$$

3 Complete Algorithm

Having all the posteriors in hand, we can employ a Gibbs sampler and each iteration will be based on the following steps:

- Generate $\boldsymbol{Z}_i$ from Eq. 6, $i = 1,\ldots,N$.
- Update the number of represented components M.
- Update n_j and $P_j = \frac{n_j}{N+\eta}$, $j = 1,\ldots,M$.
- Update the mixing parameters of unrepresented components $P_U = \frac{\eta}{\eta+N}$.
- Generate $\boldsymbol{m}_j$ from Eq. 11 and $\boldsymbol{s}_j$ from Eq. 17, $j = 1,\ldots,M$.
- Update the hyperparameters: Generate ε from Eq. 14, ζ from Eq. 15, σ from Eq. 20, ϖ from Eq. 21 and η from Eq. 23.

Note that in the initialization step, the algorithm started by assuming that all the vectors are in the same cluster and the initial parameters are generated as random samples from their prior distribution. The distributions given by Eqs. 14, 15, 20, 21 and 23 are not of standard forms. However, it is possible to show that they are log-concave (i.e it is straightforward to show that the second derivatives of the logarithms of these functions are negative), then the samples generation is based on the adaptive rejection sampling (ARS) [16] to obtain the values of ε, ζ, σ, ϖ and η. The sampling of the vectors Z_i requires the evaluation of the integral in Eq. 6 which is not analytically tractable. Thus, we have used an approach, originally proposed by Neal [6] and used with success by Rasmussen [11] in the case of infinite Gaussian mixtures, which consists on approximating this integral by generating a Monte Carlo estimate by sampling from the priors of $\boldsymbol{m}_j$ and $\boldsymbol{s}_j$. More details about this sampling method are given in [6]. The sampling of $\boldsymbol{m}_j$ and $\boldsymbol{s}_j$ is more complex, since the posteriors given by Eqs. 11 and 17 do not have known forms. Thus, we have used the Metropolis-Hastings algorithm (M-H) [17].

4 Experimental Results

In this section, we describe some experiments to compare our infinite mixture model to the finite model that we previously proposed. In particular, we compare the proposed approach in this paper, for estimation and selection, with the approach in [8] and the Bayesian approach in [10]. Besides, we compare the performances of infinite generalized Dirichlet and infinite Gaussian mixtures. In these applications our specific choice for the hyperparameters is $(\varphi, \varrho, \lambda, \mu, \phi, \kappa, \chi) = (1, 1, 1, 1, 1/8, 1, 1)$ and for E^2 is diag$[0.01, \ldots, 0.01]$.

4.1 Text Categorization

In the first experiment, we test our model for the classification of three well-known data sets consisting of a set of documents and which were used, for instance, in [18]: the industry sector, the Mod Apte split of the Reuters-21570 document collection, and the 20 newsgroups data sets. The industry sector data set [1] has a vocabulary of 55,055 words and is composed of 104 classes containing 9555 documents having an average of 606 words. The first half of this data set is used for training and the second one for testing. The Mod Apte split data set [19] is a subset of the well known corpus Reuters-21578[2], has a vocabulary of 15,996 words and is composed of 90 classes containing 7,770 training documents and 3,019 test documents. The documents in this data set have an average of 70 words and are multi-labeled. The 20 newsgroups data set [3] contains 18,828 documents grouped in 20 classes (80% of the documents is used for training and the rest for testing) with a vocabulary of 61,298 words and average document length of 116 words.

The first step in our experiment was removing all stop and rare words from the vocabularies associated with each data set. After this first pre-processing, each document was represented by a vector of counts. The dimensionality of these vectors were then reduced using LDA [20]. As a result, each document was represented by a vector of proportions. The proportions vectors in the different training sets were then modeled by infinite generalized Dirichlet mixtures, using the algorithm in the previous section, and each test vector was affected to the class that gives the highest likelihood. Following [18], the performance of our model for the industry sector and newsgroups data sets was based on the precision measure given by

$$Precision = \frac{TP}{TP + FP} \tag{24}$$

where TP and FP are the numbers of true positives and false positives, respectively. Table 1 shows the classification results, averaged over 20 random selection of the training and test sets, for the industry sector and 20 newsgroups data sets.

[1] http://www.cs.umass.edu/∼mccallum/code-data.html
[2] The corpus is from http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html
[3] people.csail.mit.edu/people/jrennie/20Newsgroups

Table 1. Classification results for the industry sector and 20 newsgroups data sets

	Infinite mixture	Finite mixture + HSEM [8]	Finite mixtue + Bayesian [10]
Industry sector	0.90± 0.01	0.88± 0.03	0.88± 0.01
20 newsgroups	0.87± 0.02	0.85± 0.02	0.86± 0.1

Table 2. Classification results for the Mod Apte split data set

	Infinite mixture	Finite mixture + HSEM [8]	Finite mixtue + Bayesian [10]
Macro	0.60	0.57	0.58
Micro	0.79	0.76	0.78

The infinite model produces better results than the finite one estimated either using the HSEM or the Bayesian methods. For the Mod Apte split data set, however, the precision alone is not a sufficient measure, since the data is multi-labeled, then we have also considered the recall measure given by [18]

$$Recall = \frac{TP}{TP + FN} \tag{25}$$

where FN is the number of false negatives. The precision and the recall are then combined using the break-even point defined by the micro or macro averaging given by [18,21]

$$Micro = \frac{1}{N} \sum_{j=1}^{M} n_j \frac{TP}{TP + FP} \tag{26}$$

$$Macro = \frac{1}{M} \sum_{j=1}^{M} \frac{TP}{TP + FP} \tag{27}$$

where M is the number of document classes, N is the total number of documents and n_j the number of documents in class j. Table 2 shows the classification results for the Mod Apte split data set[4].

4.2 Image Databases Categorization Using Bag-of-Visual Words Representations

Images categorization can be based on global and/or local features and is an important step for different problems such as image recommendation [22]. Recently, local feature-based approaches have shown excellent results. One of these methods, that we will follow in our experiments, is the *bag of keypoints* approach [23] originated from an analogy to learning methods in the case of text categorization. After detecting keypoints (or local interest points) in the images using

[4] As mentioned in [18], there is only a single standard training/test set split for this data set, thus the standard deviation is not given.

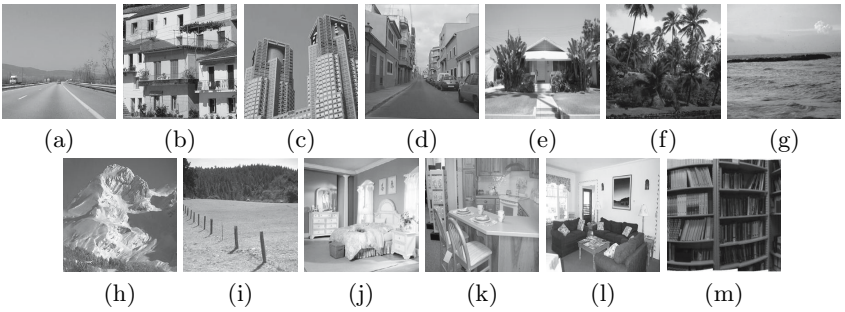


Fig. 1. Sample images from each group. (a) Highway, (b) Inside of cities, (c) Tall buildings, (d) Streets, (e) Suburb residence, (f) Forest, (g) Coast, (h) Mountain, (i) Open country, (j) Bedroom, (k) Kitchen, (l) Livingroom, (m) Office

one the various existing detectors [24], an important step in this approach is the extraction of local descriptors that should be invariant to images transformations, occlusions and lighting variations [23]. keypoints are then grouped into a number of homogenous clusters V, using a clustering or vector quantization algorithm such as K-means, according to the similarity of their descriptors. Each cluster center is then treated as a visual word, and we obtain a vocabulary of V visual words describing all possible local image patterns. Having this vocabulary in hand, each image can be represented as a V-dimensional vector containing the proportion of each visual word in that image. The resulted vector can be used then for the categorization task.

The performance of our infinite mixture model was evaluated on a database containing 13 categories of natural scenes [25,26]: highway (260 images), inside of cities (308 images), tall buildings (356 images), streets (292 images), suburb residence (241 images), forest (328 images), coast (360 images), mountain (374 images), open country (410 images), bedroom (174 images), kitchen (151 images), livingroom (289 images), and office (216 images). Figure 1 shows examples of these images which have an average size of approximately 250×300 pixels. The keypoints were detected using the Harris affine detector [24]. Then, we have used Scale Invariant Feature Transform (SIFT), which performs better than the majority of the existing descriptors [27], computed on detected keypoints of all images and giving 128-dimensional vector for each keypoint. Moreover, SIFT vectors, extracted from 650 training images from all categories (50 images from each category), were clustered using the K-Means algorithm providing 150 visual-words. Each image in the database was then represented by a 150-dimensional vector of proportions.

Our categorization approach is based on a classifier. The inputs to the classifier are the 150-dimensional vectors extracted from the different database classes. These vectors are separated into the unknown or test set of vectors, whose class is unknown, and the training set of vectors (we take randomly 100 vectors for training from each class), whose class is known. The training set is necessary to adapt the classifier to each possible class before the unknown set is submitted

Table 3. Loglikelihoods (average and standard deviation over 10 runs) of the training data in the different classes

	Infinite mixture	Finite mixture + HSEM [8]	Finite mixtue + Bayesian [10]
Highway	-705.74± 2.32	-731.53± 2.43	-723.28± 2.40
Inside	-884.39± 2.17	-890.43± 2.81	-886.28± 2.27
Tall buildings	-598.87± 2.91	-639.81± 3.25	-619.43± 3.03
Streets	-451.29± 1.13	-481.20± 2.17	-469.28± 1.95
Suburb	-728.79± 4.33	-774.26± 5.01	-763.35± 4.58
Forest	-841.55± 3.84	-880.08± 4.23	-871.02± 3.99
Coast	-923.61± 3.60	-953.32± 3.89	-942.86± 3.52
Mountain	-539.72± 2.94	-568.11± 3.12	-559.98± 3.04
Open country	-956.28± 2.89	-997.09± 2.99	-978.28± 2.45
Bedroom	-578.74± 2.26	-592.31± 2.29	-582.06± 2.11
Kitchen	-611.20± 1.65	-642.17± 2.03	-627.19± 1.87
Livingroom	-686.94± 1.45	-703.21± 4.14	-695.62± 3.80
Office	-616.34± 1.21	-645.57±2.32	-636.38±2.14

to the classifier. Then, we apply our algorithm, presented in Section 3, to the training vectors in each class. After this stage, each class in the database is represented by a generalized Dirichlet mixture. Finally, in the classification stage each unknown image is assigned to the class increasing more its loglikelihood.

Table 3 shows the loglikelihoods of the training data in the different classes when using the model in this paper, the finite mixture model proposed in [8] and the Bayesian approach described in [10]. In the reported results, the values of the loglikelihoods are divided by the number of vectors in each training class. The table shows the clear dominance of the infinite model. The results can be explained by the fact that the infinite model outperforms by its ability to incorporate uncertainties related to the selection of the correct number of clusters. We determined the average confusion matrices reported by the generalized Dirichlet and the Gaussian mixtures by running the estimation algorithms 10 times with varying random selection of the training and test sets. The average classification accuracies were 74.21% (634 misclassified images) and 63.68% (893 misclassified images) when we used generalized Dirichlet and Gaussian mixtures, respectively. The fact that the generalized Dirichlet outperforms the Gaussian is actually an expected result, since it is well known that the Dirichlet distribution is an excellent choice to model normalized histograms [2,7]. According to the confusion matrices, an important part of the misclassification errors occurs among the categories: bedroom, livingroom, kitchen and office which is the same conclusion reached in [25].

5 Conclusion

We have described and illustrated a Bayesian nonparametric approach based on infinite generalized Dirichlet mixtures. We have shown that the problem of determining the number of clusters can be avoided by using infinite mixtures

which model the structure of the data well. Indeed, the resulting optimal clustering is obtained by averaging over all number of clusters of different possible models. The inference of the infinite generalized Dirichlet mixture was implemented through Markov chain Monte Carlo methods. The experimental results have shown that our infinite model offer excellent modeling capabilities.

Acknowledgment

The completion of this research was made possible thanks to the Natural Sciences and Engineering Research Council of Canada (NSERC), and a NATEQ Nouveaux Chercheurs Grant.

References

1. McLachlan, G.J., Peel, D.: Finite Mixture Models. Wiley, New York (2000)
2. Bouguila, N., Ziou, D.: Unsupervised Selection of a Finite Dirichlet Mixture Model: An MML-Based Approach. IEEE Transactions on Knowledge and Data Engineering 18(8), 993–1009 (2006)
3. Ferguson, T.S.: Bayesian Density Estimation by Mixtures of Normal Distributions. In: Rizvi, H., Rustagi, J. (eds.) Recent Advances in Statistics, pp. 287–302. Academic Press, New York (1983)
4. Escobar, M.D., West, M.: Bayesian Density Estimation and Inference Using Mixtures. Journal of the American Statistical Association 90(430), 577–588 (1995)
5. MacEachern, S.N., Müller, P.: Estimating Mixture of Dirichlet Process Models. Journal of Computational and Graphical Statistics 7, 223–238 (1998)
6. Neal, R.M.: Markov Chain Sampling Methods for Dirichlet Process Mixture Models. Journal of Computational and Graphical Statistics 9, 249–265 (2000)
7. Bouguila, N., Ziou, D., Vaillancourt, J.: Unsupervised Learning of a Finite Mixture Model Based on the Dirichlet Distribution and its Application. IEEE Transactions on Image Processing 13(11), 1533–1543 (2004)
8. Bouguila, N., Ziou, D.: A Hybrid SEM Algorithm for High-Dimensional Unsupervised Learning Using a Finite Generalized Dirichlet Mixture. IEEE Transactions on Image Processing 15(9), 2657–2668 (2006)
9. Bouguila, N., Ziou, D.: High-Dimensional Unsupervised Selection and Estimation of a Finite Generalized Dirichlet Mixture Model Based on Minimum Message Length. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(10), 1716–1731 (2007)
10. Bouguila, N., Ziou, D., Hammoud, R.I.: A Bayesian Non-Gaussian Mixture Analysis: Application to Eye Modeling. In: Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) (2007)
11. Rasmussen, C.E.: The Infinite Gaussian Mixture Model. In: Advances in Neural Information Processing Systems (NIPS), pp. 554–560 (2000)
12. Antoniak, C.E.: Mixtures of Dirichlet Processes With Applications to Bayesian Nonparametric Problems. The Annals of Statistics 2(6), 1152–1174 (1974)
13. Teh, Y.W., Jordan, M.I., Beal, M.I., Matthew, J., Blei, D.M.: Hierarchical Dirichlet Processes. Journal of the American Statistical Association 101(476), 1566–1581 (2006)

14. Gelman, A.: Prior Distributions for Variance Parameters in Hierachical Models. Bayesian Analysis 1(3), 515–533 (2006)
15. Carlin, B.P., Louis, T.A.: Bayes and Empirical Bayes Methods for Data Analysis, 2nd edn. Chapman & Hall/CRC (2000)
16. Gilks, W.R., Wild, P.: Algorithm AS 287: Adaptive Rejection Sampling from Log-Concave Density Functions. Applied Statistics 42(4), 701–709 (1993)
17. Robert, C.P., Casella, G.: Monte Carlo Statistical Methods. Springer, Heidelberg (1999)
18. Madsen, R.E., Kauchak, D., Elkan, C.: Modeling Word Burstiness Using the Dirichlet Distribution. In: Proc. of the 22^{nd} International Conference on Machine Learning (ICML), pp. 545–552 (2005)
19. Apte, C., Damerau, F.J., Weiss, S.M.: Automated Learning of Decision Rules for Text Categorization. ACM Transactions on Information Systems 12, 233–251 (1994)
20. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
21. Sebastiani, F.: Machine Learning in Automated Text Categorization. ACM Computing Surveys 34(1), 1–47 (2002)
22. Boutemedjet, S., Ziou, D., Bouguila, N.: Unsupervised Feature Selection for Accurate Recommendation of High-Dimensional Image Data. In: Advances in Neural Information Processing Systems (NIPS), pp. 177–184 (2007)
23. Csurka, G., Dance, C.R., Fan, L., Willamowski, J., Bray, C.: Visual Categorization with Bags of Keypoints. In: Workshop on Statistical Learning in Computer Vision, 8th European Conference on Computer Vision (ECCV) (2004)
24. Mikolajczyk, K., Schmid, C.: Scale and Affine Invariant Interest Point Detectors. International Journal of Computer Vision 60(1), 63–86 (2004)
25. Fei-Fei, L., Perona, P.: A Bayesian Hierarchical Model for Learning Natural Scene Categories. In: Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), pp. 524–531 (2005)
26. Oliva, A., Torralba, A.: Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope. International Journal of Computer Vision 42(3), 145–175 (2001)
27. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)

Safe-Level-SMOTE: Safe-Level-Synthetic Minority Over-Sampling TEchnique for Handling the Class Imbalanced Problem

Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap

Department of Mathematics, Faculty of Science, Chulalongkorn University
chumphol@chiangmai.ac.th, Krung.S@chula.ac.th,
lchidcha@chula.ac.th

Abstract. The class imbalanced problem occurs in various disciplines when one of target classes has a tiny number of instances comparing to other classes. A typical classifier normally ignores or neglects to detect a minority class due to the small number of class instances. SMOTE is one of over-sampling techniques that remedies this situation. It generates minority instances within the overlapping regions. However, SMOTE randomly synthesizes the minority instances along a line joining a minority instance and its selected nearest neighbours, ignoring nearby majority instances. Our technique called Safe-Level-SMOTE carefully samples minority instances along the same line with different weight degree, called safe level. The safe level computes by using nearest neighbour minority instances. By synthesizing the minority instances more around larger safe level, we achieve a better accuracy performance than SMOTE and Borderline-SMOTE.

Keywords: Class Imbalanced Problem, Over-sampling, SMOTE, Safe Level.

1 Introduction

A dataset is considered to be imbalanced if one of target classes has a tiny number of instances comparing to other classes. In this paper, we consider only two-class case [5], [17]. The title of a smaller class is a minority class, and that of a bigger class is a majority class. The minority class includes a few positive instances, and the majority class includes a lot of negative instances.

In many real-world domains, analysts encounter many class imbalanced problems, such as the detection of unknown and known network intrusions [8], and the detection of oil spills in satellite radar images [13]. In these domains, standard classifiers need to accurately predict a minority class, which is important and rare, but the usual classifiers seldom predict this minority class.

Strategies for dealing with the class imbalanced problem can be grouped into two categories. One is to re-sample an original dataset [11], [14], [15], either by over-sampling a minority class or under-sampling a majority class until two classes are nearly balanced. The second is to use cost sensitive learning by assigning distinct costs to correctly classified instances or classifications errors [7], [9], [16].

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 475–482, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Table 1. A confusion matrix for a two-class imbalanced problem

	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

The performance of classifiers is customarily evaluated by a confusion matrix as illustrated in Table 1. The rows of the table are the actual class label of an instance, and the columns of the table are the predicted class label of an instance. Typically, the class label of a minority class set as positive, and that of a majority class set as negative. *TP*, *FN*, *FP*, and *TN* are True Positive, False Negative, False Positive, and True Negative, respectively. From Table 1, the six performance measures on classification; *accuracy*, *precision*, *recall*, *F-value*, *TP rate*, and *FP rate*, are defined by formulae in (1)-(6).

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{FN} + \text{FP} + \text{TN}) . \quad (1)$$

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN}) . \quad (2)$$

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP}) . \quad (3)$$

$$\text{F-value} = ((1 + \beta)^2 \cdot \text{Recall} \cdot \text{Precision}) / (\beta^2 \cdot \text{Recall} + \text{Precision}) . \quad (4)$$

$$\text{TP Rate} = \text{TP} / (\text{TP} + \text{FN}) . \quad (5)$$

$$\text{FP Rate} = \text{FP} / (\text{TN} + \text{FP}) . \quad (6)$$

The objective of a classifier needs to aim for high prediction performance on a minority class. Considering the definition of *accuracy*, if most instances in a minority class are misclassified and most instances in a majority class are correctly classified by a classifier, the *accuracy* is still high because the large number of negative instances influences the whole classification result on *accuracy*. Note that *precision* and *recall* are effective for this problem because they evaluate the classification rates by concentrating in a minority class. In addition, *F-value* [3] integrating *recall* and *precision*, is used instead of *recall* and *precision*. Its value is large when both *recall* and *precision* are large. The β parameter corresponding to relative importance of *precision* and *recall* is usually set to 1. Furthermore, ROC curve, The Receiver Operating Characteristic curve, is a standard technique for summarizing classifier performance over a range of tradeoffs between *TP rate*, benefits, and *FP rate*, costs. Moreover, AUC [2], Area under ROC, can also be applied to evaluate the performance of a classifier.

The content of this paper is organized as follows. Section 2 briefly describes related works for handling the class imbalanced problem. Section 3 describes the details of our over-sampling technique, Safe-Level-SMOTE. Section 4 shows the experimental results by comparing Safe-Level-SMOTE to SMOTE and Borderline-SMOTE. Section 5 summarizes the paper and points out our future works.

2 Related Works

Re-sampling is a preprocessing technique which adjusting the distribution of an imbalanced dataset until it is nearly balanced, before feeding it into any classifiers. The simplest re-sampling techniques are a random over-sampling technique [14] and a random under-sampling technique [14]. The former randomly duplicates positive instances into a minority class, while the latter randomly removes negative instances from a majority class. Both techniques are sampling the dataset until the classes are approximately equally represented. However, the random over-sampling technique may cause the overfitting problem [19] because the technique may create the decision regions smaller and more specific. The random under-sampling technique encounters the problem that diminishing some important information of a dataset. For handling these problems, improved re-sampling techniques were studied and are described as follows.

Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W. (2002) designed the State of the Art over-sampling technique, namely SMOTE, Synthetic Minority Over-sampling TEchnique [4]. It over-samples a minority class by taking each positive instance and generating synthetic instances along a line segments joining their k nearest neighbours in the minority class. This causes the selection of a random instance along the line segment between two specific features. The synthetic instances cause a classifier to create larger and less specific decision regions, rather than smaller and more specific regions. More general regions are now learned for positive instances, rather than those being subsumed by negative instances around them. The effect is that decision trees generalize better. However, SMOTE encounters the overgeneralization problem. It blindly generalizes the region of a minority class without considering a majority class. This strategy is particularly problematic in the case of highly skewed class distributions since, in such cases, a minority class is very sparse with respect to a majority class, thus resulting in a greater chance of class mixture.

Han, H., Wang, W., Mao, B. (2005) designed the improvement of SMOTE, namely Borderline-SMOTE [10]. The authors divided positive instances into three regions; noise, borderline, and safe, by considering the number of negative instances on k nearest neighbours. Let n be the number of negative instances among the k nearest neighbours. The three regions are defined by the definitions in Table 2. Borderline-SMOTE uses the same over-sampling technique as SMOTE but it over-samples only the borderline instances of a minority class instead of over-sampling all instances of the class like SMOTE does. Unfortunately, considering two positive instances those n values are equal to k and k-1 for the first and second instances consecutively. These instances are not obviously difference but they are divided into the different regions; noise and borderline. The first instance is declined but the second instance is selected for over-sampling.

Table 2. The definitions of noise, borderline, and safe regions in Borderline-SMOTE

Region	Definition
Noise	$n = k$
Borderline	$\frac{1}{2}k \leq n < k$
Safe	$0 \leq n < \frac{1}{2}k$

3 Safe-Level-SMOTE

Based on SMOTE, Safe-Level-SMOTE, Safe-Level-Synthetic Minority Over-sampling TEchnique, assigns each positive instance its *safe level* before generating synthetic instances. Each synthetic instance is positioned closer to the largest *safe level* so all synthetic instances are generated only in safe regions.

The *safe level* (*sl*) is defined as formula (7). If the *safe level* of an instance is close to 0, the instance is nearly noise. If it is close to *k*, the instance is considered safe. The *safe level ratio* is defined as formula (8). It is used for selecting the safe positions to generate synthetic instances.

$$\text{safe level } (sl) = \text{the number of a positive stances in } k \text{ nearest neighbours .} \quad (7)$$

$$\text{safe level ratio} = sl \text{ of a positive instance} / sl \text{ of a nearest neighbours .} \quad (8)$$

Safe-Level-SMOTE algorithm is showed in Fig. 1. All variables in this algorithm are described as follows. *p* is an instance in the set of all original positive instances *D*. *n* is a selected nearest neighbours of *p*. *s* included in the set of all synthetic positive instances *D'* is a synthetic instance. sl_p and sl_n are *safe level* of *p* and *safe level* of *n* respectively. *sl_ratio* is *safe level ratio*. *numattrs* is the number of attributes. *dif* is the difference between the values of *n* and *p* at the same attribute id. *gap* is a random fraction of *dif*. *p*[*i*], *n*[*i*], and *s*[*i*] are the numeric values of the instances at i^{th} attribute. *p*, *n*, and *s* are vectors. sl_p, sl_n, *sl_ratio*, *numattrs*, *dif*, and *gap* are scalars.

After assigning the *safe level* to *p* and the *safe level* to *n*, the algorithm calculates the *safe level ratio*. There are five cases corresponding to the value of *safe level ratio* showed in the lines 12 to 28 of Fig. 1.

The first case showed in the lines 12 to 14 of Fig. 1. The *safe level ratio* is equal to ∞ and the *safe level* of *p* is equal to 0. It means that both *p* and *n* are noises. If this case occurs, synthetic instance will not be generated because the algorithm does not want to emphasize the important of noise regions.

The second case showed in the lines 17 to 19 of Fig. 1. The *safe level ratio* is equal to ∞ and the *safe level* of *p* is not equal to 0. It means that *n* is noise. If this case occurs, a synthetic instance will be generated far from noise instance *n* by duplicating *p* because the algorithm want to avoid the noise instance *n*.

The third case showed in the lines 20 to 22 of Fig. 1. The *safe level ratio* is equal to 1. It means that the *safe level* of *p* and *n* are the same. If this case occurs, a synthetic instance will be generated along the line between *p* and *n* because *p* is as safe as *n*.

The fourth case showed in the lines 23 to 25 of Fig. 1. The *safe level ratio* is greater than 1. It means that the *safe level* of *p* is greater than that of *n*. If this case occurs, a synthetic instance is positioned closer to *p* because *p* is safer than *n*. The synthetic instance will be generated in the range [0, 1 / *safe level ratio*].

The fifth case showed in the lines 26 to 28 of Fig. 1. The *safe level ratio* is less than 1. It means that the *safe level* of *p* is less than that of *n*. If this case occurs, a synthetic instance is positioned closer to *n* because *n* is safer than *p*. The synthetic instance will be generated in the range [1 - *safe level ratio*, 1].

After each iteration of *for* loop in line 2 finishes, if the first case does not occurs, a synthetic instance *s* will be generated along the specific-ranged line between *p* and *n*, and then *s* will be added to *D'*.

After the algorithm terminates, it returns a set of all synthetic instances *D'*. The algorithm generates |*D*| - *t* synthetic instances where |*D*| is the number of all positive instances in *D*, and *t* is the number of instances that satisfy the first case.

```
Algorithm: Safe-Level-SMOTE
Input: a set of all original positive instances D
Output: a set of all synthetic positive instances D'
 1. D' = ∅
 2. for each positive instance p in D {
 3.    compute k nearest neighbours for p in D and
       randomly select one from the k nearest neighbours, call it n
 4.    sl_p = the number of positive stances in k nearest neighbours for p in D
 5.    sl_n = the number of positive stances in k nearest neighbours for n in D
 6.    if (sl_n ≠ 0) {  ; sl is safe level.
 7.       sl_ratio = sl_p / sl_n  ; sl_ratio is safe level ratio.
 8.    }
 9.    else {
10.       sl_ratio = ∞
11.    }
12.    if (sl_ratio = ∞ AND sl_p = 0) {  ; the 1st case
13.       does not generate positive synthetic instance
14.    }
15.    else {
16.       for (atti = 1 to numattrs) {  ; numattrs is the number of attributes.
17.          if (sl_ratio = ∞ AND sl_p ≠ 0) {  ; the 2nd case
18.             gap = 0
19.          }
20.          else if (sl_ratio = 1) {  ; the 3rd case
21.             random a number between 0 and 1, call it gap
22.          }
23.          else if (sl_ratio > 1) {  ; the 4th case
24.             random a number between 0 and 1/sl_ratio, call it gap
25.          }
26.          else if (sl_ratio < 1) {  ; the 5th case
27.             random a number between 1-sl_ratio and 1, call it gap
28.          }
29.          dif = n[atti] - p[atti]
30.          s[atti] = p[atti] + gap·dif
31.       }
32.       D' = D' ∪ {s}
33.    }
34. }
35. return D'
```

Fig. 1. Safe-Level-SMOTE algorithm

4 Experiments

In our experiments, we use four performance measures; *precision*, *recall*, *F-value*, and AUC, for evaluating the performance of three over-sampling techniques; Safe-Level-SMOTE, SMOTE, and Borderline-SMOTE. The value of β in *F-value* is set to 1 and the value of k in all over-sampling techniques are set to 5. The performance measures are evaluated through 10-fold cross-validation. Three classifiers; decision trees C4.5 [18], Naïve Bayes [12], and support vector machines (SVMs) [6], are applied as classifiers in the experiments. We use two quantitative datasets from UCI Repository of Machine Learning Databases [1]; Satimage and Haberman, illustrated in Table 3. The first to last column of the table represents the dataset name, the number of instances, the number of attributes, the number of positive instances, the number of negative instances, and the percent of a minority class, respectively.

The experimental results on the two datasets are illustrated in Fig. 2. The x-axis in these figures represents the over-sampling percent on a minority class. The y-axis in these figures represents the four performance measures; *precision*, *recall*, *F-value*, and AUC, in order from Fig. 2 (a) to Fig. 2 (d). In these figures, ORG, SMOTE, BORD, and SAFE are the label of the original dataset, SMOTE, Borderline-SMOTE, and Safe-Level-SMOTE, respectively.

For Satimage dataset, we select the class label 4 as the minority class and merge the remainder classes as the majority class because we only study the two-class imbalanced problem. The results on *F-value* using decision trees C4.5 are illustrated in Fig. 2 (c). It is apparent that *F-value* is improved when over-sampling percent on the minority class is increased. Moreover, Safe-Level-SMOTE achieved higher *F-value* than SMOTE and Borderline-SMOTE. The results on *recall* using Naïve Bayes are illustrated in Fig. 2 (b). Analyzing the figure, Borderline-SMOTE gains the higher performance on *recall*, while Safe-Level-SMOTE comes second.

For Haberman dataset, the minority class is about one quarter of the whole dataset. The results on *precision* using decision trees C4.5 are illustrated in Fig. 2 (a). The performance of Safe-Level-SMOTE is the best performance on *precision*. The results on AUC using SVMs are illustrated in Fig. 2 (d). Analyzing the figure, Safe-Level-SMOTE and SMOTE show similar performance on AUC. In addition, Borderline-SMOTE shows poor performance on higher percent.

For all experimental results, Safe-Level-SMOTE obviously achieve higher performance on *precision* and *F-value* than SMOTE and Borderline-SMOTE when decision trees C4.5 are applied as classifiers. Borderline-SMOTE only achieve a better performance on *recall* when Naive Bayes are applied as classifiers since the independent assumption on the borderline region is valid. Moreover, the SVMs show no improvement in all over-sampling techniques. Theses causes by the convex regions of all over-sampling techniques are similar. Therefore, the results of hyperplanes in SVMs are indistinguishable.

Table 3. The descriptions of UCI datasets in the experiments

Name	Instance	Attribute	Positive	Negative	% Minority
Satimage	6,435	37	626	5,809	9.73
Haberman	306	4	81	225	26.47

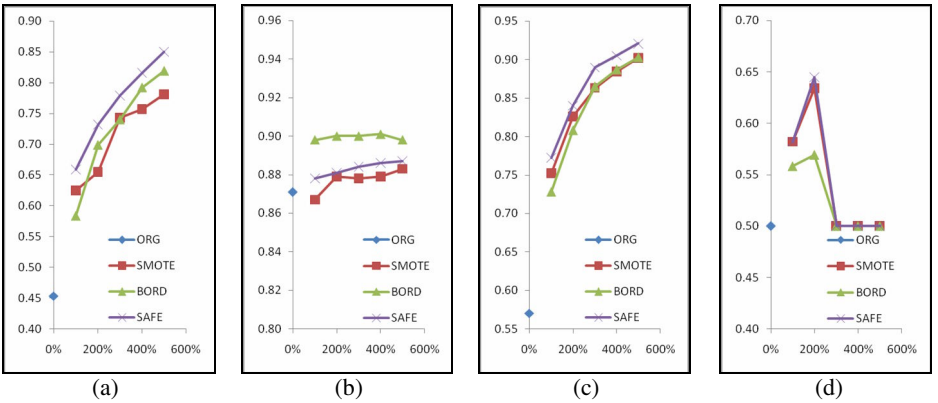


Fig. 2. The experimental results; (a) Precision evaluated by applying C4.5 with Haberman, (b) Recall evaluated by applying Naïve Bayes with Satimage, (c) F-value evaluated by applying C4.5 with Satimage, (d) AUC evaluated by applying SVMs with Haberman

5 Conclusion

The class imbalanced problem has got more attentions among data miners. There are many techniques for handling such problem. However, traditional data mining techniques are still unsatisfactory. We present an efficient technique called Safe-Level-SMOTE to handle this class imbalanced problem.

The experiments show that the performance of Safe-Level-SMOTE evaluated by *precision* and *F-value* are better than that of SMOTE and Borderline-SMOTE when decision trees C4.5 are applied as classifiers. This comes from the fact that Safe-Level-SMOTE carefully over-samples a dataset. Each synthetic instance is generated in safe position by considering the *safe level ratio* of instances. In contrast, SMOTE and Borderline-SMOTE may generate synthetic instances in unsuitable locations, such as overlapping regions and noise regions. We can conclude that synthetic instances generated in safe positions can improve prediction performance of classifiers on the minority class.

Although the experimental results have provided evidence that Safe-Level-SMOTE can be successful classified numeric datasets in the class imbalanced problem, there are several future works left to be studied in this line of research. First, different definitions to assign *safe level* would be valuable. Second, additional methods to classify datasets which have nominal attributes are useful. Third, automatic determination of the amount of synthetic instances generated by Safe-Level-SMOTE should be addressed.

References

1. Blake, C., Merz, C.: UCI Repository of Machine Learning Databases. Department of Information and Computer Sciences, University of California, Irvine, CA, USA (1998), `http://archive.ics.uci.edu/ml/`
2. Bradley, A.: The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms. Pattern Recognition 30(6), 1145–1159 (1997)

3. Buckland, M., Gey, F.: The Relationship between Recall and Precision. Journal of the American Society for Information Science 45(1), 12–19 (1994)
4. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: SMOTE: Synthetic Minority Over-Sampling Technique. Journal of Artificial Intelligence Research 16, 321–357 (2002)
5. Chawla, N., Japkowicz, N., Kolcz, A.: Editorial: Special Issue on Learning from Imbalanced Data Sets. SIGKDD Explorations 6(1), 1–6 (2004)
6. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, Cambridge (2000)
7. Domingos, P.: Metacost: A General Method for Making Classifiers Cost-sensitive. In: The 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 1999), pp. 155–164. ACM Press, San Diego (1999)
8. Fan, W., Miller, M., Stolfo, S., Lee, W., Chan, P.: Using Artificial Anomalies to Detect Unknown and Known Network Intrusions. In: The 1st IEEE International Conference on Data Mining (ICDM 2001), San Jose, CA, USA, pp. 123–130 (2001)
9. Fan, W., Salvatore, S., Zhang, J., Chan, P.: AdaCost: misclassification cost-sensitive boosting. In: The 16th International Conference on Machine Learning (ICML 1999), Bled, Slovenia, pp. 97–105 (1999)
10. Han, H., Wang, W., Mao, B.: Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In: Huang, D.-S., Zhang, X.-P., Huang, G.-B. (eds.) ICIC 2005. LNCS, vol. 3644, pp. 878–887. Springer, Heidelberg (2005)
11. Japkowicz, N.: The Class Imbalance Problem: Significance and Strategies. In: the 2000 International Conference on Artificial Intelligence (IC-AI 2000), Las Vegas, NV, USA, pp. 111–117 (2000)
12. Kamber, M., Han, J.: Data mining: Concepts and Techniques, 2nd edn., pp. 279–327. Morgan-Kaufman, NY, USA (2000)
13. Kubat, M., Holte, R., Matwin, S.: Machine Learning for the Detection of Oil Spills in Satellite Radar Images. Machine Learning 30, 195–215 (1998)
14. Kubat, M., Matwin, S.: Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. In: The 14th International Conference on Machine Learning (ICML 1997), pp. 179–186. Morgan Kaufmann, Nashville (1997)
15. Lewis, D., Catlett, J.: Uncertainty Sampling for Supervised Learning. In: The 11th International Conference on Machine Learning (ICML 1994), pp. 148–156. Morgan Kaufmann, New Brunswick (1994)
16. Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., Brunk, C.: Reducing Misclassification Costs. In: The 11th International Conference on Machine Learning (ICML 1994), pp. 217–225. Morgan Kaufmann, San Francisco (1994)
17. Prati, R., Batista, G., Monard, M.: Class Imbalances versus Class Overlapping: an Analysis of a Learning System Behavior. In: Monroy, R., Arroyo-Figueroa, G., Sucar, L.E., Sossa, H. (eds.) MICAI 2004. LNCS (LNAI), vol. 2972, pp. 312–321. Springer, Heidelberg (2004)
18. Quinlan, J.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo (1992)
19. Tetko, I., Livingstone, D., Luik, A.: Neural network studies. 1. Comparison of Overfitting and Overtraining. Chemical Information and Computer Sciences 35, 826–833 (1995)

Using Highly Expressive Contrast Patterns for Classification - Is It Worthwhile?

Elsa Loekito and James Bailey

NICTA Victoria Laboratory
Department of Computer Science and Software Engineering
University of Melbourne, Australia
{eloekito,jbailey}@csse.unimelb.edu.au

Abstract. Classification is an important task in data mining. Contrast patterns, such as emerging patterns, have been shown to be powerful for building classifiers, but they rarely exist in sparse data. Recently proposed disjunctive emerging patterns are highly expressive, and can potentially overcome this limitation. Simple contrast patterns only allow simple conjunctions, whereas disjunctive patterns additionally allow expressions of disjunctions. This paper investigates whether expressive contrasts are beneficial for classification. We adopt a statistical methodology for eliminating noisy patterns. Our experiments identify circumstances where expressive patterns can improve over previous contrast pattern based classifiers. We also present some guidelines for i) using expressive patterns based on the nature of the given data, ii) how to choose between the different types of contrast patterns for building a classifier.

Keywords: Expressive contrasts, emerging patterns, disjunctive emerging patterns, classification, quantitative association rules.

1 Introduction

Classification is a well studied area in data mining. Contrast patterns [1,2] capture strong differences between classes, and useful for building accurate classifiers. Existing pattern-based classifiers consider simple contrasts, such as emerging patterns [3], which are conjunctions of attribute values. A highly expressive class of contrast, namely disjunctive emerging patterns [4], allows disjunctions as well as conjunctions of attribute values. Their use for classification is an open question though, which we aim to answer in this paper.

Expressive contrasts can potentially overcome some of the limitations of simple contrasts. E.g. the following disjunctive pattern from the *income* [5] data set: $[age \in [30..39] \wedge (industry =$ 'manufacturing' $\vee$ 'transportation')] differentiates males from females, being true for more than 10% of the males but not true for any female. If the two industries were considered individually, the non-disjunctive combination $[age \in [30..39] \wedge industry =$ 'manufacturing'], would be true for far fewer males, thus, a weaker contrast. This issue often arises when the data is sparse, or lacking in data instances. Despite their low frequency, rare contrasts can be useful for classification, but they are often not identified.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 483–490, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Since emerging patterns assume discrete data, the rarity of contrasts can also result from the data discretisation used, when the input data set has a continuous-valued domain. We call this problem the *resolution problem*. In coarsely discretised data, patterns may be lacking class-distinguishing ability, but in a finely discretised data, patterns may be lacking frequencies (or support). Expressive patterns provide a solution to this problem, by allowing several discrete attribute-values to be combined into a disjunction.

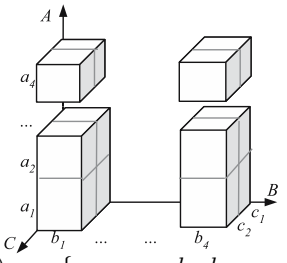
Expressive contrasts may help remedy the above-mentioned situations, but they may also have limitations: i) an increased number of patterns become available, ii) more patterns may be noisy. E.g., $[(age \in [20..24] \vee [40..44]) \wedge industry =$ 'manufacturing'] is a valid disjunctive pattern, but the two age groups, [20..24] and [40..44], may be irrelevant. Such irrelevance within a pattern may, in turn, cause misclassification. To address this issue, we propose a method for statistically testing the significance of disjunctive patterns.

This paper investigates the advantages and disadvantages of using highly expressive contrasts, instead of simple contrasts, for classification. We aim to answer the following questions: i) When should disjunctions be allowed in contrast patterns for building a classifier? ii) Which types of contrast patterns are most suitable for various data characteristics? Our contributions are three-fold:

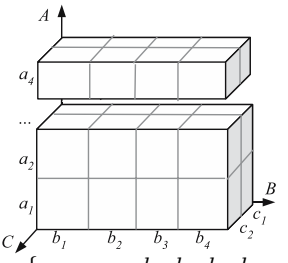
- We propose a classifier model based on disjunctive emerging patterns [4]. To eliminate noise, we use a statistical significance method, similar to that used in [6], which is based on the Fisher's Exact Test. To test the significance of each element in a pattern, we extend the testing methodology by using the *negative representation* of the pattern, which is a conjunction of the negated attribute values. The use of statistical tests on negative conjunctions has not been previously studied.
- We present experimental results using several real [5] data sets, to study the accuracy of our classifier. We use an existing contrast pattern based classifier [1] as a baseline. It shows that the disjunctive classifier is superior for sparse data, and as good as the baseline for dense data. Moreover, data discretisation or data sparsity has low influence on the classification accuracy when expressive contrasts are used.
- Based on our findings, we present a series of recommendations for practitioners, which answer the two questions posed earlier, regarding when disjunctions should be allowed in contrast patterns, and which types of contrasts are most suitable for classifying data with particular characteristics.

2 Contrast Pattern Definitions

A dataset D is defined upon a set of k attributes (also referred as dimensions) $\{A_1, A_2, \ldots, A_k\}$. For every attribute A_i, the domain of its values (or items) is denoted by $dom(A_i)$. Let I be the aggregate of the domains across all the attributes, i.e. $I = \bigcup_{i=1}^{k} dom(A_i)$. An *itemset* is a subset of I. Let P and Q be two itemsets. We say P *contains* Q if Q is a subset of P, $Q \subseteq P$, and P is a superset of Q. A *dataset* is a collection of transactions, each transaction T is a



(a) $x = \{a_1, a_2, a_4, b_1, b_4, c_1, c_2\}$



(b) $y = \{a_1, a_2, a_4, b_1, b_2, b_3, b_4, c_1, c_2\}$

Fig. 1. Geometric representations of disjunctive patterns

set of attribute-values, i.e. $T \subset I$. The number of transactions in D is denoted by $|D|$. The *support* of an itemset P in dataset D, denoted by $support(P, D)$, is the transactions in D which contain P, divided by $|D|$ ($0 \leq support(P, D) \leq 1$).

Assume two classes in dataset D, namely D_p (the positive class) and D_n (the negative class). The support ratio of an itemset between two classes, termed as *growth rate (gr)*: $gr(P, D_p, D_n) = \frac{support(P,D_p)}{support(P,D_n)}$. Each itemset is associated with a discriminating power (or *contrast strength*): $strength(P, D_p, D_n) = support(P, D_p) * \frac{gr(P,D_p,D_n)}{1+gr(P,D_p,D_n)}$. Given support thresholds α and β, an **Emerging Pattern (EP)** [3] is a simple contrast pattern, defined as an itemset P, s.t. $support(P, D_n) \leq \beta$ (i.e. infrequent in D_n), and $support(P, D_p) \geq \alpha$ (i.e. frequent in D_p). Moreover, P is a **minimal emerging pattern** if it does not contain other emerging patterns. A **Jumping Emerging Pattern (JEP)** is an EP which has an infinite growth rate. In the remainder of this paper we use the term *pattern* to refer to an emerging pattern. The *support* of a pattern refers to its support in the positive class.

A **Disjunctive Emerging Pattern (DEP)** is an itemset P which contains one or more items from the domain of every attribute, and satifies two support constraints: i) $support(P, D_p) \geq \alpha$, and ii) $support(P, D_n) \leq \beta$. E.g. Given a dataset with three attribute domains $\{a_1, a_2, a_3, a_4\}$, $\{b_1, b_2, b_3, b_4\}$, $\{c_1, c_2, c_3, c_4\}$, and $x = \{a_1, a_2, a_4, b_1, b_4, c_1, c_2\}$ is a DEP. DEPs express contrasts as conjunctions of disjunctions (CNF), where disjunctions are only allowed between items within attributes. The boolean function that x represents, denoted $f(x)$, is $(a_1 \vee a_2 \vee a_4) \wedge (b_1 \vee b_4) \wedge (c_1 \vee c_2)$. The dataset projection into multi-dimensional space considers x as a subspace (see Fig. 1a). Thus, we can calculate *support* by counting the transactions which are subsets of x.

For an attribute with an ordered domain, a set of adjacent items within the same dimension (or attribute) is called a *contiguous* itemset. We call a set of non-occuring items between two adjacent items (within the same dimension) as a *gap*. If the gap in each dimension of x is no larger than a given minimum threshold g, then we say that x is a *g-contiguous* itemset, where $0 \leq g \leq k - 2$, k is the number of domain items for that attribute. Moreover, a disjunctive pattern is a **g-contiguous pattern** if it does not contain non g-contiguous itemsets in any of its dimensions. Consider Fig. 1, x is g-contiguous for $g \geq 2$, and y is g-contiguous for $g \geq 1$.

3 Classification by Significant Expressive Contrast Patterns

For the purpose of our study, we use the existing JEP-classifier framework [2] as a baseline, which is highly accurate for dense and large datasets. It is based on minimal JEPs, which are considered the most powerful JEPs for classification, since their supports are largest. To adapt the framework, our classifier uses maximal disjunctive patterns which have infinite growth rate. Given a test instance T, all patterns which contain T can be found from each class. Based on its distinguishing class frequencies, a JEP favors D_p over D_n. Each pattern which occurs in T makes a *contribution* to classify T as an instance of D_p, based on its *support*. The JEP classifier then chooses the class which has the highest total contribution to be the winner.

Since disjunctive patterns are relatively longer (i.e. contain more items) than the simple patterns, intuitively not every item makes an equally-high contribution to the contrast strength of a pattern. Thus, we propose two levels of significance testing: i) external significance: tests whether the pattern is highly associated with the class, ii) internal significance: tests whether each element in a pattern makes a significant contribution in the pattern's strength.

3.1 Statistical Fisher Exact Test and Externally Significant Patterns

Work in [6] showed that the Fisher Exact Test (FET) is useful for finding statistically significant association rules, which makes it potentially useful for contrast patterns as well. To test the significance of a pattern P, FET uses a 2x2 contingency table containing the support of P and its complemented support in each class (shown in Table 1). The test returns a p-value, which is a probability that the *null-hypothesis* should be accepted, i.e. there is no significant association between the pattern and the class. If the p-value is below the *significanceLevel* (typically 0.05), we reject the hypothesis and say P is **externally significant**. Given a contingency table $[a, b; c, d]$, and $n = a + b + c + d$. The p-value is computed by:

$$p([a,b;c,d]) = \sum_{i=0}^{min(b,c)} \frac{(a+b)!(c+d)!(a+c)!(b+d)!}{n!(a+i)!(b-i)!(c-i)!(d+i)!} \quad (1)$$

3.2 Internally Significant Disjunctive Emerging Patterns

The testing methodology for significant association rules [6] tests whether the inclusion of each condition significantly contributes to the rule's association.

Table 1. The contingency table for testing the significance of association between pattern P and class C

D	P	$\neg P$
C	$a = support(P, C)$	$b = support(\neg P, C)$
$\neg C$	$c = support(P, \neg C)$	$d = support(\neg P, \neg C)$

However, it was originally fashioned for purely conjunctive rules. To adapt the method for our needs, we use a *negative representation* of a disjunctive pattern, which is a pure conjunction of negated items. A pattern is significant if each of the negated items makes a significant contribution. This differs from previous work on significant association rules, which consider conjunctions of positive items, instead of negative items. E.g. The NNF (Negative Normal Form) representation of a disjunctive pattern x in Fig. 1a, denoted $f_N(x)$, is the conjunction of the non-occurring items: $f_N(x) = (\neg a_3) \wedge (\neg b_2 \wedge \neg b_3) \wedge (\neg c_3 \wedge \neg c_4)$.

Given ordered attribute domains, a disjunctive pattern can be projected to a subspace, possibly with some holes in it (correspond to *gaps*). Small holes may not be worth retaining if they contain very few data instances from the positive class. On the other hand, big holes may be necessary if they contain many data instances from the negative class. A gap is a *significant gap* if it passes the internal significance test. We call the generalisation of a pattern that is obtained by filling-in a gap as the *gap-filled generalisation*. A gap is maximal if it is not a subset of another gap. If all maximal gaps in a pattern are significant, then we say that the pattern is **internally significant**.

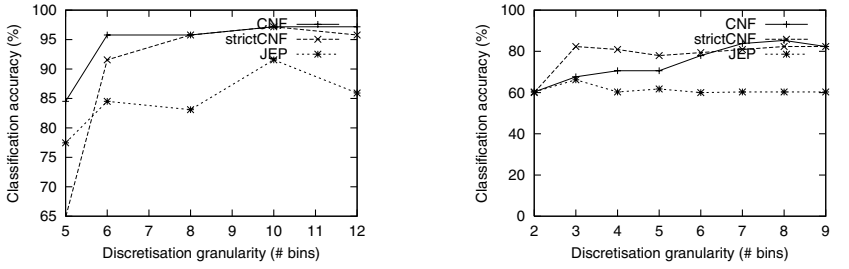
E.g. Reconsider pattern $x = \{a_1, a_2, a_4, b_1, b_4, c_1, c_2\}$. It contains three maximal gaps: $\neg\{a_3\}$, $\neg\{b_2, b_3\}$, $\neg\{c_3, c_4\}$. These correspond to the negative representation of x. The significance of a gap $\neg z$ is calculated between x and its generalisation (by inverting $\neg z$ to z). Let $z = \{b_2, b_3\}$. Let y be the gap-filled generalisation of x s.t. $y = x \cup \{b_2, b_3\} = \{a_1, a_2, a_4, b_1, b_2, b_3, b_4, c_1, c_2\}$. We can calculate the p-value using Eq. 1 and the contingency table in Table 1, by letting $P = \neg z = \neg\{b_2, b_3\}$, $C = D_p|y$, and $\neg C = D_n|y$, where $D_p|y$ (resp. $D_n|y$) refers to transactions in D_p (resp. D_n) which support y. A low p-value indicates the significance of gap $\neg\{b_2, b_3\}$ in x.

3.3 Classification by Significant Disjunctive Emerging Patterns

Our classifier is built based on the maximal disjunctive patterns which have an infinite growth rate. Using only those patterns, however, may overfit the training data. In real situations, there may be training instances which have significant association with the class, but are overlooked, due to the strict *infinite growth rate* constraint. To eliminate this problem, our classifier allows some limited constraint violation by filling-in the insignificant gaps, based on two criteria: i) the gap is not significant in the original pattern, and ii) the resulting gap-filled pattern is externally significant. Thus, all patterns which are used by the classifier are externally and internally significant. We refer to such patterns as **significant disjunctive patterns**.

4 Experimental Results and Discussion

In this section, we study the performance of our classifier described in Section 3.3, based on significant disjunctive patterns, which we call *CNF-Classifier*. We will compare its classification performance against *strictCNF-Classifier*, which is also



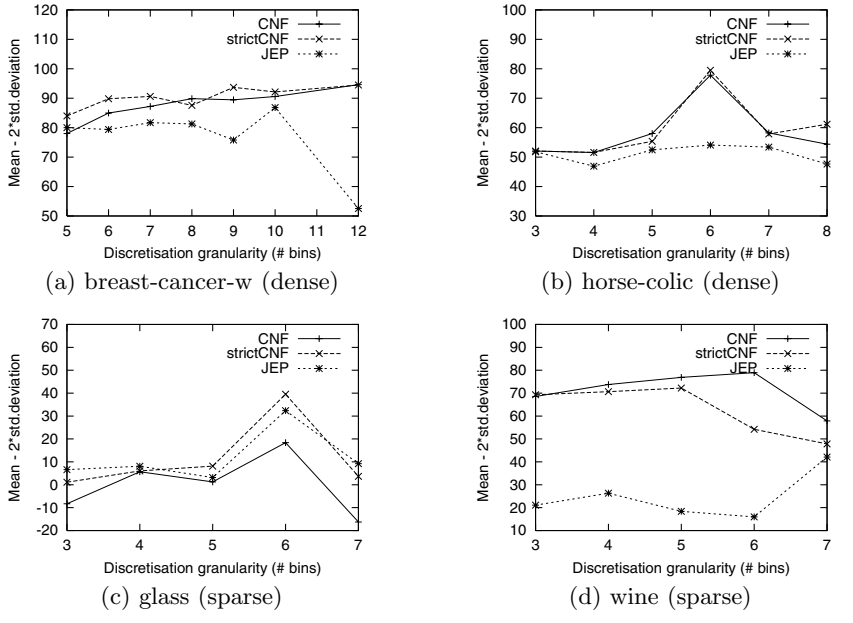
(a) breast-cancer-w (dense) (b) horse-colic (dense)

Fig. 2. Comparison of classification accuracy w.r.t. data discretisation's granularity

based on significant disjunctive patterns, but does not employ the significance testing (strictly imposing the support constraints on the patterns). As a baseline, we also use the Jumping Emerging Pattern Classifier (*JEPC*) [2]. The accuracy is based on 10-fold stratified cross validation. We use four data sets [5], which contain continuous attributes, and categorise them by their sparsity/density. The first two data sets are dense, namely *breast-cancer-w* and *horse-colic*, which contain two classes. The other data sets, *wine* and *glass*, contain multiple classes and are considered sparser. The *glass* data set is greatly imbalanced and extremely sparse, having 7 classes with only a few instances in each class.

Performance comparison with respect to discretisation granularity: In this experiment, we vary the number of bins (or discretised intervals) when discretising each data set using equal-density discretisation. Fig. 2 shows the classification accuracies from two data sets. In the *breast-cancer-w* data set, it is shown that the CNF-Classifier has the highest accuracy for all scenarios. Given finer granularties (i.e. more bins), strictCNF-Classifier and CNF-Classifier are able to outperform JEPC by 12% accuracy. In the *horse-colic* data set, the CNF-Classifier is more accurate than the strictCNF-Classifier when 6 or more bins are used, but it is less accurate otherwise. It shows that the significance test is useful when the data is finely discretised. The JEPC has the lowest accuracy in this data set.

Sensitivity of classification with respect to the support constraint: We now compare the sensitivity of the classifier w.r.t. to the minimum support of the contrast patterns. Fig. 3 shows the lower bound of the accuracy for various support thresholds, which is computed as *(mean - 2 st.dev)*, for each discretisation granularity. In the dense data sets, JEPC has the lowest lower bound and its accuracy greatly varies across the discretisation granularities. When a 12-bin discretisation was used for the *breast-cancer-w* data set, the JEPC has a mean accuracy of 79%, which indicates its large deviation or sensitivity w.r.t. the support constraint. (the relevant figure is not included in this paper, due to space limitation). The other classifiers, on the other hand, have mean accuracies of 96% and 99%, showing their low sensitivity w.r.t. the support constraint.



(a) breast-cancer-w (dense) (b) horse-colic (dense)

(c) glass (sparse) (d) wine (sparse)

Fig. 3. Comparison of mean - 2 st.dev of the classification accuracy (over various minimum support of the patterns) w.r.t. the discretisation granularities

In the sparse *glass* data set, the strictCNF-Classifier has a similar performance to JEPC, whereas the CNF-Classifier has a high sensitivity w.r.t the minimum support threshold and the data discretisation. In the less sparse *wine* data set, CNF-Classifier has the highest lower bound accuracy, and JEPC has the lowest lower bound.

Practical recommendations for users: Answering the questions posed at the beginning of this paper, we now present our recommendations:

When should disjunctions be allowed in contrast patterns for building a classifier? Disjunctions should be allowed in contrast patterns when the data is sparse, that is when the classes are imbalanced, or when the data is finely discretised, e.g. 8 bins or finer.

Which types of contrast patterns are most suitable for various data characteristics? When the data is sparse, expressive contrasts are more appropriate than simple contrasts. The significance test should be performed, except when the data is greatly imbalanced. Simple contrasts are useful for dense and coarsely discretised data sets.

5 Related Work

A contrast pattern is similar to a highly confident class association rule [7]. More expressive association rules have been studied [8], but they allow DNF

(disjunction of conjunctions) rules, instead of CNF, which is the kind of rules considered in this paper. Our significance testing methodology could be extended for disjunctive association rules. Previous work on significant association rules [9,6] only considers conjunctive rules. In an ordered domain, contiguous disjunctive patterns correspond to quantitative association rules [10], which are conjunctions of intervals of ordered values, however gaps are disallowed in a quantitative association rule. The negative representation of a disjunctive pattern in this paper is similar to a negative association rule [11], but the rule's antecedent contains only negative items, and the consequent contains a class label.

6 Conclusion and Future Work

In this paper, we investigated the advantages and disadvantages of using expressive (in the form of CNF combinations) contrast patterns in classification. We proposed a statistical testing for finding significant CNF patterns, which can also be adopted for disjunctive association rules or negative association rules. As our results suggest, expressive forms of patterns can be beneficial for classification, being less sensitive to the data sparsity. For future research, we would like to investigate their use in other types of classifiers and other data mining tasks.

References

1. Dong, G., Zhang, X., Wong, L., Li, J.: CAEP: Classification by aggregating emerging patterns. Discovery Science, 30–42 (1999)
2. Dong, G., Li, J., Ramamohanarao, K.: Making use of the most expressive jumping emerging patterns for classification. KAIS 3, 131–145 (2001)
3. Dong, G.: Li, J.: Efficient mining of emerging patterns: Discovering trends and differences. In: Proc. of KDD 1999, pp. 43–52 (1999)
4. Loekito, E., Bailey, J.: Fast mining of high dimensional expressive contrast patterns using ZBDDs. In: Proc. of KDD 2006, pp. 307–316 (2006)
5. Murphy, P.M., Aha, D.W.: UCI repository of machine learning databases. University of California, Dept. of Information and Computer Science, Irvine, CA
6. Verhein, F., Chawla, S.: Using significant, positively associated and relatively class correlated rules for associative classification of imbalanced datasets. In: Proc. of ICDM 2007, pp. 679–684 (2007)
7. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: Proc. of KDD 1998, pp. 80–86 (1998)
8. Navati, A.A., Chitrapura, K.I., Joshi, S., Krishnapuram, R.: Mining generalised disjunctive association rules. In: Proc. of CIKM 2001, pp. 482–489 (2001)
9. Webb, G.I.: Discovering significant rules. In: Proc. of KDD 2006, pp. 434–443 (2006)
10. Srikant, R., Agrawal, R.: Mining quantitative association rules in large relational tables. In: Proc. of SIGMOD 1996 (1996)
11. Wang, H., Zhang, X., Chen, G.: Mining a complete set of both positive and negative association rules from large databases. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS, vol. 5012, pp. 777–784. Springer, Heidelberg (2008)

Arif Index for Predicting the Classification Accuracy of Features and Its Application in Heart Beat Classification Problem

M. Arif[1], Fayyaz A. Afsar[2], M.U. Akram[2], and A. Fida[3]

[1] Department of Electrical Engineering, Air University, Islamabad, Pakistan
arif@mail.au.edu.pk
[2] Department of Computer and Information Sciences, PIEAS, Islamabad, Pakistan
[3] Department of Electrical Engineering, COMSATS Institute of Information Technology, Pakistan

Abstract. In this paper, Arif Index is proposed that can be used to assess the discrimination power of features in pattern classification problems. Discrimination power of features play an important role in the classification accuracy of a particular classifier applied to the pattern classification problem. Optimizing the performance of a classifier requires a prior knowledge of maximum achievable accuracy in pattern classification using a particular set of features. Moreover, it is also desirable to know that this set of features is separable by a decision boundary of any arbitrary complexity or not. Proposed index varies linearly with the overlap of features of different classes in the feature space and hence can be used in predicting the classification accuracy of the features that can be achieved by some optimal classifier. Using synthetic data, it is shown that the predicted accuracy and Arif index are very strongly correlated with each other ($R^2 = 0.99$). Implementation of the index is simple and time efficient. Index was tested on Arrhythmia beat classification problem and predicted accuracy was found to be in consistent with the reported results.

Keywords: Clustering, Pattern Classification, Features, Nearest Neighbor Search, Classification Accuracy.

1 Introduction

In pattern classification problem, classification accuracy depends on proper selection of features that can discriminate different classes and design of a good classifier. Design of a classifier includes ability of classifier to approximate decision boundary of arbitrary complexity among different classes in the feature space and generalization power of the classifier. Discrimination power of features decides maximum possible classification accuracy achievable by any classifier. A prior knowledge of maximum achievable classification accuracy can help a lot in designing appropriate optimal classifier. Moreover, if classes are separable in the feature space by a decision boundary of any arbitrary complexity, then it is possible to achieve maximum classification accuracy. In real life, feature representing different classes can cluster in the feature

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 491–498, 2009.
© Springer-Verlag Berlin Heidelberg 2009

space as scattered multi-modal clusters. A class can be represented by multiple clusters scattered in the feature space. These clusters can be point or line or any arbitrary shaped clusters. Moreover, they can be compact and well separated (separated by large margin) within class or overlapping each other. Overlapping of intra-class clusters has less effect on classification accuracy as compared to overlap among inter-class clusters. Hence, for a good feature classification, clusters in a particular class need not be very compact and well separated in the feature space but the decision boundary among classes should be well separated and non-overlapping. Hence an index is required to assess the discrimination power or quality of the features in the feature space when information of class labels is available. This index should posses following characteristics. It should be sensitive to the amount of overlap of features among different classes that results in the decrease of classification accuracy and strong correlation may exist between index value and amount of overlap or classification accuracy. If classes in the feature space are separable by any decision boundary of arbitrary complexity, it should give a consistent value and the value of index may not vary with the complexity of decision boundary. Index may not vary with the number of clusters per class, shape of the clusters and their intra-class overlap and their location with respect to other classes in the feature space. Such an index can give us a prior knowledge of maximum achievable classification accuracy by any perfect classifier.

Three different kinds of clustering validity indices exist in the literature, namely, external criteria based indices, internal criteria based indices and relative criteria based indices [2]. In internal criteria based indices, clusters generated from a clustering algorithm are evaluated by considering the data itself based on intra-cluster and inter-cluster distances. In these indices, Dunn's Index [3], Alternative Dunn's Index [1], Davies-Boulden Index [4] and Xie and Beni's Index [5] are worth mentioning. In external criteria based indices, user defined partition of data is provided (class labels) that can be compared with the clustering structure revealed by a clustering algorithm. Different external criteria based indices like Rand Index [6], Jaccard Co-efficient [2], Folkes and Mallows [7], Mirkin Index [8] and Adjusted Rand Index [9] are reported in the literature. Density based clustering algorithms can find arbitrary shaped clusters. Many such algorithms are reported in the literature like DBSCAN [11], BRIDGE [12], DBCLASD [13] and DENCLUE [14]. These indices are not useful in evaluating the discrimination power of the features in classification problem. These indices cannot be applied directly to the feature representation of different classes in pattern classification problem by simply assuming classes as clusters. Clustering structure within class is required before applying these indices. A survey of clustering algorithms can be found in [10]. Performance of these indices is very sensitive to the performance of clustering algorithm. Clustering algorithm should be capable of discovering cluster structure of any arbitrary shape. In this paper, an index called Arif index is proposed which does not require any clustering algorithm. This index spreads linearly on the scale of zero and one where zero value shows no overlap among clusters of different classes. This index is independent of number of clusters per class, type of clusters and their location in feature space. We have used Arif Index to assess the discrimination power of the features used in Arrhythmias beat classification problem [15].

2 Description of Proposed Arif Index

Let number of classes is N_C having data points $n_i, i = 1,...,N_c$ and total data points are $N = \sum_{i=1}^{N_C} n_i$. Dimension of the feature space is *d*. Algorithm of Arif index is described as below,

Step 1: Normalize the feature vectors by making their means equal to zero and variances equal to one.

Step 2: initialize a variable Status S_t of size $N \times 1$ with zeroes.

Step 3: For a data point *y* of size $d \times 1$ of j^{th} Class, Find nearest neighbor of y in the rest of classes which are different from j^{th} Class. Let this nearest neighbor is nn_d having distance δ_{y,nn_d} from data point *y*.

Step 4: Find all data points of j^{th} Class whose distance are less than δ_{y,nn_d} from data point *y*. Let number of nearest neighbors of y in j^{th} Class whose distance are less than δ_{y,nn_d} are $nn_s(k), \quad k = 1,2,...,n_j$. Average number of neighbors near to each other is defined as,

$$C(j) = \frac{1}{n_j} \sum_{k=1}^{n_j} nn_s(k) \tag{1}$$

Step 5: If number of nearest neighbors nn_s is greater than a user defined threshold value γ, consider this data points as clustered in the data points of the same class and make the status of the set of data points $\{y \quad nn_s\}$ equals to 1.

Step 6: Run steps 2 to 5 for all the data points in the feature space.

Step 7: Arif index will be defined as follows,

$$AF_{index} = \frac{N - \sum_{i=1}^{N} S_t(i)}{N} \tag{2}$$

Hence Arif index gives the ratio of data points which are not surrounded by data points of its class to the total number of data points. Strength of clustering data points of the same class near a particular data point is controlled by a user defined threshold value γ. The values of γ should be greater than 1. Value of Arif index varies from 0 to 1, where value of 0 means no overlapping and maximum accuracy of 100% is achievable and values of 1 means complete overlap and accuracy depends on the data representation of different classes. In case of two classes with equal number of

representation, for Arif index equals to 1 means 50% accuracy is possible by just assigning label of one class to all the data. $C(j)$ is the average number of nearest neighbor of all the feature vectors of j^{th} class. This value will give the density estimate of the clustering structure of a particular class. Low value of $C(j)$ shows sparse representation of a class in the feature space. Hence, it will also help in better understanding the quality of features representing the pattern classification problem in the feature space.

3 Results and Discussion

Checkerboard data as shown in Figure 1 is used to evaluate the Arif index for separable classes. Furthermore, data of two classes are brought together in steps so that overlapping of both classes increases and at the end they completely overlap each other. K-nearest neighbor is used as a classifier and value of K is set to 5. Half of the data is used for the training and rest of the data is used for the testing purpose. Arif index is calculated on training data and classification accuracy of K-nearest neighbor classifier is calculated on testing data. In Figure 2, scatter plot between Arif index and the classification accuracy is plotted. It can be observed from the Figure 2 that when two classes are separable, the value of Arif index was zero and as overlapping between classes increased, the value of Arif index also increased and the classification accuracy decreased. Arif index approached near to its maximum value of 1 for completely overlapping classes. A very nice agreement between values of Arif index and classification accuracy is observed. A linear trend is very prominent in the scatter plot and straight line is fitted with very high value of $R^2 = 0.99$. For the value of Arif index near to 1, accuracy dropped to almost 50%. This observation is very obvious as we have used equal number of data points for each class in the testing data. If we do not apply any classifier and assign one class label to all the data points of the testing data, we will get the accuracy of 50% without applying any classifier. This shows the usefulness of using Arif index to get an idea about the upper limit of maximum achievable accuracy before applying any classifier to the data.

For multi-class problems, lower bound on the accuracy without applying any classifier will be the percentage of representation of majority class in the data set if we set all the class labels to the class label of majority class. A linear trend can be interpolated between 100% classification accuracy and the lower bound of the classification accuracy. An ideal linear trend can be described as below,

$$Accuracy = 100 - (100 - Accuracy_{lower_bound}) \times Arif\ \ Index \qquad (3)$$

Where $Accuracy_{lower_bound}$ is the lower bound on the accuracy depending on the percentage representation of the majority class. To prove this hypothesis, we have generated data for four separable classes as shown in Figure 3a and slowly move them towards each other until they completely overlap each other as shown in Figure 3b. During this process, we have calculated Arif index and accuracy by applying a K-nearest neighbor classifier and plotted it in Figure 4. It can be observed from Figure 4 that for Arif index equals to 1 (case of completely overlap), accuracy dropped to 25%.

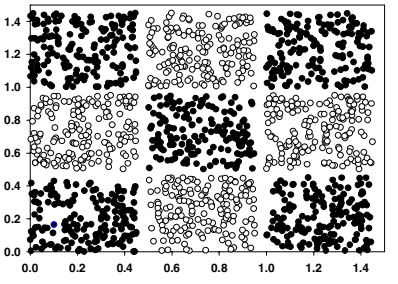


Fig. 1. Checkerboard data of two classes

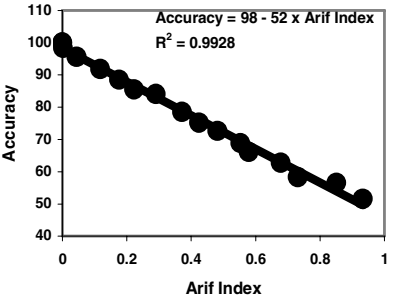


Fig. 2. Plot of Classification Accuracy and Arif Index for all five data sets

A linear trend can be fitted on these data points with very high value of regression coefficient $R^2 = 0.98$. Since there were four classes containing equal number of data points, the classification accuracy at complete overlap will be 25%. This supports our hypothesis and applicability of equation (3). Hence Arif index can be used to predict the quality of features before applying any classifier and value of Arif index can be used to predict maximum achievable classification accuracy that can help in proper tuning or selection of the classifiers.

ECG beat classification, being an integral part of any ECG based automatic decision support system, has been studied by a number of researchers. Different feature extraction methods for beat classification include use of Fourier Transform [16], multi-resolution analysis [17], wavelet transform [18] etc. In our previous work [15], we have used features extracted from two-level wavelet decomposition of an ECG signal from MIT-BIH Arrhythmia Database [19]. It contains two-channel ambulatory ECG recordings from 47 subjects studied by the BIH Arrhythmia Laboratory between 1975 and 1979. ECG recordings were digitized at the sampling rate of 360 Hz with 11-bit resolution. We have used the annotations of the cardiologist originally provided by the database. In this paper, same set of features are used as in [15] to predict the classification accuracy of the beat classification using Arif index. Eleven features extracted from the wavelet decomposition of the ECG beat signal and RR interval are used for classification. Further Principal Component Analysis is used to reduce the

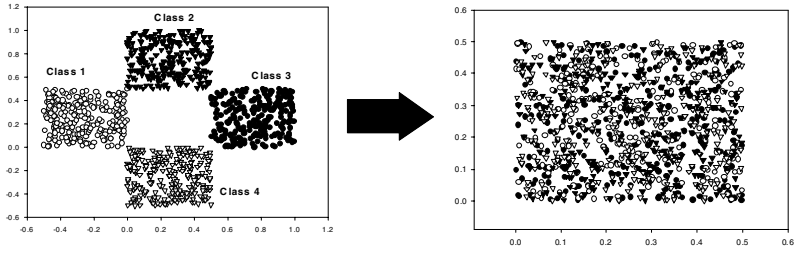


Fig. 3. (a) Set 2: Four class data having four separable square clusters (b) Complete overlap of all four classes

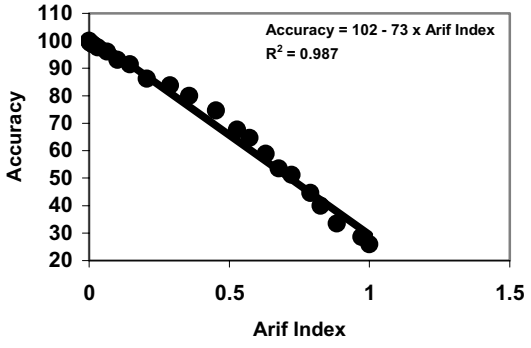


Fig. 4. Plot of Classification Accuracy and Arif Index for Set 6

Table 1. PPV and C values for Set 6

Class	PB	APB	LBBB	N	RBBB	PVC
Data #	2400	1700	4800	7200	4800	2300
PPV (Without PCA)	99.82	99.00	99.24	99.16	99.11	99.95
C	890	550	526	2874	513	337
PPV (With PCA)	99.78	99.17	99.27	98.80	98.85	99.89
C	934	524	645	2656	508	350

dimension of the feature set from eleven to six. For details of the feature extraction, please refer to [15]. Two sets, Set6 and Set9, are constructed from the database. Set6 consists of 23200 beats of six types (Paced Beats (PB), Atrial Premature Beat (APB), Premature Ventricular Contraction (PVC), Normal (N), Left and Right Bundle Branch Blocks (LBBB & RBBB)), whereas Set9 consists of 104690 beats of nine types (PB, APB, PVC, N, LBBB, RBBB, Fusion of paced and Normal beats (P&N), Fusion of Normal and Ventricular beats (N&V) and Ventricular Flutter (VF)). A simple *K*-nearest neighbor classifier has been employed for the classification of different types of beats. Arif index along with *C* were calculated on Set6 and Set9. Classification accuracies were calculated by using half of the data sets for training and rest of the half for testing.

Positive Predictive Values (*PPV*) of each class was calculated as follows,

$$PPV_c = \frac{TP_c}{TP_c + FP_c} \tag{4}$$

Where TP_c and FP_c representing the number of true and false positives for a given class *c*. PPV for Set6 is tabulated in Table1 with the values of C obtained from Arif index as explained in Section 2. Values of *C* represent number of neighbors of a data point of same class within the distance of the data point to the data point of other class. Hence high value of *C* shows denser clustering and low value corresponds to

Table 2. Arif Index, predicted and achieved accuracies for Set 6

	Arif Index	Predicted Accuracy	Achieved Accuracy
Without PCA	0.004	99.7	99.49 [15]
With PCA	0.004	99.7	99.47 [15]

Table 3. PPV and C values for Set 9

Class	VF	PB	APB	N&V	LBBB	N	RBBB	PVC	P&N
Data #	470	3616	2495	774	8067	74716	7247	7058	258
PPV (Without PCA)	80.48	99.66	83.54	77.68	94.46	98.27	98.16	92.6	88.07
C	7	737	73	11	176	1100	424	143	10
PPV (With PCA)	79.51	99.43	82.51	77.08	93.95	98.15	98.00	92.3	78.48
C	7.8	624	62	11	188	1041	379	133	7.8

Table 4. Arif Index, predicted and achieved accuracies for Set 9

	Arif Index	Predicted Accuracy	Reported Accuracy
Without PCA	0.023	99.3	97.04 [20]
With PCA	0.068	98	96.82 [20]

sparse representation of a class. Arif index, predicted accuracy and achieved accuracy are given in Table 2. It can be seen from the table that Arif index for Set6 is low; nearly zero, and predicted accuracies and achieved accuracies match each other. *C* values tabulated in Table 1 are high showing denser clustering and hence PPV values for all six classes with or without PCA are very high, i.e. above 99%. For Set9 with larger number of beat types, results are tabulated in Table 3 and 4. It can be observed from Table 4 that for low values of C (VF, N&V and APB), PPV values are also low as compared to denser classes. For the reduced feature sets (With PCA), values of *C* further reduced and this reduction is reflected in the decrease of PPV values of less denser classes. Arif index, predicted accuracy and achieved accuracy for Set9 are tabulated in Table 4. Arif index is increased a bit as compared to Set6 and correspondingly predicted and achieved accuracies are also dropped slightly. Hence it is proved that Arif index and C values can be used efficiently to predict the classification accuracy and PPV of individual classes.

4 Conclusions

A novel index to assess the classification quality of the feature vectors is proposed in this paper. This index is a model free index and does not require any clustering algorithm. It only uses the information of local neighborhood of the feature vectors to calculate the overlap or region of confusions in the feature space. Results have shown that value of the proposed index does not depend on the shape, location or the structure of the clusters in the feature space. Moreover, values of the Arif index are shown to be strongly correlated with the classification accuracies. Predicted accuracies of different physiological sets are also found to be consistent with the reported accuracies in the literature. Hence this index will be very useful for the pattern classification problems.

References

1. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: On Clustering Validation Techniques. Journal of Intelligent Information Systems 17(2/3), 107–145 (2001)
2. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: Cluster validity methods: Part 1. In: SIGMOD Record, vol. 31(2), pp. 40–45 (2002)
3. Dunn, J.C.: Well Separated Clusters and Optimal Fuzzy Partitions. J. Cybern. 4, 95–104 (1974)
4. Davies, D.L., Bouldin, D.W.: A Cluster Separation Measure. IEEE Transactions on Pattern Analysis and Machine Intelligence 1(2), 224–227 (1979)
5. Xie, X.L., Beni, G.: A Validity Measure for Fuzzy Clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(4), 841–846 (1991)
6. Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association 66, 846–850 (1971)
7. Fowlkes, E., Mallows, C.: A method for comparing two hierarchical clustering. Journal of the American Association 78 (1983)
8. Mirkin, B.G., Cherny, L.B.: On a distance measure between partitions of a finite set. Automation and remote Control 31(5), 91–98 (1970)
9. Hubert, L., Arabie, P.: Comparing partitions. Journal of Classification, 193–218 (1985)
10. Xu, R., Wunsch, D.: Survey of Clustering Algorithms. IEEE Transactions on Neural Networks 16(3), 645–678 (2005)
11. Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of Second International Conference on Knowledge Discovery and Data Mining, pp. 226–231 (1996)
12. Dash, M., Liu, H., Xu, X.: 1+1>2: Merging Distance and Density Based Clustering. In: Proceedings of Seventh International Conference on Database Systems for Advanced Applications, pp. 32–39 (2001)
13. Xu Ester, X., Kriegel, M., Sander, H.-P.: A distribution-based clustering algorithm for mining in large spatial databases. In: Proceedings of 14th International Conference on Data Engineering, pp. 324–331 (1998)
14. Hinneburg, A., Keim, D.A.: An Efficient Approach to Clustering in Large Multimedia Databases with Noise. In: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, pp. 58–65 (1998)
15. Afsar, F.A., Arif, M.: Robust electrocardiogram (ECG) beat classification using discrete wavelet transform. Physiological Measurement 29, 555–570 (2008)
16. Minami, K., Nakajima, H., Toyoshima, T.: Real-time discrimination of ventricular tachyarrhythmia with Fourier-transform neural network. IEEE Transactions on Biomedical Engineering 46(2), 179–185 (1999)
17. Prasad, G.K., Sahambi, J.S.: Classification of ECG arrhythmias using multiresolution analysis and Neural Networks. In: Conference on Convergent Technologies, India (2003)
18. Yu, S.N., Chen, Y.H.: Electrocardiogram beat classification based on wavelet transformation and probabilistic neural network. Pattern Recognition Letters 28(10), 1142–1150 (2007)
19. Mark, R., Moody, G.: MIT-BIH Arrhythmia Database Directory. MIT Press, Cambridge (1988)
20. Usman Akram, M.: Application of Prototype Based Fuzzy Classifiers for ECG based Cardiac Arrhythmia Recognition, BS Thesis, Pakistan Institute of Engineering and Applied Sciences (2008)

UCI++: Improved Support for Algorithm Selection Using Datasetoids

Carlos Soares

LIAAD-INESC Porto LA/Faculdade de Economia
Universidade do Porto, Portugal
csoares@fep.up.pt

Abstract. As companies employ a larger number of models, the problem of algorithm (and parameter) selection is becoming increasingly important. Two approaches to obtain empirical knowledge that is useful for that purpose are empirical studies and metalearning. However, most empirical (meta)knowledge is obtained from a relatively small set of datasets. In this paper, we propose a method to obtain a large number of datasets which is based on a simple transformation of existing datasets, referred to as *datasetoids*. We test our approach on the problem of using metalearning to predict when to prune decision trees. The results show significant improvement when using datasetoids. Additionally, we identify a number of potential anomalies in the generated datasetoids and propose methods to solve them.

1 Introduction

An increasingly important problem for data mining (DM) users is the management of large sets of models [9]. One particular problem is the selection of the best algorithm and corresponding parameters for a particular dataset. When developing hundreds or thousands of models, common trial-and-error approaches are too expensive in terms of computational and human resources. These issues are of particular concern to DM software vendors, who are beginning to integrate functionalities to address them into their tools.[1]

One approach to the problem of algorithm selection is *metalearning* [3]. It consists of applying learning algorithms on *metadata*, i.e., data that describe the performance of learning algorithms on previously processed datasets. Thus, the *metamodel* obtained maps characteristics of those datasets to the performance of the algorithms. Besides providing knowledge that can be useful to understand those algorithms better, it can be used to provide recommendation (e.g., a ranking of those algorithms) for new datasets [4], thus reducing the effort required to obtain satisfactory models for them.

[1] A few examples are SAS Model Manager (http://www.sas.com/technologies/analytics/modelmanager/manager/index.html), Oracle's Mining Activities (http://www.oracle.com/technology/products/bi/odm/pdf/data-mining-11g-datasheet.pdf) and Salford System's modeling automation techonologies (http://salford-systems.com/cart.php).

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 499–506, 2009.
© Springer-Verlag Berlin Heidelberg 2009

One of the most important problems in using metalearning for algorithm selection is the availability of a number of datasets that is sufficient to enable reliable (meta-)induction. The UCI Repository (UCI-R) [1] is the most common source of examples for metalearning and it contains slightly over 100 datasets. Given that each dataset represents one meta-example, most metalearning research is based on approximately 100 examples. This is a small number to ensure that highly reliable models are obtained, particularly in such a complex application such as metalearning. Two common approaches to increase the amount of datasets used for metalearning is the generation of synthetic datasets or the manipulation of existing ones [2,6,7]. In this paper, we propose a new method to generate new datasets from existing ones, referred to as *datasetoids*, by applying very simple manipulations, which addresses the shortcomings of existing methods. We test the new method on datasets from the UCI-R and use the datasetoids obtained to (meta)learn when to use pruning learning in decision trees.

In the next section we summarize related work on the generation of datasets by manipulating existing ones and on metalearning. In Section 3 we describe the method to generate datasetoids. The metalearning application that illustrates the usefulness of datasetoids is given in Section 4. A discussion of potential anomalies of datasetoids and how to deal with them is given in Section 5 and we close with conclusions.

2 Related Work

Manipulation of Datasets. The manipulation of existing datasets to obtain new ones is not a novel idea (e.g., [6]). Typical approaches apply changes separately to independent (e.g., adding redundant attributes) and dependent variables (e.g., adding noise to the target attribute). The metaknowledge that can be obtained from such datasets is focused on a certain aspect of the behavior of algorithms (e.g., resilience to redundant attributes). Although useful, it is too specific to provide much insight on the general performance of algorithms, which is the focus of this paper. What ultimately affects the performance of algorithms is the joint distribution between the dependent variables and the target. The changes that can be forced on the joint distribution of a given dataset are either random or made according to some model. The former case can be reduced to the case of adding noise to the data. In the latter case, the model used in the latter entails a bias, which will, naturally, favor some algorithms relative to others. This restricts the generality of the metaknowledge that can be obtained.

Similar drawbacks apply to methods that generate artificial datasets. However, given that no data is available to start with, the joint distribution must be defined *a priori*. If it is random, the data is mostly useless.Otherwise, some kind of bias is favored.

An important issue when generating datasets with the aim of obtaining general metaknowledge is relevance. The question is whether the generated datasets are representative of the population of real world datasets, or an interesting part of it, at least. Metaknowledge generated from a set of datasets is only useful

in practice if they are representative of real world problems. Two approaches that can be followed to obtain empirical metaknowledge is by carrying out experimental studies or by metalearning, which is the approach followed in this work.

Metalearning for Algorithm Selection. Metalearning can be defined as the use of learning algorithms to obtain knowledge of the behavior of learning algorithms [3]. One particular application of metalearning is to provide models that can predict the (relative) performance of algorithms on new datasets and can, thus, be used to support the user in selecting which algorithm to use. Metalearning for algorithm selection involves three steps: (1) the generation of *metadata*; (2) induction of a metamodel by applying a learning algorithm on the metadata; and (3) application of the metamodel to support the selection of the algorithms to be used in new datasets. These steps are summarized next, but for a more thorough description, the reader is referred to [3].

In this context, metadata are data that describe the (relative) performance of the selected algorithms on a set of datasets, which were already processed with those algorithms. They consist of a set of *meta-examples*, each one representing one dataset. Each meta-example consists of a set of attributes and a target.

The attributes, which are also known as *metafeatures*, are measures that characterize the datasets. These measures represent general properties of the data which are expected to affect the performance of the algorithms. A few examples of commonly used metafeatures are the number of examples, the proportion of symbolic attributes, class entropy and the mean correlation between attributes.

The target of the metalearning problem, or *metatarget*, represents the relative performance of the algorithms on the dataset. Here, it is represented as the algorithm that is expected to perform best, thus transforming the metalearning problem into a classification task

By applying a learning algorithm to the metadata, we obtain a *metamodel* that relates the characteristics of datasets (given by the metafeatures) with the relative performance of the algorithms (metatarget). In this work, standard classification algorithms can be used for metalearning.

The metamodel can then be used to support the data miner in selecting the algorithm to use on a new dataset. To do this, it is rst necessary to compute the metafeatures that are used by the metamodel to generate the prediction.

3 Datasetoids

The method proposed here also creates new datasets by manipulating existing ones. It simply consists of generating a new dataset for each symbolic attribute of a given dataset, obtained by switching the selected attribute with the target variable (Figure 1). We call *datasetoids* to the the newly generated datasets because, although they are sets of data, they probably do not represent a learning application which is interesting in the corresponding domain (e.g., predicting the product purchased, given the customer and the amount purchased).

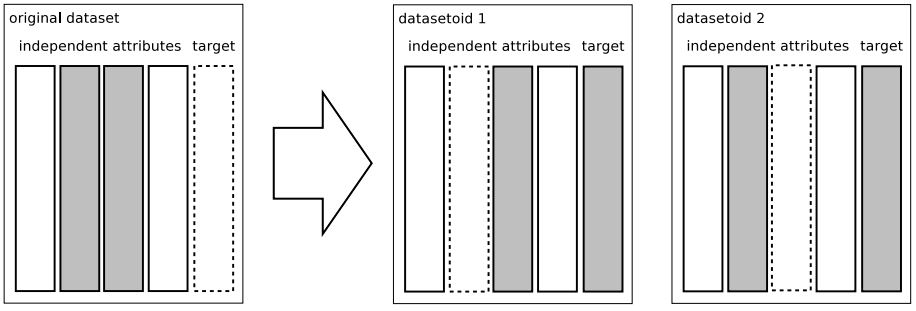


Fig. 1. Datasetoids created from a dataset with two symbolic attributes (gray)

Given that there are approximately 3000 symbolic attributes in the UCI-R datasets, this method enables us to create up to that number of datasetoids. In practice, however, some restrictions apply, as will be discussed below. Still, a considerable number of new datasets can be created. Here, we used 64 datasets from the UCI-R, obtaining a total of 983 datasetoids.

4 Using Datasetoids to Learn When to Prune

Our goal in this paper is to investigate whether it is possible to obtain useful metaknowledge from datasetoids. To achieve this, we chose a simple metalearning setting, namely in terms of the algorithm selection problem, the metafeatures and the meta-level algorithm. The algorithm selection problem consists of predicting, a priori, if pruning a decision tree will improve the quality of the model or not. The implementation of the algorithm for induction of decision tress we used is provided by the `rpart` library included in the R statistical package [8]. Pruned decision trees were obtained using the default parameters of this implementation and unpruned trees were obtained by setting the complexity parameter (`cp`) to 0, which means that a split can make only a very small (> 0) improvement to the model fit. The measure of performance used was classification accuracy,estimated using 10-fold cross-validation.

As mentioned earlier, each meta-example represents one problem (i.e., a dataset or a datasetoid). The class of each meta-example is based on the results of the experiments on the corresponding problem. The class values are `p`, `u` or `t`, meaning, respectively, the winner is the pruned tree, the unpruned tree or that they are tied. Table 1 presents the class distribution, both in the metadata obtained from the datasets as in the metadata from the datasetoids. The table shows that the class distributions are different, which indicates that there may be some underlying differences between datasetoids and datasets. However, we also note that the differences are not so large and that the relative proportions are the same (i.e., ties are the most common, followed by the case when pruning improves accuracy).

Table 1. Class distribution (%) of the metadata corresponding to the datasets and to the datasetoids

metadata	pruned tree wins (`p`)	unpruned tree wins (`u`)	tie (`t`)
datasets	36	23	41
datasetoids	37	10	53

Table 2. Results of using datasetoids on the problem of learning to predict when to prune. The numbers represent accuracy at the meta-level (%). The row "default" represents the proportion of the most frequent class in the metadata of the datasets

algorithm/training set	UCI-R	datasetoids	UCI++
default		41	
`dt`	41	55 (+14)	55
`ld`	41	47 (+6)	48
`rf`	47	62 (+15)	59
`svm`	41	52 (+11)	53
`nn`	45	53 (+8)	50

An important issue in metalearning is the choice of metafeatures. Our goal is not to maximize metalearning results but to assess whether datasetoids provide useful information for metalearning, so we use simple measures. As the induction of decision trees uses entropy, we selected two measures that are based on this concept: the entropy of the class and the average entropy of the attributes [4].

To estimate the accuracy of the metalearning approach, we use the original UCI-R datasets as test set. To ensure independence between the training and test sets, we must guarantee that the datasetoids generated from a given dataset are not used in the training set used to make a prediction for that dataset. Therefore, we used a leave-one-out (LOO) approach. A model is induced for each dataset using all the datasetoids except the ones for the test dataset. For metalearning, we use several algorithms from the R package [8], namely decision trees (`dt`), linear discriminant (`ld`), random forest (`rf`), support vector machines (`svm`) and multi-layer perceptron (`nn`), all with default parameters (which, in the case of `svm` and `nn` imply that some optimization of the parameters is carried out). The measure of metalearning performance is also the classification accuracy, i.e., the proportion of datasets for which a correct decision was prediction was made, in terms of whether it is best to prune or not, or if there is a tie.

As a baseline, we compare the same metalearning approach using only UCI-R datasets. In this case, LOO consists of inducing a model for each dataset using all the other datasets for training. Finally, we have tested a combination of datasets and datasetoids. In this case, the training data for a given test dataset includes all the other datasets plus the datasetoids that were not generated from the test dataset (UCI++).

In the results presented in Table 2, we observe that using a limited number of datasets is not sufficient to predict when to prune a decision tree, at least in the metalearning setting considered in this work. The accuracy obtained is not higher

than the default accuracy, except for `rf` and `nn`. However, by using datasetoids as training data, we obtain systematic gains for all algorithms, ranging from 6% to 15%. These results clearly show that datasetoids contain useful information for predicting whether it is worthwhile to do pruning or not. Note that the level of accuracy achieved (in the 50%-60% range) can still be regarded as relatively low. However, as mentioned earlier, we have used a simple metalearning context with plenty of room for improvement. Additionally, we observe that the impact of combining datasets and datasetoids (column "UCI++") is much smaller (probably not significant from a statistical perspective). This could be expected because there are far less datasets than datasetoids.

5 Discussion

We start by discussing a number of potential anomalies in datasetoids and methods to overcome them. Then, we briefly discuss the relevance of datasets, particularly in terms of the application domain to which they belong. Finally, we discuss further extensions to this approach.

Potential Anomalies. Datasetoids may contain some anomalies which must be detected and corrected.

The first case is the existence of missing values in the symbolic attribute used as the target attribute in the datasetoid. In the pure supervised classification setting we are assuming, this is not possible. One possible correction is to eliminate the lines containing the missing class values. This is one of the possible causes of the second type of anomaly, which is an insufficient number of cases. One possible correction is to eliminate datasetoids which have less than a given number of examples.

When the new target attribute is very skewed, it is difficult, without proper handling (e.g., different missclassification costs), to induce a classification model other than simply predicting the default class. One possible solution is to discard datasets for which the proportion of examples with the majority class is higher than a given level. Note that skewed targets may be interesting for other purposes (e.g., outlier detection, problems with different missclassification costs). Conversely, the new attribute may be very sparse, meaning that there are not enough cases for each different class value to enable any reliable generalization to be done. Again, one solution is to set a maximum threshold on the number of classes or a minimum threshold on the proportion of examples from each class.

It may also be the case that a datasetoid is random because the corresponding target attribute is completely unrelated to all the other attributes and, therefore, the corresponding concept cannot be learned from the available data. A dataset with the same characteristics can also be generated when dealing with a real problem, if the data analyst has extremely poor knowledge engineering skills or there are no predictive attributes available at all. This is, however, much more improbable with real datasets than with datasetoids. One possiblity to identify datasetoids which do not contain a learnable problem is to run several algorithms and eliminate those for which none of the algorithms obtains a satisfactory result.

Finally, it may be the case that two or more attributes represent exactly the same property or that one attribute is completely determined by another one. An example of the latter are the zip code and the city, where the former determines the latter. Therefore, the datasetoids using any of those attributes are target will be trivial. To determine these situations, we can execute several algorithms, as in the previous case, and identify the ones with an accuracy very close to 100%. The models obtained by those algorithms can then be analyzed to check if there are such attributes. Again, we may remove the corresponding datasetoids or, a less radical solution is to remove the dependent attribute.

Relevance. One may question the relevance of the learning problems that a datasetoid represents. According to [5], there are three types of classification datasets in terms of the type of class. The class can be (1) a label, (2) a random variable or (3) determined by a partition of the attribute space. Given that this distinction also applies to the independent attributes in a dataset, new datasets of all the types are generated by the method described. Analyzing the symbolic attributes of the `soybean (large)` dataset, for instance, we observe that the date of the observation, the air temperature and condition of the stem will generate datasetoids of each of the three types, respectively.

From the point of view of the application domain of the data, most datasetoids are pure nonsense. For instance, predicting the product purchased, given the customer and the amount purchased is probably not an interesting application. However, validity from the point of view of the original domain is important if the goal is to compare the performance of algorithms on the corresponding application. However, this is not true in the kind of problem we are concerned with, where the goal is to obtain metaknowledge, i.e., to relate properties (metafeatures) of the data with the (relative) performance of algorithms.

Future Work. The good results reported in this paper, make a range of similar approaches equally appealing. For instance, datasetoids can also be generated by discretizing continuous attributes. This approach has the very interesting possibility of controlling the number of classes and their distribution. This method could also be used to generate classification problems from regression problems.

Finally, it seems that is possible to generate datasetoids for all kinds of supervised learning problems. For instance, regression datasetoids can be generated by using continuous attributes as target. Additionally, it may be possible to generate datasetoids from any kind of dataset and not necessarily from learning datasets. If these expectations are met, a huge body of highly reliable metaknowledge can be generated.

6 Conclusions

We proposed a method to generate new learning datasets from existing ones, simply consisting of switching the target attribute with each of the independent ones. The new datasets, referred to as *datasetoids*, are not interesting as applications *per se*. But, as shown by our experiments on the problem of metalearning

when to prune a decision tree, they can be useful in situations where it is required to carry out experiments on a group of datasets to obtain metaknowledge. This metasknowledge may consist of models relating the results obtained with abstract properties of the data or may simply be a general assessment of the competitiveness of one or more methods, as is usual in published machine learning research. A larger amount of datasets available to perform these studies also decreases the possibility of drawing wrong conclusions due to results obtained by chance.

We identified a number of anomalies that may occur in the new data sets, due to the artificial nature of the problems they represent. Methods to identify and correct those anomalies have been suggested.

The successful results obtained, opens a wide range of extensions to this method, therefore enabling the generation of new datasets that could solve current shortcomings in the production of metaknowledge.

Acknowledgements. Thanks to Alípio Jorge for the inspiration for naming the new datasets as datasetoids. This work was partially funded by FCT (Programa de Financiamento Plurianual de Unidades de I&D and project Rank! - PTDC/EIA/81178/2006).

References

1. Asuncion, A., Newman, D.J.: UCI machine learning repository (2007)
2. Blockeel, H., Vanschoren, J.: Experiment databases: Towards an improved experimental methodology in machine learning. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS, vol. 4702, pp. 6–17. Springer, Heidelberg (2007)
3. Brazdil, P., Giraud-Carrier, C., Soares, C., Vilalta, R.: Metalearning: Applications to Data Mining. In: Cognitive Technologies. Springer, Heidelberg (2009)
4. Brazdil, P., Soares, C., Costa, J.: Ranking learning algorithms: Using IBL and metalearning on accuracy and time results. Machine Learning 50(3), 251–277 (2003)
5. Henery, R.J.: Classification. In: Michie, D., Spiegelhalter, D.J., Taylor, C.C. (eds.) Machine Learning, Neural and Statistical Classification, Ellis Horwood, vol. 2, pp. 6–16 (1994)
6. Hilario, M., Kalousis, A.: Quantifying the resilience of inductive classification algorithms. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS, vol. 1910, pp. 106–115. Springer, Heidelberg (2000)
7. Macià, N., Orriols-Puig, A., Bernadó-Mansilla, E.: Genetic-based synthetic data sets for the analysis of classifiers behavior. his 0, 507–512 (2008)
8. R Development Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2008) ISBN 3-900051-07-0
9. Soulié-Fogelman, F.: Data mining in the real world: What do we need and what do we have? In: Ghani, R., Soares, C. (eds.) Proceedings of the Workshop on Data Mining for Business Applications, pp. 44–48 (2006)

Accurate Synthetic Generation of Realistic Personal Information

Peter Christen[1] and Agus Pudjijono[2]

[1] School of Computer Science, The Australian National University
Canberra ACT 0200, Australia,
peter.christen@anu.edu.au
[2] Data Center, Ministry of Public Works of Republic of Indonesia
Jakarta, 12110, Indonesia,
apudjijono@pu.go.id

Abstract. A large portion of data collected by many organisations today is about people, and often contains personal identifying information, such as names and addresses. Privacy and confidentiality are of great concern when such data is being shared between organisations or made publicly available. Research in (privacy-preserving) data mining and data linkage is suffering from a lack of publicly available real-world data sets that contain personal information, and therefore experimental evaluations can be difficult to conduct. In order to overcome this problem, we have developed a data generator that allows flexible creation of synthetic data containing personal information with realistic characteristics, such as frequency distributions, attribute dependencies, and error probabilities. Our generator significantly improves earlier approaches, and allows the generation of data for individuals, families and households.

Keywords: Artificial data, data matching, data linkage, privacy, data mining pre-processing.

1 Introduction

Today, massive amounts of data are collected by many organisations in both the private and public sectors. A large proportion of this data is about people, and often personal identifying details are stored together with application specific information, for example employment or medical details. When such data is analysed within an organisation, then normally, depending upon the desired outcomes, only parts of the personal information is used for an analysis (like age, gender, or postcode). In these cases, privacy and confidentiality are generally not of great concern, as the results of the analysis are only used within an organisation, and no detailed private or confidential information is released.

However, when data is being shared between organisations, privacy and confidentiality become of paramount importance, because personal information is commonly required to match records from different databases [1]. The aim of such linkages is to match all records that refer to the same entity. Because real-world data is commonly dirty [2] (contains errors and variations) and often no

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 507–514, 2009.
© Springer-Verlag Berlin Heidelberg 2009

unique entity identifiers are available, sophisticated approximate matching algorithms are required that use the available personal identifiers [3,4].

The process of *data linkage* or *matching* has in the past decade been recognised as an important and challenging problem, and a variety of novel linkage algorithms have been developed [3,4]. They mainly address the technical challenges of matching accuracy and scalability to very large databases. Another challenge for data linkage research is the lack of publicly available real-world test data sets that allow evaluation of new algorithms. This lack is due to privacy concerns, because it is illegal in most countries to publish data that contains, for example, personal details of customers or patients. As a result, data linkage researchers have to use publicly available data sets, or use their own (confidential) data, which prevents others from repeating experimental studies [5].

An alternative is to use synthetically generated data. This approach has several advantages. First, a user can control the size (number of records) and quality (error characteristics) of the generated data sets. Second, such data can be published, and thus allows other researchers to repeat experiments and better evaluate algorithms. Third, the generator itself can be published, allowing others to generate data that is specifically tailored to their use, for example to their country or application domain. Fourth, because it is known which of the generated records are matches, it is possible to calculate matching rates [3].

Besides data linkage research, any application area where data containing personal information is required for research purposes can benefit from synthetically generated data, because such data removes privacy and confidentiality concerns. Examples include research into privacy-preserving data sharing [1], publishing and mining, or statistical micro-data confidentiality.

The challenges when generating synthetic data are that it is not easy to create data with characteristics that are similar to real-world data. The frequency and error distributions of values have to follow real-world distributions, and dependencies between attributes have to be modelled. This paper describes a data generator with such characteristics. It is a significant improvement over earlier generators [2,5,6], which created data in less realistic ways.

2 Synthetic Data Generation

As illustrated in Fig. 1, the data generator works in two steps. First, a user specified number of *original* records is created based on real values and their frequencies and dependencies, or using specific attribute generation rules [5,7]. Second, randomly selected original records are modified into *duplicate* records. Alternatively, family and household records can be generated. As can be seen in Fig. 4, each record is given a unique identifier (*'rec_id'*) that will facilitate the calculation of matching rates [3].

2.1 Original Record Generation

For original records, the values in name and address attributes are created randomly using frequency tables. Such tables can, for example, be extracted from

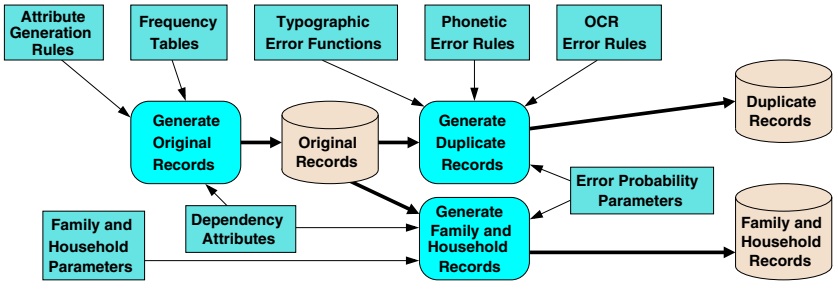


Fig. 1. Overview of the data generation process

telephone directories. For date, telephone and social security number attributes, a user can specify generation rules that determine the range of dates (such as start and end birth dates), or the number of digits (for example for telephone numbers). In the following, we describe the two major novel features of our data generator [7]: family and household data, and attribute dependencies.

Generating Family and Household Data. The records for a family are generated by first selecting an original record at random. According to its age and gender values, it is assigned one of the roles *husband*, *wife*, *son* or *daughter*. The next step is to randomly select how many other records are to be generated for this family. These records are then created by keeping the surname of the first family record, but modifying given name, gender and age values. Address attribute values are generally kept the same for all members of a family. Depending upon the age of son and daughter records, however, a new address will be created with a certain probability, assuming the child has left home.

Household records are generated similarly, with the main difference being that all records in a household have different names but the same address, and that all age values are above 18 (one of many parameters that a user can set [7]).

Attribute Dependencies. A dependency occurs if the values in an attribute depend upon the values in one or more other attributes. For example, given names depend on the gender and the cultural background of a person, while suburb/town names depend on the state/territory they are located in. These dependencies are based on frequency tables, such as the one shown in Fig. 2.

When generating the original records, the *key* attributes (the attributes that others depend on) are generated first, and according to a selected key attribute value, a value from the dependent attribute is randomly chosen according to the corresponding frequency distribution. For example, using the values from Fig. 2, if the state *'QLD'* has been selected, the suburb name *'Allansford'* would be chosen with likelihood 6.25%, *'Allendale'* with likelihood 68.75%, and *'Allestree'* with likelihood 25%. To introduce randomness into the data, with a certain likelihood, as set by the user, a dependency is not followed, but rather a value is randomly chosen from the overall frequency table of the dependent attribute.

```
ACT  :  Acton;5, Ainslie;10, Amaroo;7, Belconnen;12
NSW  :  Albanvale;3, Albert Park;6, Alberton;4, Albion Park;9
QLD  :  Allansford;1, Allendale;11, Allestree;4
```

Fig. 2. Sample from a combined dependency–frequency look-up table with Australian state names on the left, and suburb names and their frequencies on the right

2.2 Error Modelling and Duplicate Record Modification

As illustrated in Fig. 3, data can be entered through a variety of channels, each having its own error characteristics. For example, handwritten forms that are processed using optical character recognition (OCR) software will likely include substitutions between similar looking characters. On the other hand, phonetic errors, like the variations *'Dickson'* and *'Dixon'*, are introduced when information is dictated using speech recognition, or typed manually. Typing itself introduces certain errors more likely than others. Depending upon keyboard layout, mistyping neighbouring keyboard keys, such as *'a'* and *'s'*, can occur. Often, depending upon the data entry channel, a combination of error types is introduced.

Our data generator can model typographic, phonetic and OCR errors. For each error type, a user can set how likely they are introduced when the duplicate records are generated. Setting the likelihood of typographic and phonetic errors to 0, for example, will result in duplicate records that only contain OCR errors.

Typographic Errors. These errors include insertion, deletion, and substitution of a character; and transposition of two adjacent characters. They are implemented as functions that apply the corresponding modification to a given input string with a certain likelihood (as set by the user), and return the modified string. Following studies of error distributions [8], the position of a modification is randomly chosen such that it more likely occurs in the middle or towards the end of a string, because real errors are less likely at the beginning of names.

Optical Character Recognition Errors. OCR modifications are based on rules that consider shape similarity among characters, such as *'5'* and *'S'* or *'w'* and *'vv'*. Around fifty such rules are used, representing the most likely OCR variations that might occur. When duplicate records are generated, one or more possible OCR modifications will be randomly selected and applied to an input string, and the modified string will be inserted into the duplicate record. Like phonetic errors, OCR errors can be a single character modification or a combination of modifications (like a substitute and delete, or a delete and insert).

Phonetic Errors. These errors are usually more complex than typographic or OCR errors, as they often include changes of character groups and depend upon the position within a string. The idea behind our approach in modelling phonetic errors is to employ the rules that are used in phonetic encoding methods [9], such as *Phonix* and *Double-Metaphone*. In encoding methods, such rules are used to

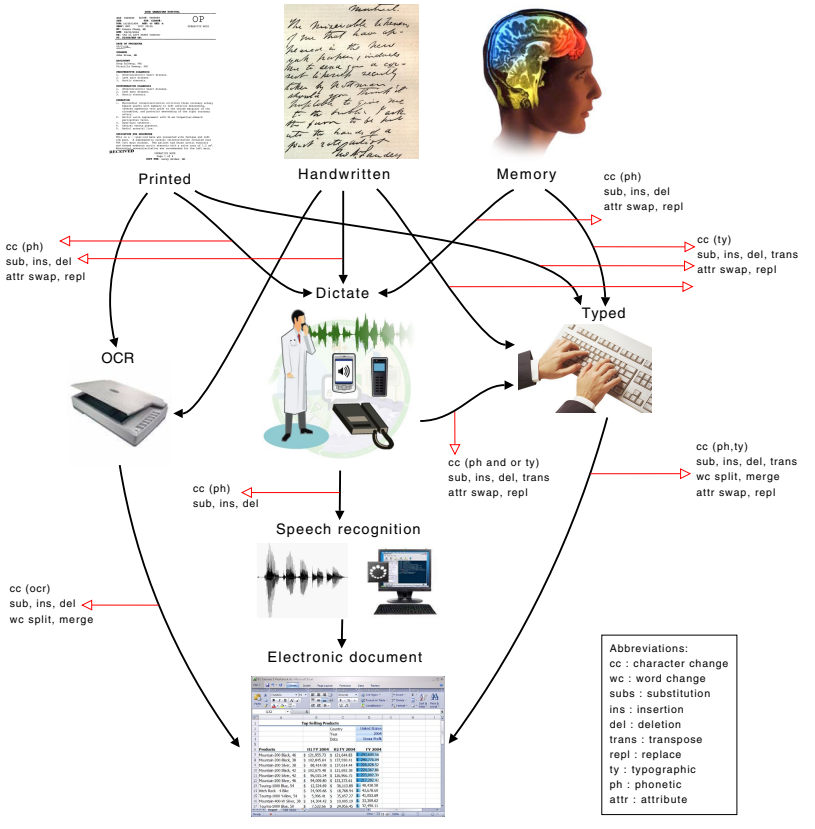


Fig. 3. Model of data sources and possible errors introduced during data entry

group similar sounding names together, while we employ them to modify a name in order to generate a similar sounding variation of it. Currently, around 350 phonetic rules are used, each made of the following seven components.

1. **Position.** The position within the input string where the *original* string pattern can occur. The four possible values are: *ALL* (can occur anywhere), *START* (must occur at the beginning), *MIDDLE* (must occur in the middle, but not at the beginning or end), and *END* (must occur at the end).
2. **Original pattern.** This is the string (made of one or more characters) that will be replaced with the *substitute* string pattern if the rule is applied.
3. **Substitute pattern.** This is the string (made of zero or more characters) that will replace the *original* string pattern if the rule is applied.
4. **Precondition.** A condition that must occur before the *original* string pattern in order for this rule to become applicable. A precondition can be that the character immediately before the original pattern is a vowel (*'V'*), a consonant (*'C'*), or a more complex expression [7].

5. **Postcondition.** Similarly, a condition that must occur after the *original* string pattern in order for this rule to become applicable.
6. **Pattern existence condition.** This condition requires that a certain given string pattern does (*'y'* flag) or does not (*'n'* flag) occur in the input string.
7. **Start existence condition.** Similarly, this condition requires that the input string starts with a certain string pattern (*'y'* flag) or not (*'n'* flag).

The last four components of a rule (its conditions) can be set to *'None'* if they are not required. In the following we give two illustrative examples.

- `ALL, h, @, None, None, None, None` *(mustapha → mustapa)*
  In this rule there are no conditions, so any occurrence of the character *'h'* is being removed (replaced with an empty string – denoted with a *'@'*).
- `END, le, ile, C, None, None, None` *(bramble → brambile)*
  The precondition in this rule is *'C'*, which means the character before the original pattern must be a consonant. The character before the pattern *'le'* in this example is a *'b'*, so the modification *'le'* into *'ile'* is applied.

3 Experimental Evaluation

Our data generator is implemented as part of the *Febrl* (Freely Extensible Biomedical Record Linkage) open source data linkage system [10],[1] and is written in the *Python* programming language. Due to the availability of its source code, it can be modified and extended according to a user's needs. A large number of parameters can be set by the user, including the number of original and duplicate records to be generated, the frequency and dependency look-up tables to be used, the distributions used for household and family records, and the various error characteristics to be applied when duplicates are created [7].

We used a variety of data sets to create our look-up tables, including a data set containing 99,571 names and their culture of origin (37 different cultures) [11], a data set with Australian postcode, suburb and state values as available from the *Australia Post* Web site,[2] and the various look-up tables supplied with the *Febrl* data linkage system [10]. Error and modification probabilities were set according to real world studies on typographic and other errors [8,12,13,14].

Some example data that was created using our generator is shown in Fig. 4. As can be seen, the record identifiers (*'rec_id'*) designate if a record is an original or a duplicate, and the duplicate records are numbered and refer back to their original record, in order to allow the calculation of matching rates [3].

We have conducted a large number of experiments to validate our data generator, by comparing real data sets with synthetic data that was generated using frequency tables based on the real data. A detailed analysis and discussion is provided elsewhere [7]. A user can repeat our experiments by downloading the *Febrl* system and run the generator program supplied with it (and possibly change parameter settings according to her or his needs).

[1] Available from: `https://sourceforge.net/projects/febrl/`
[2] Available from: `http://www.post.com.au/postcodes/`

rec_id, age, given_name, surname, street_number, address_1, address_2, state, suburb, postcode
rec-1-org, ***33***, ***Madison***, Solomon, ***35***, Tazewell ***Circuit***, Trail View, ***VIC***, ***Beechboro***, ***2761***
rec-1-dup-0, 33, Madisoi, Solomon, 35, Tazewell Circ, Trail View, VIV, Beech Boro, 2761
rec-1-dup-1, , Madison, Solomon, 36, Tazewell Crct, Trail View, VIC, Bechboro, 2716
rec-2-org, 39, ***Desirae***, ***Contreras***, 44, Maltby Street, ***Phillip Lodge***, NSW, ***Burrawang***, 3172
rec-2-dup-0, 39, Desirae, Kontreras, 44, Maltby Street, Phillip Loge, NSW, Burrawank, 3172
rec-2-dup-1, 39, Desire, Contreras, 44, Maltby Street, Fillip Lodge, NSW, Buahrawang, 3172
rec-3-org, ***81***, ***Madisyn***, Sergeant, 6, ***Howitt*** Street, Creekside Cottage, VIC, ***Nangiloc***, 3494
rec-3-dup-0, 87, Madisvn, Sergeant, 6, Hovvitt Street, Creekside Cottage, VIC, Nanqiloc, 3494

Fig. 4. Examples of generated data. Records '1' include typographical errors, records '2' phonetic errors, and records '3' OCR errors. The original values that were modified are in bold-italics, and their corresponding modifications are underlined.

4 Related Work

A first data generator for personal information was developed in the 1990s [2]. It allowed the generation of data based on lists of names, cities, states and postcodes, however without using any frequency distributions. A user could set the size of the data sets to be generated, and the types and amount of errors to be introduced. An improved generator was described more recently [6]. It allowed attribute values to become missing, and it improved the variability of the created values. It is however unclear if this generator is using frequency information, as not many details have been described.

A first simple version [5] of our generator has been freely available as part of the *Febrl* [10] data linkage system. It improved upon earlier generators by including frequency tables of attribute values, more flexible setting of individual error probabilities, as well as inclusion of look-up tables with name variations (to be used for example for nick-names, known phonetic variations, and common misspellings). This generator however does not include attribute dependencies, does not allow creating family or household record groups, and it does not model errors as accurately as the new version described in this paper.

The phonetic error model presented in Sect. 2.2 is based on rules that were originally developed for phonetic encoding methods [9]. A common feature of phonetic encodings is that they convert name strings into codes according to how a name is being pronounced. Names that sound similar are converted into the same code. This is obviously a language dependent process, and most phonetic encoding methods have been developed for the English language.

Our work is also based on various studies that have analysed spelling and data entry errors and their corrections [8,12,13,14]. These studies found that most errors are single character errors, and that the distribution of error types depends upon the mode of data entry. For example, OCR output contains almost exclusively substitution errors, while this type of error accounts for less than 20% of errors with keyboard based manual data entry [8]. Typically, up to 95% of misspellings in keyboard entry only contain one error; with only around 8% of first letters incorrect, compared to almost 12% of second and nearly 20% of third letters.

5 Conclusions

We have presented a data generator for personal information that allows the generation of realistic synthetic data based on frequency tables and attribute generation rules. There are various ways to improve our generator. First, allowing the generation of not just personal information, but also application specific attributes (like medical, employee, or customer details) will make our generator applicable to the wider data mining community. Second, extending family records to include other roles (such as cousins, aunts, uncles, etc.) and allow culture specific parameter settings will enable the generation of the complex family connections that occur in real life. Third, enabling Unicode characters will make our generator more international and will allow the generation of data sets containing, for example, Thai, Chinese, or Arabic characters. Finally, adding a graphical user interface will facilitate the setting of the many possible parameters. Another part for our future work will be to fully integrate our new data generator into our *Febrl* data linkage system [10].

References

1. Christen, P.: Privacy-preserving data linkage and geocoding: Current approaches and research directions. In: ICDM PADM workshop, Hong Kong (2006)
2. Hernandez, M., Stolfo, S.: Real-world data is dirty: Data cleansing and the merge/purge problem. Data Mining and Knowledge Discovery 2(1), 9–37 (1998)
3. Christen, P., Goiser, K.: Quality and complexity measures for data linkage and deduplication. In: Quality Measures in Data Mining. Studies in Computational Intelligence, vol. 43, pp. 127–151. Springer, Heidelberg (2007)
4. Elmagarmid, A., Ipeirotis, P., Verykios, V.: Duplicate record detection: A survey. IEEE Transactions on Knowledge and Data Engineering 19(1), 1–16 (2007)
5. Christen, P.: Probabilistic data generation for deduplication and data linkage. In: Gallagher, M., Hogan, J.P., Maire, F. (eds.) IDEAL 2005. LNCS, vol. 3578, pp. 109–116. Springer, Heidelberg (2005)
6. Bertolazzi, P., De Santis, L., Scannapieco, M.: Automated record matching in cooperative information systems. In: DQCIS, Siena, Italy (2003)
7. Pudjijono, A.: Probabilistic data generation. Master of Computing (Honours) thesis, Department of Computer Science, The Australian National University (2008)
8. Pollock, J., Zamora, A.: Automatic spelling correction in scientific and scholarly text. Communications of the ACM 27(4), 358–368 (1984)
9. Christen, P.: A comparison of personal name matching: Techniques and practical issues. In: ICDM MCD workshop, Hong Kong (2006)
10. Christen, P.: Febrl – An open source data cleaning, deduplication and record linkage system with a graphical user interface. In: ACM KDD, Las Vegas (2008)
11. Phua, C., Lee, V., Smith-Miles, K.: The personal name problem and a recommended data mining solution. In: Encyclopedia of Data Warehousing and Mining, 2nd edn., Information Science Reference (2008)
12. Damerau, F.: A technique for computer detection and correction of spelling errors. Communications of the ACM 7(3), 171–176 (1964)
13. Hall, P., Dowling, G.: Approximate string matching. ACM Computing Surveys 12(4), 381–402 (1980)
14. Kukich, K.: Techniques for automatically correcting words in text. ACM Computing Surveys 24(4), 377–439 (1992)

An Efficient Approximate Protocol for Privacy-Preserving Association Rule Mining

Murat Kantarcioglu[1], Robert Nix[1], and Jaideep Vaidya[2]

[1] University of Texas at Dallas, Richardson, TX 75080, USA
{muratk,robertn}@utdallas.edu
[2] Rutgers University, Newark, NJ, 07102, USA
jsvaidya@rbs.rutgers.edu

Abstract. The secure scalar product (or dot product) is one of the most used sub-protocols in privacy-preserving data mining. Indeed, the dot product is probably the most common sub-protocol used. As such, a lot of attention has been focused on coming up with secure protocols for computing it. However, an inherent problem with these protocols is the extremely high computation cost – especially when the dot product needs to be carried out over large vectors. This is quite common in vertically partitioned data, and is a real problem. In this paper, we present ways to efficiently compute the approximate dot product. We implement the dot product protocol and demonstrate the quality of the approximation. Our dot product protocol can be used to securely and efficiently compute association rules from data vertically partitioned between two parties.

1 Introduction

Privacy-preserving data mining has made major advances in the recent years. Many protocols have been proposed for different data mining algorithms such as classification, association rule mining, clustering, and outlier detection, etc. [1] provides a comprehensive survey. There are two main types of technique – perturbation based methods and secure multiparty computation techniques. In the perturbation methods the data is locally perturbed before delivering it to the data miner. Special techniques are used to reconstruct the original distribution (not the actual data values), and the mining algorithm needs to be modified to take this into consideration. The seminal paper by Agrawal and Srikant [2] introduced this approach in the form of a procedure to build a decision tree classifier from perturbed data. The second approach assumes that data is distributed between two or more sites that cooperate to learn the global data mining results without revealing the data at individual sites. This approach was introduced by Lindell and Pinkas [3], with a method that enabled two parties to build a decision tree without either party learning anything about the other party's data, except what might be revealed through the final decision tree. Typically, the techniques used are cryptographic. While the first approach has known security problems[4,5] and cannot lead to provably secure solutions, the second approach has typically been too computationally intensive. This is especially the case for

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 515–524, 2009.
© Springer-Verlag Berlin Heidelberg 2009

vertically partitioned data (e.g., [6]), where unlike horizontally partitioned data (e.g.,[7]) little data summarization can be carried out before engaging in the distributed protocol.

So the essential question is – how can we mine data in an *efficient* and *provably secure* way? Since most data mining deals with aggregates, one way of solving this conundrum is through approximation. It may be possible to create highly efficient provably secure protocols that approximate the answer. In this paper, we present one of the first such approaches. Instead of directly approach one particular data mining technique, we present an approximate protocol for computing the scalar (dot) product of two vectors owned by two different parties. This is very important, since the dot product is actually used as a sub-protocol in many data mining tasks such as classification, association rule mining, etc. As a sample application, we show how a completely secure and efficient association rule mining protocol could easily be created using our protocol. We implement our protocol and demonstrate the significant efficiency and quality of the approximation, and discuss the implications for association rule mining.

1.1 Related Work

While there has been a lot of related work in privacy-preserving data mining, due to space constraints, we only focus on the tightly related efforts. Several protocols have been proposed for computing the scalar product[8,6,9,10]. Out of these [6] is an algebraic protocol that requires quadratic complexity though the constituent operations are quite simple. The other protocols all require significant cryptographic operations and are significantly (orders of magnitude) less efficient. [11] present a simple sampling based protocol for approximately computing the scalar product that is similar in notion to ours. However, since our protocol is based on Bloom filters, it is more sophisticated, has better bounds on accuracy, and is highly efficient. [12] proposes a technique using bloom filters to do association rule mining. However, here transactions are represented via bloom filters and the association rule mining is done centrally, where as enable efficient distributed association rule mining. There is also work on privacy-preserving association rule mining for vertically partitioned data[6,13]. However, by comparison, this is the first secure, highly efficient protocol proposed for this problem.

Securely computing the dot product of two vectors is a very important subprotocol required in many privacy-preserving data mining tasks. Many secure dot product protocols have been proposed in the past. [6,8,9,10,11]. For comparison purposes, we use the method of Goethals et al., described in [10], as an exact scalar product protocol, since it is quite simple and provably secure. Its runtime is $O(n)$ in public-key encryptions.

1.2 Bloom Filters

We now give some background on Bloom Filters. Bloom filters have been extensively used for various application domains ranging from networking to databases [14]. Basically, a bloom filter represents a set $S = \{x_1, x_2, \ldots, x_n\}$ of n elements

using array of m bits ($m \leq n$), initially all set to 0. For each element $x \in S$, we use k independent random hash functions $h_1(), h_2(), \ldots, h_k()$ with range $\{1, \ldots, m\}$ such that the bits $h_i(x)$ of the array are set to 1 for $1 \leq i \leq k$. In this basic version, a location can be set to 1 multiple times but only the first change has an effect. To check whether an item $t \in S$, we need to check whether all $h_i(t)$ for $1 \leq i \leq k$ are set to 1. If they are not all set to 1, we can conclude that $t \notin S$. On the other hand, if all $h_i(t)$ for $1 \leq i \leq k$ set to 1, we can assume that $t \in S$, with some nonzero probability.

Bloom filters can be used to approximate the intersection size between two sets. Given two bloom filters with the same m and k values that represent two sets S_1 and S_2, we can approximate $|S_1 \cap S_2|$ by getting the dot product of the two filters. More precisely, let Z_1 (resp. Z_2) be the number of 0s in the filter S_1 (resp. S_2) and Z_{12} be the number of 0s in the inner product, then we can approximate $|S_1 \cap S_2|$ using the following formula [14]:

$$\frac{1}{m}\left(1 - \frac{1}{m}\right)^{-k|S_1 \cap S_2|} \approx \frac{Z_1 + Z_2 - Z_{12}}{Z_1 Z_2} \tag{1}$$

$$\implies |S_1 \cap S_2| \approx \frac{\ln\left(m(Z_1 + Z_2 - Z_{12})\right) - \ln(Z_1) - \ln(Z_2)}{-k \ln\left(1 - \frac{1}{m}\right)} \tag{2}$$

2 Algorithm

Instead of looking at the computation of the scalar product, we look at a slight variant of this problem – checking if the scalar product of two distributed vectors is greater than some threshold. In other words, given two vectors X_1 and X_2, we need to compute whether $X_1 \cdot X_2 = \sum_{i=1}^{n} X_1[i].X_2[i] \geq t$ for a given threshold

Algorithm 1. Approximate threshold dot product algorithm

Require: X_1 belongs to P_1 and X_2 belongs to P_2 each of size n. Publicly known common parameters: $m << n$ (size of the bloom filters), hash functions $h_1, \ldots, h_k$, constant integer c such that $c\frac{1}{m}(1 - \frac{1}{m})^{-kt}$ is an integer, prime p such that $n^2 c < p$, threshold value t.

1: **for** each P_i **do**
2: Create $S_i = \{j \mid X_i[j] = 1\}$
3: Create a bloom filter B_i of size m using S_i and the common k hash functions.
4: **end for**
5: Each party participates in a secure dot product protocol (e.g., [10]) to get random share u_i where $u_1 + u_2 = B_1 \cdot B_2 \bmod p$ {Note that $Z_{12} = m - B_1 \cdot B_2$}
6: Each party participates in a secure multiplication protocol (e.g., [3]) to get random share v_i where $v_1 + v_2 = (c\frac{1}{m}(1 - \frac{1}{m})^{-kt}) Z_1 Z_2 \bmod p$
7: P_1 sets $y_1 \leftarrow c.(Z_1 + u_1 - m) \bmod p$
8: P_2 sets $y_2 \leftarrow c.(Z_2 + u_2) \bmod p$
9: Each party participates in a secure comparison protocol (e.g, [3]) to learn whether $((y_1 + y_2) \bmod p) \geq ((v_1 + v_2) \bmod p)$

value t. This variation is more useful in many applications, such as computing association rules (as will be seen later).

To do this, we use an approximation based on bloom filters. Note that for a given binary vector X_i, we can create a set S_i where j is an element of S_i if the j^{th} bit of X_i is set to one (i.e., $S_i = \{j \mid X_i[j] = 1\}$). It is easy to see that $X_1 \cdot X_2 = |S_1 \cap S_2|$. Using this fact, we can efficiently approximate $X_1 \cdot X_2$, as follows: First, for each X_i, we can create the corresponding set $S_i = \{j \mid X_i[j] = 1\}$. Given S_i, and k hash functions, we can create a bloom filter of size m for each S_i. Let Z_1 (resp. Z_2) be the number of 0s in the filter S_1 (resp. S_2) and Z_{12} be the number of 0s in the inner product of the bloom filters for S_1 and S_2. Using equation 1, we can compute the following approximations:

$$X_1 \cdot X_2 = |S_1 \cap S_2| \geq t \tag{3}$$

$$\iff -k|S_1 \cap S_2| \leq -kt \tag{4}$$

$$\iff (1 - \frac{1}{m})^{-k|S_1 \cap S_2|} \geq (1 - \frac{1}{m})^{-kt} \tag{5}$$

$$\iff \frac{Z_1 + Z_2 - Z_{12}}{Z_1 Z_2} \approx \frac{1}{m}(1 - \frac{1}{m})^{-k|S_1 \cap S_2|} \geq \frac{1}{m}(1 - \frac{1}{m})^{-kt} \tag{6}$$

$$\iff Z_1 + Z_2 - Z_{12} \geq Z_1 Z_2 \frac{1}{m}(1 - \frac{1}{m})^{-kt} \tag{7}$$

The above approximations could be used to give an efficient secure approximate protocol for checking whether $X_1 \cdot X_2 \geq t$ for given binary vectors X_1 and X_2 and threshold value t. The algorithm 1 gives the details of our secure approximate threshold dot product algorithm. In the algorithm, each party first creates its own bloom filter using the common bloom filter parameters (line: 3). Secondly, they participate in a secure dot product algorithm using their own private bloom filters and get the random shares of the dot product result (line: 5). Thirdly, each party participates in secure multiplication protocol using their private Z_i values to get the random share of the multiplication result (line: 6). Finally, they combine their local inputs and random shares accordingly to get the final result (lines: 7-9).

We would like to stress that similar secure approximate dot product algorithm could be given using existing secure ln() function evaluation techniques [3] and the equation 2. For secure approximate dot product case, we need to use just one invocation of the secure ln() protocol to get the shares of the $ln(Z_1 + Z_2 - Z_{12})$. Similarly, Z_{12} could be computed by using any secure dot product protocol.

2.1 Security

It is easy to prove that algorithm 1 is secure in semi-honest model [15] assuming that the dot product, multiplications and comparison protocols that are used are secure. Below, we give a sketch of the security proof.

Theorem 1. *Given secure dot product, multiplication and comparison protocols, algorithm 1 is secure in the semi-honest model.*

Proof. **(Sketch)** After secure multiplication and secure dot product protocols, each party only gets random shares. Therefore simulator for each P_i could simply generate random values to simulate u_i and v_i. Since each y_i value only depends on the local or publicly known inputs, each party does not learn anything while computing y_i values. Finally, each party participates in a secure comparison protocol to learn the final result. Again assuming that comparison protocol is secure, we can provide a simulator for each P_i that can just return the final result. Using composition theorem [15], we can conclude our proof sketch.

2.2 Computational and Communicational Cost

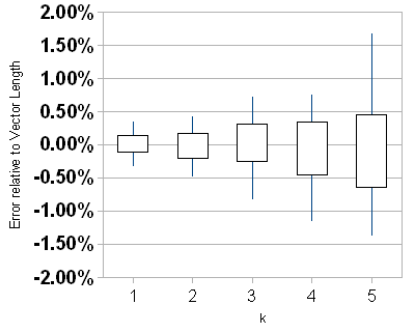
It is clear from the description of algorithm 1 that we need to use a secure dot product algorithm on binary vectors of size m (i.e., the bloom filters). All the other operations are constant time and do not depend on either n or m. In our method, we also have an additional cost of $O(nk)$ (n is the size of the original binary vectors, k is the number of hash functions used for bloom filter) hash function operations to create the bloom filters. As we show in the next section, the hashing cost is negligible compared to expensive public key operations used in the secure dot product protocol. As long as, $m << n$, our method would be much faster than the typical implementation of a secure dot product protocol. Even more, if more efficient techniques are found for securely computing dot product of two binary vectors, we may just use it to improve our performance even further. The communication cost of our protocol is also much lower, since it is only proportional to m. In the next section, we discuss the appropriate values of m and k that need to be used in practice for efficiency and accuracy.

3 Experimental Results

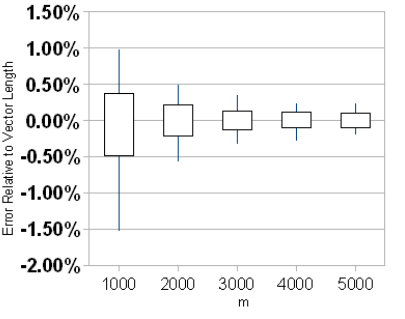
To evaluate the efficiency and accuracy of our approximation algorithm, we designed and ran several experiments. We considered the effects of vector length (ℓ), vector density (d), the actual intersection of the two vectors (i), and the bloom filter parameters m (length of filter) and k (number of hash functions) on the performance of the algorithm. As an underlying secure dot product protocol, we used the completely secure homomorphic encryption-based protocol [10]. The Paillier key length was set to 512 bits. The hash functions used were keyed versions of SHA-1 via the HMAC protocol [16].

3.1 Accuracy

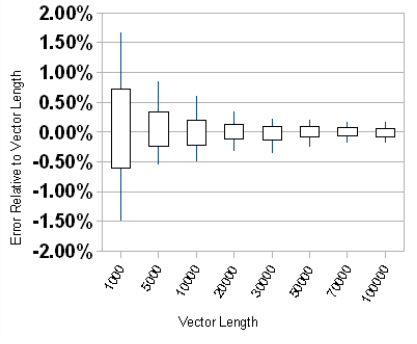
Several experiments were carried out to determine the accuracy of the approximate dot product method. The results are depicted in the figures below. Each data point represents the results of 100 trials, depicting the distribution in which the dot product estimates fell. Note that the bars on the figures represent plus or minus one standard deviation from the mean, while the extended lines denote the minimum and maximum error. Error is given relative to the vector length.



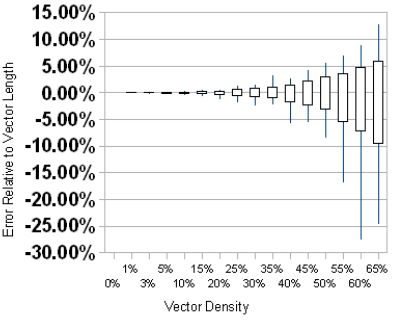
(a) Varying k



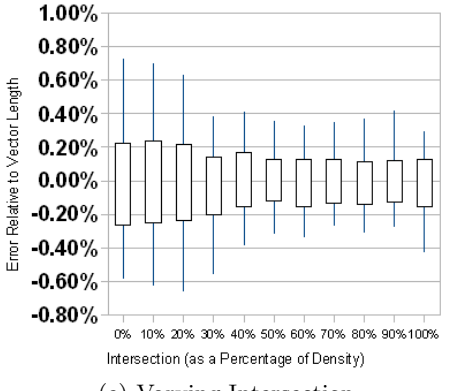
(b) Varying m



(c) Varying Vector Length



(d) Varying Vector Density



(e) Varying Intersection

length of vector (l)	20000
number of filters (k)	1
length of filter (m)	15% of l
density (d)	10% of l
intersection	50% of d

(f) Default Values

Fig. 1. Accuracy

Figure 1 shows the accuracy when different parameters are varied. Figure 1(f) gives the default values of the different parameters in all of the experiments. Figure 1(a) plots the accuracy for different values of k (number of hash functions). The results show that accuracy actually decreases with increasing k. This

is because the distortion increases while increasing k when the filter is small. In a separate experiment, it was determined that an increase in k did increase accuracy for $m = 40000$, however, this is not a practical use of the protocol, since the Bloom filters would no longer give us the efficiency we desire. Overall, the experiment shows that for both accuracy and efficiency, nothing is gained by increasing k for small filters.

In Figure 1(b), we study the relation of accuracy with filter length. We can see that the accuracy of the approximation increases as m increases. This makes sense, since as the filter length increases, there is less distortion (and collision) in the data, and more accurate results occur. However, even in the lowest case of $m = 1000$ (95% data compression), the error was no more than 1.5% of the vector length, and the average absolute error was only around 9, meaning that with a significant probability, even a high compression ratio can result in high utility.

Next, we study the effect of vector length. Figure 1(c) reports accuracy results when the vector length varies from 1000 to 100000. The figure shows that the relative error decreases as the vector length increases. This makes sense since we keep the filter length a fixed percentage of the vector length. A key property of Bloom filters is that increase in vector length requires only sub-linear increase in the Bloom filter length to maintain similar accuracy. This helps us significantly for large vectors. As can be seen, the relative error was always within 2% of the vector length.

Next, we study the effect of varying the density of the vector. Figure 1(d) shows the relative error when the density was varied from 0% to 65%. The rest of the parameters were kept at the default values. As expected, the accuracy drastically fell off as the density of the vectors increased, although reasonable (within 5%) accuracy could still be achieved for vectors as dense as 35%. One interesting thing to note is the fact that at 0% density (that is, no 1's in either vector) the approximate algorithm will give an exact answer of 0, and there will be no error.

Finally, we vary the vectors to achieve the size of the intersections to be from 0% to 100% of their densities. The results are shown in figure 1(e). The error was, in all cases, less than 1%, although slightly greater error was produced toward the lower end of the intersection spectrum. We hypothesize that this is because the lower end is closer to the "average" dot product for two randomly generated 10% dense vectors, and is therefore less distinguishable. On another interesting note, when there was zero intersection between the two vectors, the approximation function sometimes produced a negative estimate. On the whole, though, the function produced exceptional accuracy, no matter what the actual intersection was.

3.2 Efficiency

We also demonstrate the efficiency of the algorithm. An experiment was carried out on a 2.8GHz dual core AMD Athlon machine with 2 GB of RAM to determine the run time efficiency of the approximation algorithm versus the run

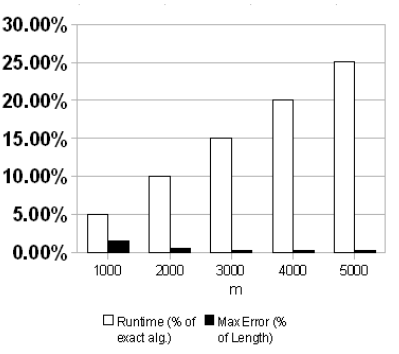


Fig. 2. Expected Run Time vs. Max Error

time efficiency of the exact algorithm. On two vectors of 20000 elements, whose density was 10%, (that is each vector contained 2000 1's) and whose intersection was 50% of the density (that is, 1000), the exact version took 27 minutes and 51 seconds to run. On the same vectors, the approximation algorithm with bloom filter parameters $m = 3000$, $k = 2$, ran in only 4 minutes and 4 seconds.

We also compared the relative run times of the public key encryption used versus the hash functions used in the bloom filter. On average, a single Paillier encryption (t_e) took 82.454ms, a Paillier decryption (t_d) took 81.583ms, and the hash function computation (t_h) was 0.00418ms. Since, with the approximation algorithm, we need only run m public key encryptions instead of the full ℓ public key encryptions, we can account for the vast majority of the difference in the algorithms' run times. In fact, the approximate runtime of the approximate algorithm is given by $t_e m + t_d + t_h \ell \cdot d \cdot k$. By comparison, the runtime of the exact algorithm is given by $t_e \ell + t_d$. Since $m < \ell$, usually significantly (80-90%), and $t_h << t_e$, the runtime of the approximate algorithm is significantly lower than the exact algorithm. Figure 2 is a chart of the expected run times (relative to the exact algorithm) of the previous experiments varying m, together with the maximum error observed in any run with that given m (relative to the vector length), in order to show the trade-off between accuracy and efficiency. This clearly demonstrates the great accuracy/efficiency trade-off of our algorithm.

4 Privacy-Preserving Association Rule Mining

We now show how the protocol described above would affect the process of privacy-preserving assocation rules mining. We assume that the data is vertically partitioned – each site holds some attributes of each transaction, and the sites wish to collaborate to identify globally valid association rules. However, the sites must not reveal individual transaction data.

The association rule mining problem can be formally stated as follows[17]: Let $\mathcal{I} = \{i_1, i_2, \cdots, i_m\}$ be a set of literals, called items. Let $\mathcal{D}$ be a set of transactions, where each transaction T is a set of items such that $T \subseteq \mathcal{I}$. Associated

with each transaction is a unique identifier, called its *TID*. We say that a transaction T *contains* X, a set of some items in $\mathcal{I}$, if $X \subseteq T$. An *association rule* is an implication of the form $X \Rightarrow Y$, where $X \subset \mathcal{I}$, $Y \subset \mathcal{I}$, and $X \cap Y = \emptyset$. The rule $X \Rightarrow Y$ holds in the transaction set $\mathcal{D}$ with **confidence** c if c% of transactions in $\mathcal{D}$ that contain X also contain Y. The rule $X \Rightarrow Y$ has **support** s in $\mathcal{D}$ if s% of the transactions in $\mathcal{D}$ contain $X \cup Y$. Vaidya and Clifton[6] show that this problem can be reduced to the simple calculation of a zero-one dot product of vectors whose length is the size of the transaction set.

Our experimental results show that the Bloom filter approximation method should apply to association rule mining very well. In terms of accuracy, the Bloom filter approximation method is extremely good for sparse vectors, which tends to be the norm in association rule mining. In addition, the filter length can be increased more gradually for larger vectors, allowing the algorithm to scale better than linearly in terms of efficiency.

As an example, consider an association rule mining whose required confidence is 70%, and support is 7%. Using our protocol, assuming 10% of the sample had entries for a given item on either side, and based on figure 1(e), the absolute least confidence that a selected rule could have is 65%, and the absolute most that an unselected rule could have is 75%, and with a high probability, most selected would have at least 68% and most not selected would have at most 72%. The number of false positives and false negatives, therefore, would be quite small, even given large data compression.

5 Conclusion

The main contribution of paper is an efficient and provably secure protocol that approximately computes the scalar (dot) product of two vectors based on Bloom Filters. We show how this can be used to mine association rules in a privacy-preserving way. Our experimental results show that protocol is highly efficient, accurate and easily applicable to association rule mining. The dot product is actually useful in many different privacy-preserving data mining protocols beyond association rule mining. In the future, we intend to explore our protocol's applicability in other tasks. We also plan on extending it to work for more than two parties. Other dimension reduction techniques could also be used to approximate the vector space. Using these, the scalar product could also be computed. Different techniques may have different trade offs of security, efficiency and accuracy. In the future, we plan to look at these alternatives such as using space filling curves or techniques utilizing the Johnson-Lindenstruass theorem.

References

1. Vaidya, J., Clifton, C., Zhu, M.: Privacy-Preserving Data Mining, 1st edn. Advances in Information Security, vol. 19. Springer, Heidelberg (2005)
2. Agrawal, R., Srikant, R.: Privacy-preserving data mining. In: Proceedings of the 2000 ACM SIGMOD Conference on Management of Data, Dallas, TX, May 14-19, pp. 439–450. ACM, New York (2000)

3. Lindell, Y., Pinkas, B.: Privacy preserving data mining. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 36–54. Springer, Heidelberg (2000)
4. Kargupta, H., Datta, S., Wang, Q., Sivakumar, K.: On the privacy preserving properties of random data perturbation techniques. In: Proceedings of the Third IEEE International Conference on Data Mining (ICDM 2003), November 19-22, IEEE Computer Society, Los Alamitos (2003)
5. Huang, Z., Du, W., Chen, B.: Deriving private information from randomized data. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, June 13-16. ACM, Baltimore (2005)
6. Vaidya, J., Clifton, C.: Privacy preserving association rule mining in vertically partitioned data. In: The Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, July 23-26, pp. 639–644. ACM, Alberta (2002)
7. Kantarcıoğlu, M., Clifton, C.: Privacy-preserving distributed mining of association rules on horizontally partitioned data. IEEE Transactions on Knowledge and Data Engineering 16(9), 1026–1037 (2004)
8. Du, W., Atallah, M.J.: Privacy-preserving statistical analysis. In: Proceeding of the 17th Annual Computer Security Applications Conference, New Orleans, Louisiana, USA, December 10-14 (2001)
9. Ioannidis, I., Grama, A., Atallah, M.: A secure protocol for computing dot-products in clustered and distributed environments. In: The 2002 International Conference on Parallel Processing, Vancouver, British Columbia, August 18-21 (2002)
10. Goethals, B., Laur, S., Lipmaa, H., Mielikäinen, T.: On Private Scalar Product Computation for Privacy-Preserving Data Mining. In: Park, C.-s., Chee, S. (eds.) ICISC 2004. LNCS, vol. 3506, pp. 104–120. Springer, Heidelberg (2005)
11. Ravikumar, P., Cohen, W.W., Fienberg, S.E.: A secure protocol for computing string distance metrics. In: Proc. the Workshop on Privacy and Security Aspects of Data Mining at the Int. Conf. on Data Mining, pp. 40–46 (2004)
12. Qiu, L., Li, Y., Wu, X.: Preserving privacy in association rule mining with bloom filters. J. Intell. Inf. Syst. 29(3), 253–278 (2007)
13. Vaidya, J., Clifton, C.: Secure set intersection cardinality with application to association rule mining. Journal of Computer Security 13(4), 593–622 (2005)
14. Mitzenmacher, A.B.M.: Network applications of bloom filters: A survey. Internet Mathematics 1(4), 485–509 (2005)
15. Goldreich, O.: General Cryptographic Protocols. In: The Foundations of Cryptography, vol. 2, pp. 599–764. Cambridge University Press, Cambridge (2004)
16. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1996)
17. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: Buneman, P., Jajodia, S. (eds.) Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C, May 26–28, pp. 207–216 (1993)

Information Extraction from Thai Text with Unknown Phrase Boundaries

Peerasak Intarapaiboon, Ekawit Nantajeewarawat, and Thanaruk Theeramunkong

School of Information, Computer, and Communication Technology
Sirindhorn International Institute of Technology, Thammasat University, Thailand
{ipeerasak,ekawit,thanaruk}@siit.tu.ac.th

Abstract. Using sliding-window rule application and extraction filtering techniques, we propose a framework for extracting semantic frames from Thai textual phrases with unknown boundaries based on patterns of triggering terms. A supervised rule learning algorithm is used for constructing multi-slot extraction rules from hand-tagged training phrases. A filtering module is introduced for predicting rule application across phrase boundaries based on instantiation features of rule internal wildcards. The framework is applied to text documents in three domains with different target-phrase density and average lengths. The experimental results show that the filtering module improves precision and preserves high recall satisfactorily, yielding extraction performance comparable to frame extraction with manually identified phrase boundaries.

1 Introduction

By incorporation of ontology-based semantic annotations and appropriate extraction filtering techniques, it is expected that information extraction (IE) based on patterns of triggering class tags and triggering plain words can be realized for Thai text without deep sentence analysis. Initially inspired by the need to generate multi-slot semantic frames representing symptom descriptions from Thai text in a large-scale medical-related knowledge base construction project and subsequently by similar needs in other application domains, we aim to develop a framework for IE from Thai textual documents. A well-known supervised rule learning algorithm, called WHISK [2], is used as the core algorithm for constructing pattern-based IE rules automatically from a set of hand-tagged training phrases. Information sources of our target IE tasks are, however, collections of text paragraphs, called *information entries*, rather than collections of text portions identified beforehand as potential target phrases. Each information entry typically contains several target phrases along with other text portions. Locating potential target phrases in an information entry requires at least a shallow parser (chunk parser), which is not currently available for Thai text. A method, called *rule application using sliding windows (RAW)*, is introduced for applying IE rules to information entries without predetermining target-phrase boundaries.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 525–532, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Using sliding windows, IE rules are applied to text portions regardless of phrase boundaries and, therefore, tend to make many false-positive extractions. An extraction filtering module, called *wildcard-instantiation filtering (WIF)*, is employed for removal of incorrect extractions by employment of a binary classifier to predict rule application across a target-phrase boundary. Application of standard classification models, such as Naïve Bayes, k-Nearest Neighbor, Support Vector Machine, and Decision Tree, to such prediction is evaluated. The framework is applied to textual information entries in three different application domains with different target-phrase density and average lengths, i.e., medical-symptom descriptions, soccer match reports, and housing advertisements. Information entries in the first two domains are free text, i.e., grammatical plain text, while those in the third domain are semi-structured, i.e., a class of ungrammatical text that is often telegraphic in style but does not follow any rigid format. To begin with, Section 2 gives an overview of our framework. After recalling the basic idea of WHISK, Section 3 explains the RAW method. Section 4 describes WIF and Section 5 presents experimental results.

2 Framework Overview

Text paragraphs are taken as input *information entries* for our IE tasks. Word segmentation is applied to all information entries as part of a preprocessing step. A domain-specific ontology, along with a lexicon for it, is then employed to partially annotate word-segmented phrases with tags denoting the semantic classes of occurring words with respect to the lexicon. As outlined in Fig. 1, from manually identified target phrases in a training corpus, the WHISK algorithm is used for automatically constructing a set of IE rules. During rule learning, target-phrase lengths are observed when a rule makes correct extractions on the training instances and they are used for determining an appropriate window size for RAW. In order to prepare training data for construction of binary classifiers for predicting rule instantiation across target-phrase boundaries in the WIF module, the IE rules obtained from WHISK are applied using RAW to the

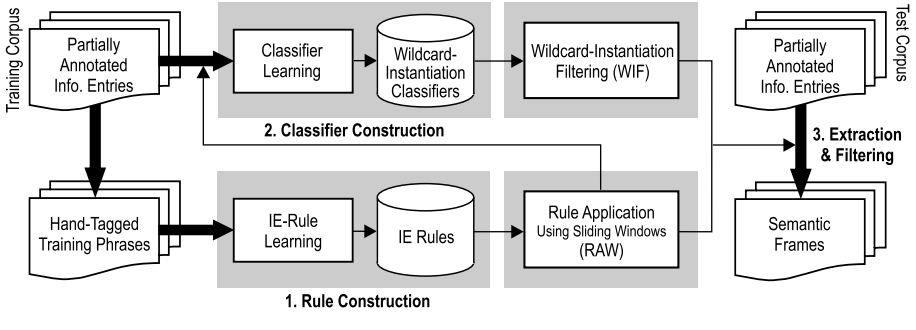


Fig. 1. An overview of the presented framework

เป็น|โรค|ที่|พบ|บ่อย|หลัง|จาก|เป็น|ไข้หวัด|~|ผู้ป่วย|ส่วนใหญ่|มัก|จะ|มี|[sec เสมหะ]|เป็น|[col สีเขียว]|~|มี|[sym อาการเจ็บ]|ที่|บริเวณ|[org หน้าอก]|อยู่|เป็น|เวลา|นาน|~|[ptime 6-12 วัน]|~|มี|[sym อาการไอ]|จน|เกิด|[sym อาการเจ็บ]|ที่|[org ชายโครง]|อยู่|นาน|~|[ptime 3-4 วัน]|~|ผู้ป่วย|อาจ|มี|สุขภาพ|ทั่วไป|แข็งแรง|...

Fig. 2. A portion of a partially annotated word-segmented information entry

Target phrase: |มี|[sym อาการเจ็บ]|ที่|บริเวณ|[org หน้าอก]|อยู่|เป็น|เวลา|นาน|~|[ptime 6-12 วัน]|
Extracted frame: {SYM [sym อาการเจ็บ]}{LOC [org หน้าอก]}{PER [ptime 6-12 วัน]}

Fig. 3. A target phrase and an extracted frame

partially annotated information entries in the training corpus. RAW and WIF are then used together to extract semantic frames from text in a test corpus.

Fig. 2 illustrates a portion of word-segmented and partially annotated information entry describing acute bronchitis, obtained from the text-preprocessing phase, where '|' indicates a word boundary, '~' signifies a space, and the tags "sec," "col," "sym," "org," and "ptime" denote the semantic classes "Secretion," "Color," "Symptom," "Organ," and "Time period," respectively, in our medical-symptom domain ontology. The portion contains three target symptom phrases, which are underlined in the figure. Fig. 3 shows the frame required to be extracted from the second underlined symptom phrase in Fig. 2. It contains three slots, i.e., SYM, LOC, and PER, which stand for "symptom," "location," and "period," respectively.

3 Rule Learning and Sliding-Window Rule Application

WHISK [2] induces rules top-down, starting from the most general rule that covers all training instances, and then specializing the initial rule by adding triggering terms one at a time in order to prevent rule application with incorrect extractions. Fig. 4 gives a typical example of a WHISK IE rule. Its pattern part contains three triggering class tags, one triggering plain word, and four instantiation wildcards. The three triggering class tags also serve as *slot markers*—the terms into which they are instantiated are taken as fillers of their respective slots in the resulting extracted frame. When instantiated into the target phrase in Fig. 3, this rule yields the extracted frame shown in the same figure.

A WHISK rule does not have ability to automatically segment a free-text information entry so that the rule can be applied to the relevant portion of text

Pattern: *(sym)*(org)*นาน*(ptime)
Output template: {SYM $1}{LOC $2}{PER $3}

Fig. 4. An IE rule example

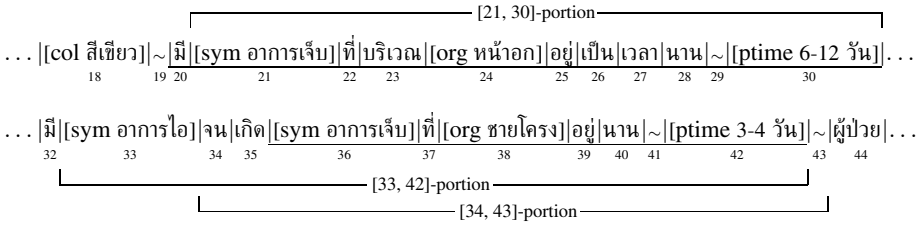


Fig. 5. Examples of text portions from which extractions are made using RAW

Table 1. Frames extracted from the text portions in Fig. 5 by the rule in Fig. 4

Portion	Extracted frame	Correctness
[21, 30]	{SYM [sym อาการเจ็บ]}{LOC [org หน้าอก]}{PER [ptime 6-12 วัน]}	Correct
[33, 42]	{SYM [sym อาการไอ]}{LOC [org ชายโครง]}{PER [ptime 3-4 วัน]}	Incorrect
[34, 43]	{SYM [sym อาการเจ็บ]}{LOC [org ชายโครง]}{PER [ptime 3-4 วัน]}	Correct

[1]. A sliding window technique is introduced to locate text portions for rule application—a method called *rule application using sliding windows (RAW)*. Using a k-word sliding window, a rule r is applied to each k-word portion of an information entry one-by-one sequentially. More precisely, assume that an information entry E consisting of n words is given and for any l, m such that $1 \leq l \leq m \leq n$, the $[l, m]$-*portion* of E is the portion beginning at the lth word position and ending at the mth word position of E. Then r is applied to the $[i, i + k - 1]$-portion of E for each i such that $1 \leq i \leq n - k + 1$. An application that results in a duplicated frame is discarded.

As illustrated in Fig. 5, when the rule in Fig. 4 is applied to the information entry in Fig. 2 using a 10-word sliding window, it makes extractions from the $[21, 30]$-portion, the $[33, 42]$-portion, and the $[34, 43]$-portion of the entry. Table 1 shows the resulting extracted frames. Only the extractions made from the first and third portions are correct. When the rule is applied to the second portion, the slot filler taken through the first slot marker of the rule, i.e., "sym," does not belong to the symptom phrase containing the filler taken through the second slot marker of it, i.e., "org," whence an incorrect extraction occurs.

4 Wildcard-Instantiation Filtering

WHISK learns an extraction pattern in terms of triggering terms for making an extraction from a "single" target phrase. Using RAW, however, an obtained IE rule may be instantiated across a target-phrase boundary (cf. Fig. 5 and Table 1), yielding an extracted frame containing unrelated slot fillers, which is definitely an incorrect extraction, i.e., a false positive. Instantiations of wildcards occurring between the first and the last slot markers of a rule, called *internal wildcards*, provide a clue to detect such an undesirable extraction—if an internal

Table 2. Examples of instantiation features for an internal wildcard

Portion	Instantiation of the 1st internal wildcard	No. of spaces	No. of plain words	Presence of "ที่"	Instantiation feature vector
$[21, 30]$	|ที่|บริเวณ|	0	2	1	$[0, 2, 1]$
$[33, 42]$	|จน|เกิด|[sym อาการเจ็บ]|	0	2	0	$[0, 2, 0]$
$[34, 43]$	|ที่|	0	1	1	$[0, 1, 1]$

wildcard of a rule is instantiated across a target-phrase boundary, then unrelated slot fillers are extracted by the resulting rule instantiation. Based on this idea, a filtering module, called *wildcard-instantiation filtering (WIF)*, is developed.

A binary classifier is constructed for each rule from observations obtained from applying the rule to textual information entries in the training corpus by assuming that rules obtained from WHISK are error-free with respect to their training target phrases. Under this assumption, an incorrect extraction in the training corpus implies the existence of an internal-wildcard instantiation across a target-phrase boundary. Given a text portion T and a rule r containing m internal wildcards $w_1, \ldots, w_m$, the *instantiation feature vector* for w_i $(1 \leq i \leq m)$ observed at T is a vector $\boldsymbol{f} = [f_1, \ldots, f_n]$ consisting of n features selected for representing the instantiation of w_i when r is applied to T, and the *wildcard-instantiation feature vector* for r observed at T is then defined as a vector $\boldsymbol{x} = \boldsymbol{f}_1 \parallel \boldsymbol{f}_2 \parallel \cdots \parallel \boldsymbol{f}_m$, where $\boldsymbol{f}_j$ $(1 \leq j \leq m)$ is the instantiation feature vector for w_j observed at T and '$\parallel$' denotes vector concatenation.

The rule in Fig. 4, for example, has three internal wildcards, i.e., those occurring between "sym" and "ptime" in its pattern. Table 2 illustrates the instantiation feature vectors for the first internal wildcard of this rule observed at the three text portions in Fig. 5 when the number of spaces, the number of plain words, and the presence of the specific plain word indicated in the title of the fifth column are selected as features. At the $[21, 30]$-portion, for example, the first internal wildcard is instantiated into a string consisting of no occurrence of space and two plain words, one of which is the indicated specific word, yielding $[0, 2, 1]$ as the resulting instantiation feature vector. Likewise, the instantiation feature vectors for this wildcard observed at the $[33, 42]$-portion and the $[34, 43]$-portion are $[0, 2, 0]$ and $[0, 1, 1]$, respectively.

5 Experimental Results

Data Sets, Output Templates, and Training Process The proposed framework is evaluated in three different domains of Thai text: *medical-symptom descriptions (MD)*, *soccer match reports (SR)*, and *housing advertisements (HA)*. An information entry in the MD domain contains free text describing various symptoms of a disease and that in the SR domain contains news-story-style free text reporting a soccer match in details. Text in the HA domain, by contrast, is semi-structured by nature. 115, 86, and 189 information entries with the average

Table 3. Output templates and their meanings

Type	Output Template	Meaning
MD1	{OBS O}{ATTR A}{PER T}	An abnormal characteristic A is found at an observed entity O for a time period of T.
MD2	{SYM S}{LOC P}{PER T}	A primitive named symptom S appears at a human-body part P for a time period of T.
SR	{PLY P}{ACT A}{TIME N}	A player P takes a game action A in the Nth minute.
HA	{AREA A}{BR N}{RR M}	A house of area size A with N bedrooms and M rest rooms

Table 4. Data set characteristics for each template type

Type	Data set	No. of distinct target phrases	Target-phrase length			No. of target phrases per info. entry			Ratio of words in all target phrases to all words
			Max.	Avg.	Min.	Max.	Avg.	Min.	
MD1	Training	90	11	3.5	2	7	3.6	1	0.207
MD1	Test	136	8	3.3	2	11	2.9	1	0.152
MD2	Training	80	15	4.1	2	3	1.4	0	0.096
MD2	Test	66	8	3.9	2	5	1.2	0	0.087
SR	Training	93	28	8.0	3	6	2.0	2	0.258
SR	Test	156	21	6.4	3	6	3.9	2	0.261
HA	Training	87	37	16.1	7	2	1.2	1	0.275
HA	Test	113	34	15.2	7	2	1.3	1	0.291

length of 45.0, 68.6, and 64.3 words per entry in the MD, SR, and HA domains, respectively, are used in our exploratory evaluation. The collected information entries are preprocessed using a word segmentation program and are then partially annotated with semantic class tags using predefined ontology lexicons.

Four types of target phrases are considered in our experiments: two of them are from the MD domain, referred to as *Type-MD1* and *Type-MD2*; one from the SR domain, referred to as *Type-SR*; and the other one from the HA domain, referred to as *Type-HA*. Table 3 gives the output-template forms for the four types along with their intended meanings. The slot PER in the Type-MD1 template as well as the slot TIME in the Type-SR template is optional. One of the slots LOC and PER, but not both, may be omitted in the Type-MD2 template. One arbitrary slot in the Type-HA template may be omitted.

For each template type, rules are created using WHISK by randomly selecting and adding hand-tagged target phrases as new training instances until the 10 most recently selected phrases cause no creation of any new rule. All remaining information entries are then used as test data. Table 4 shows the number of all distinct target phrases in the training data set and the test data set obtained accordingly for each template type, and characterizes the data sets in terms of target-phrase length and target-phrase density.

Using our implementation of WHISK, 15, 11, 8, and 9 rules, respectively, are automatically generated for Type-MD1, Type-MD2, Type-SR, and Type-HA. The

length of the longest target phrase observed when a rule yields correct extractions on its training set is taken as the base window size for the rule. By applying the obtained rules to the information entries in their respective training sets using RAW, wildcard instantiations are observed and used as training data for constructing classifiers for the WIF module. Two kinds of features are used for representing wildcard instantiations: first, the number of spaces, the number of plain words, and the number of annotated words occurring in a text portion into which a wildcard is instantiated; and secondly, the presence or absence of certain specific terms in such a text portion. For feature selection, the *information gain* calculated from the training data is used as a measure of the relevance of each feature. Top 10% of features ranked according to their information gain values are selected. The number of selected features accordingly obtained for different rules varies from 2 to 8, with an average of 4 selected features per rule. The number of space occurrences is the most commonly selected feature. The average numbers of specific words whose presence is used as a selected feature are 1.5, 3.8, and 2.6, respectively, in the MD, SR, and HA domains.

Experimental Results. The Weka machine learning suite is employed for classifier learning and evaluation, using its default parameters. Four standard models are used, i.e., Naïve Bayes (NB), k-Nearest Neighbor (kNN), Support Vector Machine (SVM) based on the RBF kernel, and Decision Tree (DT) using C4.5. 3NN performs slightly better than 1NN, 5NN, and 7NN in our training data sets and is chosen as a representative of kNN. Recall and precision are used as performance measures, where the former is the proportion of correct extractions to relevant target phrases and the latter is that of correct extractions to all obtained extractions. Table 5 shows the evaluation results obtained from RAW without any extraction filtering and RAW with WIF using NB, kNN, SVM, and DT classifiers, where 'R' and 'P' stand for recall and precision, which are given in percentage. Compared to the results obtained using RAW alone, each of the four classification models improves precision while satisfactorily preserving the recall value of RAW in every row of this table. None of the four classifiers performs obviously better than another one.

Table 6 presents the overall results obtained by doubling the window size previously used. Compared to Table 5, recall for Type-MD1, most phrase instances of which are short (cf. Table 4), increases only slightly; by contrast, for Type-SR and Type-HA, phrase instances of which are relatively long, obvious

Table 5. Evaluation results using the base window size

Type	RAW		NB		kNN		SVM		DT	
	R	P	R	P	R	P	R	P	R	P
MD1	85.0	76.4	84.4	97.1	83.8	97.1	83.1	97.8	83.8	97.8
MD2	100.0	37.5	100.0	98.6	100.0	98.6	100.0	98.6	100.0	94.5
SR	81.0	55.7	79.1	94.7	79.1	95.4	79.1	93.9	79.1	93.9
HA	82.0	40.7	79.1	90.2	79.1	84.6	79.1	89.4	80.6	86.2

Table 6. Recall improvement by doubling the window size

Type	RAW		NB		*k*NN		SVM		DT	
	R	P	R	P	R	P	R	P	R	P
MD1	88.1	60.0	83.8	97.8	85.0	97.2	85.0	97.8	86.9	97.2
MD2	100.0	26.3	100.0	95.8	100.0	94.5	100.0	94.5	100.0	90.8
SR	96.2	51.0	91.1	92.3	94.3	92.0	91.8	91.8	93.0	92.5
HA	92.8	35.6	87.1	85.8	87.1	85.2	87.1	86.4	89.9	83.9

improvements of recall are obtained. While *k*NN, SVM, and DT yield slightly higher recall than NB, all the four classifiers give similar precision.

To compare our framework to rule application with known target-phrase boundaries, we manually locate all target phrases in the test sets and apply the generated rules directly to these manually identified text portions. The recall values obtained from such known-boundary extraction for Type-MD1, Type-MD2, Type-SR, and Type-HA are 88.8, 100.0, 96.2, and 94.9, respectively, with the precision of 97.9, 100.0, 96.2, and 84.0, respectively. Compared to Tables 5 and 6, in the MD domain, where the average target-phrase length is small, the performance obtained when the base window size is used is already close to that of known-boundary extraction, and an extension of the window size does not obviously affect the extraction performance. However, in the SR and HA domains, where target phrases are longer, increasing the window size makes the extraction performance significantly closer to that of the known-boundary case.

6 Conclusions

To apply WHISK IE rules to textual paragraph-like information entries without predetermining target-phrase boundaries, rule application using sliding windows (RAW) is introduced. A filtering component is proposed for removal of false positives resulting from rule application across target-phrase boundaries (WIF). The experimental results show that this filtering method improves extraction precision while satisfactorily preserving the recall performance of sliding-window rule application. When the window size is sufficiently large, the extraction performance of the proposed framework appears to be fairly close to that of rule application with manually located target phrases.

Acknowledgement. This work was supported by the Thailand Research Fund (TRF), under Grant No. PHD/0056/2550 (TRF-RGJ) and No. BRG50800013.

References

1. Chang, C.-H., et al.: A Survey of Web Information Extraction Systems. IEEE Trans. Knowledge and Data Engineering 18, 1411–1428 (2006)
2. Soderland, S.: Learning Information Extraction Rules for Semi-Structured and Free Text. Machine Learning 34, 233–272 (1999)

A Corpus-Based Approach for Automatic Thai Unknown Word Recognition using Ensemble Learning Techniques

Jakkrit TeCho, Cholwich Nattee, and Thanaruk Theeramunkong

School of Information, Computer and Communication Technology,
Sirindhorn International Institute of Technology, Thammasat University
131 M.5 Tiwanont Rd., Bangkadi, Muang, Pathumthani, Thailand 12000
{jakkrit,cholwich,thanaruk}@siit.tu.ac.th

Abstract. This paper presents a corpus-based approach for automatic unknown word recognition in Thai. This approach applies an ensemble learning technique to generate a model for classifying unknown word candidates using features obtained from a corpus. We propose a technique called "group-based evaluation by ranking". It clusters the unknown word candidates into groups based on the occuring locations. The candidate with the highest accuracy is then identified as an unknown word. In this task, the number of positive instances is dominantly smaller than that of negative instances, forming an unbalanced data set. To improve the prediction accuracy, we apply a boosting technique with "voting under group-based evaluation by ranking". We have conducted experiments on real-world data to evaluate the performance of the proposed approach. The experiments compared the accuracy of our technique with an ordinary naïve Bayes technique. Our technique achieves the accuracy 90.93±0.50% when the first rank is selected and 97.90±0.26% when the candidates up to the tenth rank are considered. This is 6.79% to 8.45% improvement.

Keywords: Unknown word recognition; Word boundary detection; Data mining; Machine learning; Ensemble learning.

1 Introduction

Unknown word recognition plays an important role in Natural Language Processing (NLP) since a word is a fundamental unit of language that provides meaning. Most NLP applications need to identify words in sentences before further manipulation. Word recognition can be basically done by using a lexicon collected beforehand. The lexicon may include almost all widely-used words in the language. However, it is impossible to construct the lexicon that includes all available words. Therefore, a technique that efficiently identifies boundaries of unknown words in text would help increase the overall of word recognition system performance. Identifying unknown words is even more difficult in languages which have no explicit word boundary including Thai [1]. This is because

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 533–540, 2009.
© Springer-Verlag Berlin Heidelberg 2009

unknown words contain known substrings may interfere word segmentation process.

For unknown word recognition in Thai text, Kawtrakul et al. [2] have analyzed unknown words from a collection of randomly selected documents from Thai newspapers, scientific reports and non-fiction texts. They found that, on average, about 15% of words in Thai text are unknown. The result shows the necessity of an efficient unknown word recognition technique for Thai text. The process to recognize unknown word can be separated into two steps: (1) unknown word candidate generation, and (2) unknown word boundary identification. In this paper, unknown word candidates are firstly generated by detecting unregistered portions in the input text. Then, various combinations of character clusters surrounding the unregistered portion are constructed as a set of unknown word candidates. After that, boundaries of unknown words are identified by using a supervised learning algorithm with features based on statistics from the collected text. Based on our preliminary investigation, we however found that an ordinary learning algorithm is unable to work well on this problem since the number of positive and negative generated candidates are unbalanced in the training set. There are a few number of positive candidates comparing to the number of negative ones because various combinations of surrounding character clusters can be generated and most of them are negative.

We therefore propose a novel technique to solve the problem of unbalanced data set by grouping the candidates that are generated from the same unregistered portion into a group. Each group may contain almost 100 candidates but only one (or a few) candidate is labeled as positive. Candidates from all groups are then fed to the learning algorithm in order to generate a classifier. A candidate is classified as positive based on its strength to be positive comparing to the other candidate in the same group. Moreover, we present a technique improving the prediction of a classifier by applying a boosting technique with voting under group-based evaluation by ranking. This paper presents an approach using a different subset of training data with a same algorithm learning method i.e., naïve Bayes classifier as a base classifier.

2 Previous Works of Thai Unknown Word Recognition

The research on unknown word recognition in Thai language has not been widely conducted as in other languages. A few researches has studied this problem since the past decades. The unknown words are viewed as the problematic source in the NLP system. Kawtrakul et al. [2] used the combination of a statistical semantic segmentaion model and a set of context sensitive rules to detect unknown words. Charoenpornsawat et al. [1] considered the unknown word recognition as a classification problem. They proposed a feature-based approach to identify Thai unknown word boundaries. Features used in the approach are built from the specific information in context surrounding the target unknown words.

3 Thai Unknown Words with Word Segmentation Problems

Most word segmentation algorithms use a lexicon or dictionary to parse a text at the character level. Based on such algorithm, three possible results can be outputted when they face with an unknown word. The first one is to obtain one or more sequence of known words from the unknown (out-of-dictionary) word, for example, a compound word "มะม่วงอกร่อง" (meaning: a kind of mango) can be segmented into "มะม่วง" (mango), "อก" (meaning: breast), and "ร่อง" (meaning: crack). All of these subwords are found in the lexicon. This kind of unknown words is called *hidden* unknown word. The second one is to gain a sequence of unknown segments which all are undefined in the lexicon. For example, we cannot detect any subword from an out-of-dictionary word "วิสัญญี" (meaning: Anesthetic) since all of its substrings do not exist in the dictionary. The last one is to get a sequence of known words mixing with unknown segments. For instance, an unknown word "ลูคีเมีย" (meaning: leukemia) can be segmented into two portions: an unknown segment ("ลูคี") and a known word ("เมีย", meaning: wife).

The second and third types contain unknown segments that allow us to know the existence of the unknown word. In this paper, we only focus on the recognition of an unknown word in these two types, later called a *detectable* unknown word.

4 The Proposed Method

The proposed method contains three processes: (1) unregistered portion detection, (2) unknown word candidate generation and reduction, (3) unknown word identification and evaluation. The details of these processes are given in sequence.

4.1 Unregistered Portions Detection

Practically, it is difficult to define a *word* in the language that does not have explicit word boundary. To cope with ambiguities more efficiently, we apply the concept of Thai Character Cluster (TCC) [6]. TCC is an inseparable group of Thai characters based on the Thai writing system. Unlike word segmentation that usually includes ambiguity, segmenting a text into TCCs is easily done by applying a set of rules. These morphological rules guarantee grammatically correct for word boundaries in Thai writing system. Moreover, almost transliterated words in Thai is normally comprised of several short words. With this characteristic, TCC is suitable for unknown word detection approach, and improves the chance to discover a new word. Since a TCC is a group of characters smaller than a word, we therefore called a TCC that could not be found in a lexicon as an unregistered portion.

Like [7], this paper employs the combination of TCCs and the LEXiTRON dictionary [8] to facilitate word segmentation. In this work, the *longest word*

segmentation [9] is applied by considering of minimum number of unregistered portions that were produced from both left-to-right and right-to-left longest matching. Whenever the number of unregistered portions from both directions are the same the left-to-right longest matching is selected.

4.2 Unknown Word Candidate Generation and Reduction

Similar to [1], [2], a candidate is generated from merging $\pm K$ TCCS around the considering unregistered portion, and the portion itself. For example, the number of possible candidates is 100 when K is set to 9. Furthermore, we present two basic rules to filter out the generated candidates. Since it is clear that a Thai word cannot comprise of any special characters, we do not generate a candidate that includes such special characters. The second rule is to employ a set of marker words, such as "นอกจากนี้" (Besides this), "ประกอบด้วย" (comprised with) etc. These marker words are considered word separators.

4.3 Unknown Word Identification and Evaluation

The existing works on Thai unknown word recognition [1, 2, 4] consider unknown word candidates separately. Each candidate is independently labeled as a positive or negative instance. However, several candidates can be generated from an unregistered portion, and typically only one candidate is an actual unknown word. We therefore put multiple candidates generated from the same portion into a group, and only the most likely candidate is classified as a positive instance.

Features Extraction. We use a sequence of TCCs instead of a sequence of characters to denote an unknown word candidate. Then, statistical-based features for each unknown word candidate are computed. In order to fasten the process of n-gram calculation, we apply the algorithm in [12] that utilizes the sorted sistring to enable the simultaneous calculation of (n-gram of) alphabetical sorted TCCs of a text string. For each sistring (i.e., unknown word candidate), eight types of features (F1-F8) are extracted as follows: *(F1) Number of TCC*: A technical words in Thai language tends to be long. Therefore, the number of TCCs in a word is an important factor to discriminate unknown words; *(F2) Number of known words*: Like several languages, some unknown words in Thai language can be viewed as a compound word that contains a number of known words. Therefore, the number of known words in a candidate can be used as a clue to identify whether the candidate is an unknown word or not; *(F3) TCC varieties on the left and right-hand side*: Since an isolated word is often used together with various words, the high varieties of distinct TCCs on the left and right-hand side of the considering candidate indicate that the candidate is likely to be a word; *(F4) Probability of special character on the left and right-hand side*: The probabiblity that special characters occur on the left and right-hand side of the considering candidate indicates that the candidate is located near delimiters and likely to be a word; *(F5) Number of Characters*: A long candidate

is likely to be an unknown word; *(F6) Frequency*: It is obvious that the number of occurrences of the word sistring is higher than that of non-word sistring; *(F7) IDF*: An unknown word is generally a word that does not happen frequently in several documents, but it appears frequently in only some specific documents; and *(F8) TFIDF*: A weight is often used in information retrieval and text mining to measure the importance of a word in a corpus. The importance increases proportionally to the number of times a word appears in the document but is offseted by the frequency of the word in the corpus.

Classification Techniques From the features extracted from the training corpus, three techniques (T1-T3) are applied to learn classifier for identifying unknown word boundaries.

(T1) Base Classifier : naïve Bayes classifier is a machine learning technique that generates a classifier based on Bayes's theorem with independence assumptions.

$$\hat{c} = \underset{c}{\operatorname{argmax}} P(C = c) \prod_{i=1}^{p} P(F_i = f_i | C = c) \tag{1}$$

where $\hat{c}$ is the predicted class, $P(C = c)$ is the prior probability that the class is c, and $P(F_i = f_i|C = c)$ is the probability that the feature F_i has a value f_i when the class is c. For continuous attributes, Gaussian distribution with smoothing technique can be applied as

$$P(x) = \epsilon + \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2}$$

where ϵ is a small positive constant that is set to 0.000001 in the experiments, μ is the mean, and σ is the standard deviation.

(T2) Group-based Evaluation by Ranking : Unlike the evaluation model in normal classifiers, we propose a technique called "group-based evaluation by ranking (GER)". It categorizes all candidates produced by the same unregistered portion location into the same group. This technique ranks all candidates with respect to their group based on its probability to be the positive class and select the top-k candidates of that group as the prediction for unknown words.

$$C_j = \underset{x \in G_j}{\arg\max_k} \left(\mathrm{P}(c|x)\right) \tag{3}$$

where C_j is the set of top-k candidates of j-th group when k is a number of preferred answer. G_j is a candidate group which were generated from an unregistered portion location j, $P(c|x)$ is the probability that the class is c when the candidate is x.

Used as a measurement, we define the equation for calculating the accuracy as follows:

$$ACC_k = \frac{|\{j \mid A(G_i) \cap P(G_i, k) \not= \emptyset,\ 1 \leq j \leq N\}|}{N} \tag{4}$$

where ACC_k is the accuracy when top-k ranking is considered, N is the number of detected unregistered portions, each of which forms a group. The set of correct answers for group j is defined as $A(G_j) = \{x_{ij} \mid x_{ij}$ is the i-th potential unknown word in the j-th group$\}$. The set of k ranked answers from the system for group j is defined as $P(G_j, k) = \{y_{ij} \mid y_{ij}$ is the i-th candidates ranked by the system for the j-th group when $i \leq k\}$.

(T3) Voting under Group-based Evaluation by Ranking : For additional improvement of the prediction of a classifier, we propose a technique called "voting under group-based evaluation by ranking (V-GER)". Unlike the voting technique in ordinal ensemble methods, this technique ranks all candidates belonging to the group based on the summation of its probability to be the positive class on each sub-ensemble and select the top-k candidates of that group as the prediction for unknown words.

$$C_j = \underset{x \in G_j}{\arg\max}_k \left(\sum_{i=1}^{N} (\alpha_i \mathrm{P}_{M_i}(c|x)) \right) \tag{5}$$

where C_j is the set of top-k candidates when k is a number of preferred answer, G_j is a candidate group which were generated from an unregistered portion location j, α_i is a weight of i-th sub-ensemble, $P_{M_i}(c|x)$ is the probability that the class is c in the i-th sub-ensemble when the candidate is x and i is number of sub-ensemble.

5 Experimental Results

In the experiment, we occupy a corpus of 16,703 medical-related documents gathered from WWW [13] with a size of 8.4 MB for evaluation. The corpus is first preprocessed by removing HTML tags and all undesirable punctuations. To construct a set of features, we apply TCCs and the sorted sistring technique. After applying word segmentation on the running text, we have detected 55,127 unregistered portions. Based on these unregistered portions, 3,209,306 unknown word candidates are generated according to the process shown in Sect. 4.2. Moreover, these 55,127 unregistered portions came from only 3,763 distinct words. In practice, each group of candidates may contain one or two positive labels. Therefore, 62,489 unknown candidates were assigned as positive and 3,146,819 unknown candidates were assigned as negative. The average number of unknown candidates in a group is around 58.

Based on preliminarily analysis of the Thai lexicon, we found that the average number of TCCs in a word is around 4.5. To generate unknown word candidates, we set K, the maximum number of TCCs surrounding an unregistered portion, to nine in our experiment. This number is twice of the average number of TCCs in a word. With K=9, the number of generated unknown word candidates is 100. Naïve Bayes classifier is used as the base classifier. The experiments were conducted using 10-fold cross validation. To identify an unknown word, we employed

Table 1. Comparing accuracy among three alternative settings of naïve Bayes classifier.

Rank	GER Method	V-GER Method
1	84.14±0.19	90.93±0.50
2	91.71±0.22	94.92±0.43
3	93.52±0.33	96.05±0.42
4	94.86±0.25	96.63±0.43
5	95.74±0.24	97.04±0.43
6	96.29±0.20	97.27±0.40
7	96.60±0.17	97.49±0.36
8	96.83±0.19	97.65±0.34
9	97.04±0.22	97.79±0.31
10	97.25±0.17	97.90±0.26
Original method	82.48±0.12	

our proposed method (group-based evaluation by ranking). We investigate the performance of our method when the top-k candidates are considered as correct answers. Here, the k ranges from 1 to 10. The experimental results showed that the classifier gained the accuracy of 84.14%±0.19 to 97.25%±0.17 as shown in Table 1.

To further improve the prediction accuracy, we employed our proposed voting under group-based evaluation by ranking technique by using 10 committees. The average accuracy with standard deviation are shown in Table 1. The results showed that the proposed ensemble method can achieve an increase in accuracy of 6.79% to 8.45% at the first rank when compared to the original and GER technique.

6 Conclusion

In this paper, we presented an automated method to recognize unknown words from a Thai running text. We described how to map the problem to a classification task. The naïve Bayes with a smoothing technique classifier is investigated using eight features for evaluation the model. In practice, the unknown word candidates actually have relationship among them. To reduce the complexity in unknown word boundary identification, reduction approaches are employed to decrease a number of generated unknown word candidates to 49%. This paper also proposed the group-based evaluation by ranking technique. This technique considered the unknown word candidates as groups that can solved the unbalanced data sets problem. To further improve the prediction of a classifier, we apply a boosting technique with voting under group-based evaluation by ranking. We have conducted a set of experiments on real-world data to evaluate the performance of the proposed approach. From the experiment results, the proposed technique achieves the accuracy of the order of 90.93%±0.50 to 97.90%±0.26 at the first rank and tenth rank. Our proposed ensemble method can achieve an increase in classification accuracy of the order of 6.79% to 8.45% at the first

rank when compared to the original naïve Bayes evaluation and GER technique, respectively.

Acknowledgment

This work has been supported by The Thailand Research Fund (TRF) of Thailand via reseach grant BRG5080013. The authors would like to thank all members at KINDML laboratory at Sirindhorn International Institute of Technology for fruitful discussions and comments.

References

1. Charoenpornsawat, P., et al.: Feature-based thai unknown word boundary identification using winnow. In: Proc. of the APCCAS 1998, Chiang Mai, Thailand, pp. 547–550 (November 1998)
2. Kawtrakul, A., et al.: Automatic thai unknown word recognition. In: Proc. of the NLPRS 1997, Phuket, Thailand, pp. 341–346 (October 1997)
3. Zhang, J., Mani, I.: Knn approach to unbalanced data distributions: A case study involving information extraction. In: Proc. of the ICML 2003, Washigton DC, USA (2003)
4. Haruechaiyasak, C., et al.: A collaborative framework for collecting thai unknown words from the web. In: Proc. of the COLING/ACL 2006, Sydney, Australia, pp. 345–352 (July 2006)
5. Freund, Y., Schapire, R.E.: A short introduction to boosting. Journal of Japanese Society for Artificial Intelligence 14(5), 771–780 (1999)
6. Theeramunkong, T., Tanhermhong, T.: Pattern-based features vs. statistical-based features in decision trees for word segmentation. IEICE Transactions on Information and Systems E87(D(5)), 1254–1260 (2004)
7. TeCho, J., et al.: A corpus-based approach for keyword identification using supervised learning techniques. In: Proc. of the ECTI-CON 2008, Krabi, Thailand, pp. 33–36 (May 2008)
8. Lexitron version 2.2. Thai-English, English-Thai Dictionary (2008)
9. Poowarawan, Y.: Dictionary-based thai syllable separation. In: Proc. of the EECON 1986 (1986)
10. Dietterich, T.G.: Ensemble methods in machine learning. In: Kittle, J., Roli, F. (eds.) Multiple Classifiers Systems, pp. 1–15. Springer, Heidelberg (2000)
11. Kotsiantis, S., Pintelas, P.: Combining bagging and boosting. Journal of Computational Intelligence 1(4), 324–333 (2004)
12. Nagao, M., Mori, S.: A new method of n-gram statistics for large number of n and automatic extraction of words and phrases from large text data of japanese. In: Proc. of the COLING 1994, Kyoto, Japan, pp. 611–615 (August 1994)
13. Theeramunkong, T., et al.: A framework for constructing a thai medical knowledge base. In: Proc. of the KICSS 2007, JAIST, Ishikawa, Japan, pp. 45–50 (November 2007)

A Hybrid Approach to Improve Bilingual Multiword Expression Extraction

Jianyong Duan[1], Mei Zhang[2], Lijing Tong[1], and Feng Guo[1]

[1] College of Information Engineering
[2] College of Art Design
North China University of Technology, Beijing, 100141, P.R. China
duanjy@hotmail.com

Abstract. We propose a hybrid approach for bilingual multiword expression extraction. There are two phases in the extraction process. In the first phase, lots of candidates are extracted from the corpus by statistic methods. The algorithm of multiple sequence alignment is sensitive to the flexible multiword. In the second phase, error-driven rules and patterns are extracted from corpus. These trained rules are used to filter the candidates. Some related experiments are designed for achieving the best performance because there are lots of parameters in this system. Experimental results showed our approach gains good performance.

1 Introduction

Bilingual multiword expression (BMWE) is an important linguistic resource. For example, when a selected word can be recognized as part of a multiword expression, it can give the appropriate translation for that expression. In this sense, a multiword expression refers in principle to word sequences which cannot be translated literally word-for-word. However the automatic technique studied for extracting MWEs does not necessarily follow any definition rigorously[2]. High qualified BMWEs will benefit some important tasks, such as machine translation[1,13], cross-linguistic information retrieval[8,12], and knowledge base construction[16], etc.

Many efforts have been devoted to extracting these BMWEs[4,5]. Some researchers detect multiwords in bilingual dictionaries because most of them are presented without being distinguished from free combinations of words[11]. Some systems rely on syntactic criteria and only extract compact bilingual multiwords[7,15]. Bilingual asymmetry information is also used in some tasks[9]. These bilingual Multiwords can typically not be aligned word-to-word, and cause a source-target and target-source asymmetry in the alignment matrix.

In this paper we apply a hybrid approach for BMWE extraction. Our system combines some techniques, including statistics, multi-sequence aligning, error-driven rules with active learning and bilingual comparison to extract MWEs from a Chinese-English bilingual corpus.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 541–547, 2009.
© Springer-Verlag Berlin Heidelberg 2009

2 Introduction of Multiple Sequence Alignment

The idea of multiple sequence alignment originates from gene comparison in the bioinformatics. Its aim is to find gene similarity. Those similar genes in DNA sequences are the clues of family relationships and evolutionary trees over different organisms or genes shared functions or properties. Although the essential of the multiple sequence alignment coincides with the edit distance which gives an indication of how close two strings are, this measure has been developed very well in bioinformatics. The Smith and Waterman algorithm is a classical measure for sequence alignment[18]. When the text string is viewed as common string, this algorithm is also effective. The case in point is the local sequence alignment for multiword expression extraction.

$$v[i][j] = \max\{a[i][j], b[i][j], c[i][j], 0\} \tag{1}$$

$$a[i][j] = \delta[i][j] + \max\begin{cases} a[i-1][j-1] \\ b[i-1][j-1] \\ c[i-1][j-1] \end{cases} \tag{2}$$

$$b[i][j] = \max\begin{cases} -(h+g) + a[i][j-1] \\ -g + b[i][j-1] \\ -(h+g) + c[i][j-1] \end{cases} \tag{3}$$

$$c[i][j] = \max\begin{cases} -(h+g) + a[i-1][j] \\ -(h+g) + b[i-1][j] \\ -g + c[i-1][j] \end{cases} \tag{4}$$

This algorithm focuses on the similarity of the local area between two sequences while it ignores their global similarity. Its detailed information has been addressed. Their model employs dynamic programming technique by three two-dimensional matrixes and affine gap penalty to evaluate the similarity between two sequences.

3 A Hybrid Approach

We design the Polyhymnia system to extract BMWEs. This system combines methods of frequency filtering, multi-sequence aligning, error-driven rules with active learning and bilingual comparison. In addition, the Chinese-English aligned corpus and phrase dictionary are also used.

3.1 Improvement for Multiple Sequence Alignment

The goal of the multiple sequence alignment is to find the similar segments among gene sequences. This algorithm has powerful ability for pattern similarity computation. The essential of this algorithm is the local alignment that seeks for the common segment in local areas. It coincides with the multiword

expression extraction. The difficulty of MWE mining comes from the flexible pattern, such as the multiword "根据 要求" may insert arbitrary words, "根据 上级 下达 的 要求" and "根据 教育部 的 要求", etc.

We improve this model by introducing the POS information, and the entry score $p[i][j]$ is computed by this way:

$$\delta[i][j] = \begin{cases} c1 \ s[i] = t[j], pos(s[i]) = pos(t[j]) \\ c2 \ s[i] \neq t[j], pos(s[i]) = pos(t[j]) \\ inc \ else \end{cases} \quad (c1 > c2 > 0, \ inc < 0) \qquad (5)$$

Where the $pos(s[i])$ denotes the POS tag of $s[i]$, and $pos(t[j])$ denotes the POS tag of $t[j]$. This equation gives different scores on the consideration of POS tag and word in the entry.

Pairwise sequence alignment can find the similarity sequences, but it also introduces some noises when similar segments are only extracted from two sequences. Multiple sequence alignment is an extension of pairwise alignment to incorporate more than two sequences into an alignment. Multiple alignment methods try to align all of the sequences in a specified set. Star alignment is performed among similar sequences. Given n sequences to be aligned and a threshold mw_thres. One sequence q_c is picked out as the "center" sequence. For each pair (q_i, q_c) determine an optimal alignment whose similar score is over mw_thres, then aggregate pairwise alignments.Alignments help the identification of common regions between the sequences.

In addition,another improvement is the threshold for a component word of candidate multiword. Although similar text sequences have been clustered into some groups, some words don't always occur in every sequence. To improve the robust of system, we give a threshold for these candidate component words. When the ratio of the occurrence is higher than threshold, it will be extracted out as the component word of multiword expression.

3.2 Error-Driven Rule Learning

Single Rule Template. The rule template is composed of two parts, a triggering environment as condition part and a rewrite rule as action part. The triggering environment is the context of multiword expression. The rewrite rule includes status transformation among candidate, correct, wrong. The template is:

If condition part ***then*** action part

We consider those MWEs with less than two interior gaps. For example, a multiword expression "和...一起去",

我/r	**和**/c	父亲/n	(**一起**/d	**去**/vd)	(看/v	雪/n)
Prev	***First*** ↑	*Mid*		***Second***↑		*Last*

We define its five parts as $Prev, First, Mid, Second$, and $Last$. $First$ and $second$ parts are multiword expression itself and $Prev, MidandLast$ parts are its context. The triggering environment should not contain all the five parts, but the $First$ part is necessary. Every part has more than one word. These words can be discerned by its index. For example, $Second$ part has two words. $Second(-2)$ denotes "一起" and $Second(-1)$ denotes "去". The minus denote counting from right side. After these definitions, condition part has these four types, Where part denotes one of these parts:

The Combination of Single Rule. The single template is the basic trigger. It cannot describe the complicate environment of multiword expression. We combine several single templates to form the complex templates which have powerful expression ability. For example, $First(-1, -2) = (n, n)$ and $Mid(-1) = v$. In our system, these combined templates have their linguistic meaning. Every MWE has a tag to denote its status.

4 Experimental Results

4.1 Resources

We develop bilingual MWE extraction system called Polyhymnia. Our corpora are bilingual aligned sentences, where 100,000 pairs of bilingual sentences are used for training process and 3,000 pairs are for test set. Bilingual aligned phrases are nearly 20,000 Chinese-English pairs. These equivalents are used to extract MWE patterns. Many useful rules learned from corpus.

The bilingual aligned sentences are preprocessed by part-of-speech tagging. Chinese POS tagging tool is developed by natural language computing lab of SJTU. English POS tagging tool is brill tagger. We also perfom the word stemming process for English sentences.

4.2 Results on Different Thresholds

Our system involves lots of parameters. Some parameters which have little influence on experimental results will be given fixed values. The parameters of pairwise sequence alignment follow experienced values. The crucial parameter for multiple sequence alignment is the threshold of sequence similarity mw_thres.

The Fig.1 shows precision and recall curves on condition of different mw_thres values. Some interesting results are observed from it. The precisions between Chinese and English are similar while their recalls have a nearly 10% gap. The reason is that Chinese multiword expressions are not in evidence. For example, the boundary between word and multiword expression is not clear. Chinese part-of-speech patterns are acquired difficultly than English patterns. In addition, Chinese word partition process also introduces certain error information.

It also shows that the mw_thres affects both precision and recall by different degrees. With the growth of mw_thres, the precision curves rise smoothly and

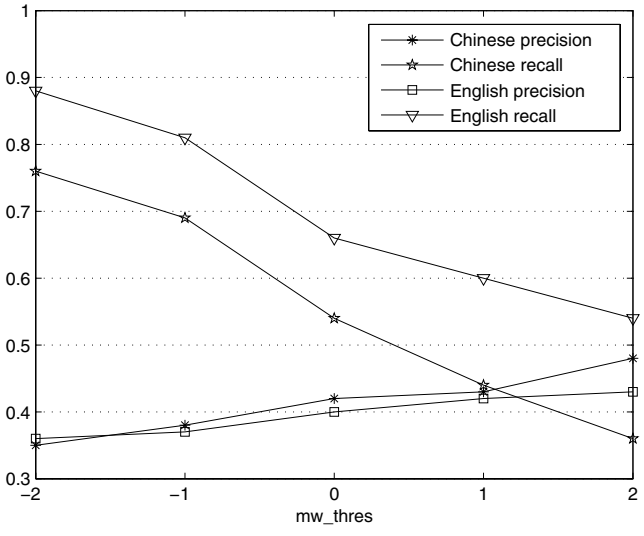


Fig. 1. Different threshold effects

the recall curves drop sharply. In the region of threshold *mw_thres* from -2 to 2, the recall drops nearly 40%. Lots of candidates massively distribute in this region. When the threshold increases, many multiword expressions are thrown away. It will lead to the insufficiency of training samples. The recall curves drop rapidly.

The lower precisions are also a concern. In the region of threshold *mw_thres* from -2 to 2, the precision curves improve only 10%. It origins the questioned sample quality. The samples are extracted by unsupervised approach. In one side, it saves the human labors. In the other side, it also introduces many noises. These samples also can be viewed as pseudo samples, which keep precisions on the lower levels. In addition, the number of position samples is fewer than the number of negative samples. For example, a majority of samples contain no multiword expression. The action of rules is mostly to change tags from candidate into error. The distribution difference of training samples can influence the precision.

5 Conclusion and Future Work

We extract BMWE from corpus by a hybrid approach. Some effective techniques are employed in this system. Every technique has its advantage. For example, the algorithm of multiple sequence alignment can extract similar patterns from corpus whatever the pattern are flexible or compact. Bilingual comparison can increase recall by its linguistic asymmetry because a multiword pattern in one language can provide additional information for its related MWE extraction in another language. In addition, the error-driven rule can introduce abundant context information. The advantage of error driven rule is that it can automated

learn the linguistic knowledge. Using the combined technique, we can fully exert the large data process of statistics and direct linguistic behavior understanding by rule providing.

Acknowledgements

This paper is supported by 2008 Scientific Research Platform and Group construction Funds of North China University of Technology(NCUT)–New Research Direction, and also supported by the special fund of NCUT for the Discipline and Graduate Education–Key Discipline–Applied Information Technology.

References

1. Bharati, A., Sangal, R., Mishra, D., Sriram, V., Papi Reddy, T.: Handling Multiword Expressions without Explicit Linguistic Rules in an MT System. In: International Conference on text, speech and dialog, Czech Republic (2004)
2. Gil, A., Dias, G.: Using Masks, Suffix Array-based Data Structures and Multidimensional Arrays to Compute Positional Ngram Statistics from Corpora. In: ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment (2003)
3. McCallumzy, A.K., Nigam, K.: Employing EM and Pool-Based Active Learning for Text Classification. In: Machine Learning: Proceedings of the Fifteenth International Conference, ICML 1998 (1998)
4. Atwell, C.O., Atwell, E., Souter, C.: Detecting Multiword Expressions Using Cross-Linguistic Corpora and Machine Learning Techniques. technical report (2004)
5. Boxing Chen, L.D.: Preparatory Work on Automatic Extraction of Bilingual Multi-Word Units from Parallel Corpora. Computational Linguistics and Chinese Language Processing 8(2), 77–92 (2003)
6. Brill, E.: Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. Computational Linguistics 21(4) (1995)
7. Wu, C.-C., Chang, J.S.: Bilingual Collocation Extraction Based on Syntactic and Statistical Analyses. Computational Linguistics and Chinese Language Processing 9(1), 1–20 (2004)
8. Evans, D.A., Zhai, C.: Noun-Phrase Analysis in Unrestricted Text for Information Retrieval. In: Proceedings of the 34th Annual Meeting of Association for Computational Linguistics, Santa Cruz, California (1996)
9. Dias, G., Gabrial, J., Lopes, P., Guillore, S.: Multilingual Aspects of Multiword Lexical Units. In: Proceedings of Workshop on Language Technologies of the 32th annual meeting fo the Societas Linguistica Europea (1999)
10. Guo, J.: Critical Tokenization and its Properties. Computational Linguistics 23(4) (1997)
11. Bentivogli, L., Pianta, E.: Detecting Hidden Multiwords in Bilingual Dictionaries. In: The Tenth EURALEX International Congress (2002)
12. Martinez, F., Martin, M.T., Rivas, V.M., Diáz, M.C., Urena, L.A.: Using Neural Networks for Multiword Recognition in IR. In: Proceedings of 2002 Conference of International Society of Knowledge Organization (ISKO) (2002)

13. Lambert, P., Banchs, R.: Grouping Multi-word Expressions According to Part-Of-Speech in Statistical Machine Translation. In: Proceedings of the EACL Workshop on Multi-word expressions in a multilingual context (2006)
14. Piao, S.S., Sun, G., Rayson, P., Yuan, Q.: Automatic Extraction of Chinese Multiword Expressions with a Statis-tical Tool. In: Proceedings of the EACL Workshop on Multi-word expressions in a multilingual context (2006)
15. Piao, S.S., Rayson, P., Archer, D., McEnery, T.: Comparing and combining a semantic tagger and a statistical tool for MWE extraction. Computer Speech and Language 19(4), 378–397 (2005)
16. Venkatsubramanyan, S., Jose, P.-C.: Multiword Expression Filtering for Building Knowledge Maps. In: Proceedings of the 2nd ACL Workshop on Multiword Expressions: Integrating Processing (MWE 2004), Barcelona, Spain (2004)
17. Simon Tong, D.K.: Support Vector Machine Active Learning with Applications to Text Classification. Journal of Machine Learning Research 2, 45–66 (2002)
18. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. J. Molecular Biology 147(1), 195–197 (1981)

Addressing the Variability of Natural Language Expression in Sentence Similarity with Semantic Structure of the Sentences

Palakorn Achananuparp, Xiaohua Hu, and Christopher C. Yang

College of Information Science and Technology
Drexel Univeristy, Philadelphia PA 19104
pkorn@drexel.edu, thu@cis.drexel.edu,
chris.yang@ischool.drexel.edu

Abstract. In this paper, we present a new approach that incorporates semantic structure of sentences, in a form of verb-argument structure, to measure semantic similarity between sentences. The variability of natural language expression makes it difficult for existing text similarity measures to accurately identify semantically similar sentences since sentences conveying the same fact or concept may be composed lexically and syntactically different. Inversely, sentences which are lexically common may not necessarily convey the same meaning. This poses a significant impact on many text mining applications' performance where sentence-level judgment is involved. The evaluation has shown that, by processing sentence at its semantic level, the performance of similarity measures is significantly improved.

Keywords: Sentence similarity, structural similarity, sentence semantics, verb-argument structure, semantic equivalence, textual entailment.

1 Introduction

A major issue that many text mining applications have to deal with is the variability of natural language expression. Due to flexibility of human language, the same information can be formulated in numerous ways. This has great implication in text summarization and question answering applications where the identification of redundant information is crucial to system performance. Another important issue pertains to the notion of similarity between sentences. Recent development has been made on this issue. In information retrieval field, different levels of topical similarity between sentences are proposed [12][15]. Relevant sentences either address the same specific topics or they might talk about the similar general topics. In natural language processing (NLP), two notions of sentence similarity, semantic equivalence and entailment [9], are understudied. Because they are strongly related, thus, a clear distinction is sometimes difficult to define. In general, two sentences are said to be semantically equivalent if they share the exact same meaning. Following this definition, paraphrase sentences are considered the most common form of semantic equivalency. On the other hand, entailment focuses on unidirectional inference. If the

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 548–555, 2009.
© Springer-Verlag Berlin Heidelberg 2009

meaning of one sentence can be inferred from the other sentence, the two sentences are said to be an entailment pair.

In this paper, we propose the method that takes into account semantic structure of sentences to measure sentences similarity. Traditionally, sentences are transformed into a bag of words for sentence similarity computation. This results in "semantic loss" because syntactic construction of the sentences is discarded. In particular, this has a crucial consequence to the identification of semantic equivalence or entailment sentences in which a specific inference has to be made. Our proposed method aims to deal with this issue by utilizing semantic roles of constituents in the sentences and computing sentence similarity at verb-argument structure level.

2 Related Work

The issue of measuring similarity of sentences is gaining more attention from various research communities. Although the perceived notions of text similarity may differ depending on the application domains, many of them share a common goal of matching up semantically similar sentences. To that end, various techniques have been proposed. First, probabilistic approaches have been adopted to identify topically related sentences [12] [15] [13] in sentence retrieval application. Next, several unsupervised approaches have been proposed for paraphrase recognition tasks [4][9][7] [14]. Recently, natural language processing community has increasingly focused on developing NLP systems to recognize entailment between sentences [8]. For this task, systems that employ extensive linguistic tools, such as logical inference engine and anaphora resolution [11][20], have started to show significant improvement in result over relatively "shallower" approaches. Nevertheless, this comes with a trade off in computational cost which makes comprehensive NLP systems currently impractical for a large text collection. Motivated by works in related areas [5][19], our work aims to address the shortcoming of existing text similarity measures when applying to sentence similarity task. That is, most techniques either simply treat sentences as single text unit or transform them into a bag of words representation; thus ignoring syntactic and semantic relations between constituents. Specifically, we are not aware of any methods that compute the similarity of sentence pairs at their semantic structure level.

3 Method

The semantic structure of sentences, referred to as *verb-argument structure*, encodes the relations between individual components and their semantic roles with respect to a given verb in a sentence. The labels of semantic roles are varied depending on the annotation scheme [3] [16]. Generally, *rel* defines a verb or relation between two or more arguments. Arg0 denotes a prototypical agent, Arg1 indicates a prototypical patient or theme of a given verb, and ArgM represents adjunctive argument (e.g. ArgM-LOC specifies location-related argument). For example, a simple sentence "John broke a glass" can be decomposed into one verb-argument structure [$_{\text{Arg0}}$ John]

[rel broke] [Arg1 a glass]. It consists of two arguments that correspond to a verb *broke*. *John* is labeled with Arg0 indicating that it is a prototypical agent and has a semantic role of *breaker* while *glass* is annotated with Arg1 denoting a prototypical patient, with *thing broken* as its semantic role. Notice that this is parallel to a grammatical relation where John is a subject and glass is a direct object of this sentence. By measuring semantic similarity of verb-argument structures, we can improve the effectiveness of sentence similarity measures despite the syntactic variability of language expression.

Based on the motivation that sentences that express the same meaning, in terms of event or idea, should share similar verb-argument structures, we decompose sentences into a set of verb-argument structures instead of comparing two unstructured text segments. Given two sentences s_i and s_j, the similarity score between verb-argument structures k of sentence s_i (v_{ik}) and verb-argument structure l of sentence s_j (v_{jl}) is determined by the similarity between their verbs ($sim_{rel}(v_{ik}, v_{jl})$) and the sum of similarities between the corresponding arguments ($sim_{arg_n}(v_{ik}, v_{jl})$). A scaling factor[1] $e^{-k \cdot score(v_{ik}, v_{jl})}$is applied to normalize similarity score to [0,1] in equation 2.

$$score(v_{ik}, v_{jl}) = \alpha \cdot sim_{rel}(v_{ik}, v_{jl}) + \beta \cdot \sum_{n} sim_{arg_n}(v_{ik}, v_{jl}) \tag{1}$$

$$sim_{va}(v_{ik}, v_{jl}) = \begin{cases} score(v_{ik}, v_{jl}) & \text{if } score(v_{ik}, v_{jl}) \leq 1 \\ e^{-k \cdot score(v_{ik}, v_{jl})} & \text{otherwise} \end{cases} \tag{2}$$

where α and β are coefficients that control the influence of semantic similarity between verbs and the sum of semantic similarity between arguments in verb-argument structures. The question is what is a reasonable weight for verb similarity and argument similarity components? Following a well-known psychological experiment of sentence sorting by [10] which suggests that verb is one of the main determinants of sentence meaning, we allocate higher importance factor to verb component than argument components[2]. For $sim_{rel}(v_{ik}, v_{jl})$, we employ WordNet-based gloss overlap measure [1] to compute word similarity score between verbs or verb phrases. To compute $sim_{arg_n}(v_{ik}, v_{jl})$, we treat argument text as phrase and utilize sentence-level similarity measures [2], e.g. Jaccard coefficient, phrasal overlap measure, etc., to determine their similarity.

$$sim(s_i, s_j) = \max_{k,l}[sim_{va}(v_{ik}, v_{jl})] \tag{3}$$

Finally, the similarity between sentence s_i and s_j is derived from verb-argument pair which produces the maximum similarity score as specified in equation 3. According to internal validation, deriving sentence similarity score by maximizing sim_{va} consistently produces better performance than linearly combining or averaging sim_{va}. This explains that sentence meaning tends to be dominated by one major verb-argument structure.

The whole computation process can be described as follows. First, we use semantic role labeler *SENNA* [6] to annotate each constituent in the sentences with their

[1] In this study, we consider k=0.05 as the optimal value.

[2] Based on our experiment, we find that weighting α and β at 0.5 gives the best result.

semantic roles. After the verb-argument structures are extracted by semantic role labeler, verb-argument text is parsed to identify the verb and individual arguments. During the parsing step, we apply syntactic rules to replace single verb with verb phrase if one exists in verb-argument text. In addition, we remove any words that are not part of the longest noun phrases in argument components. For example, given a verb-argument structure "[$_{Arg1}$ BBC] [$_{rel}$ stands] [$_{Arg2}$ for British Broadcasting Corporation]", a single verb "stands" will be expanded into a verb phrase "stands for". Arg1 text contains "BBC" and Arg2 text contains "British Broadcasting Corporation". Moreover, we perform noun denominalization on verb-argument structures of generic sentences, those which contain auxiliary verb, by expanding them into a new verb-argument structure. The expanded structure shares the same set of arguments as the original one while the auxiliary verb is replaced with a verb form of its forward adjacent noun. After that, the denominalized noun is removed from the corresponding argument. For instance, a verb-argument "[$_{Arg1}$ BBC] [$_{rel}$ is] [$_{Arg2}$ the abbreviation of British Broadcasting Corporation] will be expanded into "[$_{Arg1}$ BBC] [$_{rel}$ abbreviates] [$_{Arg2}$ of British Broadcasting Corporation].

After verb and argument components are identified, the next step is to determine the value of verb similarity component ($sim_{rel}(v_{ik}, v_{jl})$) and argument similarity component ($sim_{arg_n}(v_{ik}, v_{jl})$).WordNet-based gloss overlap measure is employed to compute word similarity score between verbs or verb phrases in $sim_{rel}(v_{ik}, v_{jl})$. Next, different sentence-level similarity measures are utilized to calculate $sim_{arg_n}(v_{ik}, v_{jl})$. For ArgM, we collapse all of its subtypes (e.g. ArgM-LOC, ArgM-TMP, ArgM-DIR, etc.) into single category and find its maximum score from all possible cross-argument pair comparison (ArgM vs. Arg0, ArgM vs. Arg1, etc.) This is done to deal with the cases where ArgM is not accurately tagged by semantic role labeler.

4 Experimental Evaluation

4.1 Data Sets

We use two publicly-available sentence pair data sets, Microsoft Research paraphrase corpus (MSRP) [9] and the third PASCAL recognising textual entailment challenge (RTE3) data set [8], to evaluate the performance of the similarity measures.

MSRP contains 5,801 sentence pairs (4,076 training pairs and 1,725 test pairs) automatically constructed from various web new sources. Each sentence pair is judged by two human assessors whether they are semantically equivalent or not. Semantically equivalent sentences may contain either identical information or the same information with minor differences in detail according to the principal agents and the associated actions in the sentences.

RTE3 consists of 800 sentence pairs from the development set and 800 sentence pairs from the test set. Each pair comprises two small text segments, which are referred to as *text* and *hypothesis*. Similarity judgment between sentence pairs is based

on directional inference between text and hypothesis. If the hypothesis can be entailed by the text, then that pair is considered to be a positive example.

4.2 Evaluation Setting

We define five evaluation metrics based on the general notion of positive and negative judgments in information retrieval and text classification as follows. *Recall* is a proportion of correctly predicted similar sentences compared to all similar sentences. *Precision* is a proportion of correctly predicted similar sentences compared to all predicted similar sentences. F_1 is a uniform harmonic mean of precision and recall. *Accuracy* is a proportion of all correctly predicted sentences compared to all sentences. A scoring threshold for positive pairs is defined at 0.5 as it is used in the literature [14].

Three baseline similarity measures are employed to compare the performance with the proposed method. These include Jaccard coefficient (sim_{jac}), Sumo-metric (sim_{sumo}) [7], and n-gram phrasal overlap measure ($sim_{overlap}$). For a detail description of these measures, please see [2]. We first compute baseline similarity scores between unstructured sentence pairs. After that, we generate structural similarity scores where each baseline sentence similarity measure is utilized in argument similarity (sim_{arg_n}) computation. For example, sim_{va}^{jac} denotes the structural similarity measure having Jaccard coefficient as the underlying method for computing argument similarity scores.

5 Results and Discussion

5.1 Experimental Results

Table 1. Comparison of the performance of structural similarity measures on paraphrase recognition task. Results with * indicate that the differences are not statistically significant compared to baseline.

Measure	Recall	Precision	F_1	Accuracy
sim_{va}^{jac}	0.9371	0.6806	0.7887	0.6701
sim_{va}^{sumo}	0.9305	0.6798	0.7856	0.6667
$sim_{va}^{overlap}$	0.9758	0.6753	**0.7982**	**0.6756***
Baseline				
sim_{jac}	0.6033	0.8347	0.7000	0.6568
sim_{sumo}	0.3967	0.5398	0.4872	0.4446
$sim_{overlap}$	0.8919	0.7001	0.7848	0.6748

Table 1 presents a comparison of structural similarity approaches to the traditional sentence similarity approaches. In this experiment, the best measures ($sim_{va}^{overlap}$)

metric performs slightly better than its baseline counterpart ($sim_{overlap}$). Comparisons between related measures (e.g. sim_{va}^{jac} and sim_{jac}, etc.) reveal that structural similarity approaches improve the performance of the corresponding sentence similarity measures. For example, sim_{jac} and sim_{sumo} perform significantly better (F_1 scores increase by 12.67% and 61.25%, respectively) when applying to verb-argument structures as opposed to unstructured sentences.

Table 2. Comparison of the performance of structural similarity measures on textual entailment recognition task. Results with * indicate that the differences are not statistically significant compared to baseline.

Measure	Recall	Precision	F_1	Accuracy
sim_{va}^{jac}	0.6724	0.5945	0.6310	**0.5911***
sim_{va}^{sumo}	0.7189	0.5424	0.6183	0.5413
$sim_{va}^{overlap}$	0.7734	0.5688	**0.6555**	0.5767
Baseline				
sim_{jac}	0.0512	0.6363	0.0948	0.4988
sim_{sumo}	0.2146	0.6069	0.3171	0.5263
$sim_{overlap}$	0.4561	0.6493	0.5358	0.5950

According to Table 2, the best performance is achieved by $sim_{va}^{overlap}$ at F_1 score of 0.6555 and sim_{va}^{jac} at accuracy level of 0.5911. A closer look at each related measures also shows a trend similar to paraphrase recognition task. The use of verb-argument structure has improved the performance of many naive sentence similarity measures. For instance, on F_1 metric, the performance of sim_{jac}, sim_{sumo}, and $sim_{overlap}$ has substantially increased by 560% (from 9.48% to 63.10%), 94.99% (from 31.71% to 61.83%), and 22.34% (from 53.58% to 65.55%), respectively. According to the result, we conclude that structural approach offers greater benefit to similarity computation of highly asymmetric sentences (those in RTE3 data) than those which are more symmetric in length.

5.2 Shallow vs. Deep Semantic Processing

The overall result in section 5.1 differs from that of text categorization task [19] where concept-based weighting has significantly improved classification performance over the traditional *tf-idf* scheme. One reason is that conceptual term frequency aims to capture the importance of a given concept in a document by leveraging the frequency of a concept in verb-argument structures. This approach is better suited for text categorization mechanism in which documents are classified according to their topics represented by terms or concepts in documents. On the other hand, the task of identifying semantic equivalence or entailment pairs requires a deeper understanding of sentence meaning. As shown in section 5.1, deeper semantic measures are able to recognize at least the same or greater number of positive pairs according to F_1 scores than those of vector space approach. The magnitude of improvement is even more

apparent in entailment task in which specific relations between constituents have to be identified.

5.3 The Advantage of Structural Approach over Linguistic Measures

Linguistic measures, those that employ natural language resources such as WordNet, has been proven to be highly effective in sentence similarity task [14]. However, a major criticism of such approaches is the lack of computational efficiency due to the exhaustive calculation of semantic similarity between word pairs. Therefore, they might not be as robust to employ in the real-world text mining applications as most naïve measures. In this regard, our approach offers a greater benefit over linguistic measures as it greatly improves the effectiveness of naïve measures while maintaining their computational efficiency, particularly at sentence processing time.

6 Conclusions

In this paper, we present the method that integrates semantic structure of the sentences to handle variability of natural language expression in sentence similarity task. Traditional similarity measures, which represent sentences as a bag of words, simply judge similarity between sentences according to their common word occurrences. However, due to the complexity of many text mining applications where the similarity judgment at semantic level is required, the performance of naïve measures are likely to degrade because of their disregard of syntactic construction. Our proposed method aims to address the issue by computing sentence similarity at verb-argument structure level. By annotating sentences with semantic roles, we can better perform similarity calculation between semantically related components. The evaluation results confirm that the inclusion of sentence semantics significantly improves the effectiveness sentence similarity tasks. Given the encouraging result, we plan to evaluate the effectiveness of the proposed measure in several application-specific contexts, such as text summarization and question answering.

Acknowledgments. This work is supported in part by NSF Career grant (NSF IIS 0448023), NSF CCF 0514679, PA Dept of Health Tobacco Settlement Formula Grant (No. 240205 and No. 240196) and PA Dept of Health Grant (No. 239667).

References

1. Achananuparp, P., Han, H., Nasraoui, O., Johnson, R.: Semantically enhanced user modeling. In: Proceedings of the 2007 ACM Symposium on Applied Computing, pp. 1335–1339. ACM Press, New York (2007)
2. Achananuparp, P., Hu, X., Xiajiong, X.: The evaluation of sentence similarity measures. In: Song, I.-Y., Eder, J., Nguyen, T.M. (eds.) DaWaK 2008. LNCS, vol. 5182, pp. 305–316. Springer, Heidelberg (2008)
3. Baker, C.F., Fillmore, C.J., Lowe, J.B.: The Berkeley FrameNet project. In: Proceedings of the COLING-ACL, Montreal, Canada (1998)

4. Barzilay, R., Elhadad, N.: Sentence Alignment for Monolingual Comparable Corpora. In: Proceedings of EMNLP, Sapporo, Japan, pp. 25–33 (2003)
5. Bilotti, M.W., Ogilvie, P., Callan, J., Nyberg, E.: Structured retrieval for question answering. In: Proceedings SIGIR 2007, pp. 351–358. ACM, New York (2007)
6. Collobert, R., Weston, J.: Fast Semantic Extraction Using a Novel Neural Network Architecture. In: Proceedings of ACL 2007, Prague, Czech Republic, June 23–30 (2007)
7. Cordeiro, J., Dias, G., Brazdil, P.: A Metric for Paraphrase Detection. In: Proceedings ICCGI 2007, p. 7. IEEE Computer Society, Washington (2007)
8. Dagan, I., Glickman, O., Magnini, B.: The PASCAL recognising textual entailment challenge. In: Proceedings of the PASCAL Workshop (2005)
9. Dolan, W., Quirk, C., Brockett, C.: Unsupervised construction of large paraphrase corpora: Exploiting massively parallel new sources. In: Proceedings of the 20th International Conference on Computational Linguistics (2004)
10. Healy, A., Miller, G.: The verb as the main determinant of sentence meaning. Pschonomic Science 20, 372 (1970)
11. Hickl, A., Bensley, J.: A Discourse Commitment-Based Framework for Reconizing Textual Entailment. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Prague, pp. 171–176 (2007)
12. Metzler, D., Bernstein, Y., Croft, W., Moffat, A., Zobel, J.: Similarity measures for tracking information flow. In: Proceedings of CIKM 2005, pp. 517–524 (2005)
13. Metzler, D., Dumais, S.T., Meek, C.: Similarity Measures for Short Segments of Text. In: Amati, G., Carpineto, C., Romano, G. (eds.) ECiR 2007. LNCS, vol. 4425, pp. 16–27. Springer, Heidelberg (2007)
14. Mihalcea, R., Corley, C., Strapparava, C.: Corpus-based and Knowledge-based Measures of Text Semantic Similarity. In: Proceedings of AAAI 2006, Boston (July 2006)
15. Murdock, V.: Aspects of sentence retrieval. Ph.D. Thesis, University of Massachusetts (2006)
16. Palmer, M., Gildea, D., Kingsbury, P.: The Proposition Bank: An Annotated Corpus of Semantic Roles. Comput. Linguist. 31(1), 71–106 (2005)
17. Park, Y., Byrd, R.J., Boguraev, B.K.: Automatic glossary extraction: beyond terminology identification. In: Proceedings of the 19th international Conference on Computational Linguistics, Taipei, Taiwan, August 24 - September 01, pp. 1–7 (2002)
18. Pradhan, S., Ward, W., Hacioglu, K., Martin, J.H., Jurafsky, D.: Shallow Semantic Parsing using Support Vector Machines. In: Proceedings of HLT/NAACL 2004, Boston, MA, May 2-7 (2004)
19. Shehata, S., Karray, F., Kamel, M.: A concept-based model for enhancing text categorization. In: Proceedings of KDD 2007, pp. 629–637. ACM, New York (2007)
20. Tatu, M., Moldovan, D.: COGEX at RTE3. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Prague, pp. 22–27 (2007)

Scalable Web Mining with Newistic

Ovidiu Dan[1] and Horatiu Mocian[2]

[1] INHOLLAND University, Diemen, The Netherlands
[2] Imperial College, London, United Kingdom

Abstract. Newistic is a web mining platform that collects and analyses documents crawled from the Internet. Although it currently processes news articles, it can be easily adapted to any other form of text. Data mining functions performed by the system are categorization, clustering and named entity extraction. The main design concern of the system is scalability, which is achieved by a modular architecture that allows multiple instances of the same component to be run in parallel.

This paper presents a novel algorithm for analysing web pages which tries to determine the title and text of a news item directly from the HTML code, discarding noise such as menus, ads, or copyright notices. Another contribution of this paper is the application of the Quality Threshold clustering algorithm for document clustering. Additionally, the algorithm has been optimized to increase its speed.

Keywords: Web mining, text mining, clustering, information extraction, quality threshold.

1 Introduction

In the last decade, more and more content has made the transition from print to online. As a result, more people are starting to turn to news aggregator websites to help them manage information overload. These systems continuously monitor Internet news sources and analyze news using text clustering, categorization, personalization and news ranking, which are research topics with deep roots in the academia. However, little is known about the internals of any of the commercial news gathering platforms, as there are no papers that describe their implementation in detail. Moreover, the number of academic projects that describe integrated systems combining two or more of the mentioned topics is limited. We try to address these problems by detailing the architecture of a web mining system in its entirety.

Newistic is a modular web mining platform that collects and analyzes news from Internet websites. This system was built to serve as a foundation for our data mining projects. A distributed and scalable design was needed to analyze the hundreds of thousand of news items published each day. First, a basic infrastructure was built, with the role of downloading, extracting and storing news and articles. On top of this were added other modules, such as clustering, categorization, or named entity extraction.

In addition to the modular architecture, the system presented in this paper brings two contributions: the implementation of an extraction algorithm that uses heuristic methods

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 556–563, 2009.
© Springer-Verlag Berlin Heidelberg 2009

to extract news from HTML pages, and the use of the Quality Threshold clustering algorithm [4] for documents. This algorithms, borrowed from gene expression analysis, is superior in quality to the k-means algorithm. Although it has quadratic complexity, this issues was addressed by optimizing it.

The remainder of the paper is organized as follows: In Section 2 we present other work related to this subject, Section 3 describes the architecture of the system which consists of the Crawler, Analyzer, Clustering, Storage and Web Server components, Section 4 presents the text mining functionalities, Section 5 discusses clustering, Section 6 details the results, and we wrap up with future work in Section 7.

2 Related Work

As we stated previously, there is almost no research published describing any of the commercial news aggregator engines. One exception is a paper on the personalization feature of Google News [14], which is only one module of the platform.

A number of news aggregation platforms have emerged in the academic world, but few of them are presented as a whole in research papers. Among the most recent systems is ComeToMyHead. The architecture and components of the system are described in [11] (here, the system is named Velthune). Its novel ranking algorithm, which takes into account the freshness of news articles and the rank of their source, is the first news ranking algorithm documented in a paper [12]. It gathers, categorizes and clusters news from more than 2000 sources using RSS and Atom. NewsBlaster [7] covers only 17 sources, but they provide summarization of text articles, along with hierarchical clustering. It was built by integrating the Topic Detection and Tracking (TDT) and summarization modules, which were researched independently before.

NewsInEssence [5], [6] established a framework for finding related articles for an input story, in real-time. They use in-site search provided by the source web sites to find possible candidates for a story, thus their crawling component is greatly simplified. The articles found are summarized. QCS [8] is a modular platform that provides querying, clustering and summarization of articles from an input set. Its modular structure allows combinations between different algorithms for the three types of operations available. It has no crawling component. NewsJunkie [9] filters news stories by taking into account their novelty and by tracking news previously viewed by the user. Thus, it is able to provide personalized news feeds.

European Media Monitor (EMM) is the most compelling research project to date. It gathers news in 35 languages, classifies them, extracts named entities and then generates cross-lingual clusters. The clustering is performed over multiple languages. NewsExplorer, the analysis component of EMM, is described in [13].

3 Architecture

The architecture of Newistic consists of several loosely coupled modules. These components can each have several instances running on multiple machines. Instances auto-detect each other and collaborate on the task at hand.

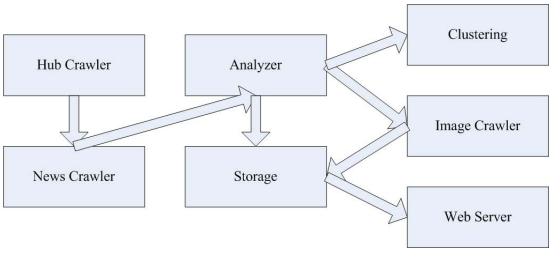


Fig. 1. Newistic's architecture

Each document goes through a processing pipeline that connects all the components of the system. Figure 1 shows the flow of data through the modules. The Hub Crawler periodically scans each Hub for new links. A Hub is a specific page from a website that links to the latest news. Any new link is possibly a fresh news story, so it is sent to the News Crawler. This module is responsible for downloading new links provided by the Hub Crawler and sending them further down the pipeline, to the Analyzer. The latter tries to extract the article from the HTML code. If successful, it detects the category of the news article and extracts Named Entities. If links to images are found in the HTML page, the links, together with the news story, are sent to the Image Crawler. Otherwise, the story is sent directly to the Storage module. The task of the Image Crawler is to download the images it receives, and store them locally. The Storage module indexes and stores the text. From the Storage module, the data is passed over to the Clustering module, which periodically builds an in-memory structure of the clusters.

The modular architecture of Newistic makes it very easy to distribute and, consequently, scale out. Almost all the backend modules can have multiples instances that run in parallel. The exceptions are the Storage component, which manages the document index, and the Clustering module, which cannot be run in parallel without rewriting its algorithm. However, the Analyzer, which is an essential and CPU-intensive component, was designed to be distributable from the start.

4 Text Analysis

Almost all the data mining functions are performed by the Analyzer. The sole exception, clustering, is presented in a separate section. The first mining function is extracting news articles out of HTML pages received as input. If successful, the output consists of the title and text of the article, along with metadata like category, named entities, or images related to it. After extracting a news article, the Analyzer extracts the named entities, which are divided in 3 categories: places, people and organizations. They can be used to influence the weights of words when performing text operations like categorization, clustering or search. For this task, the ANNIE component of the GATE framework [15] was used.

News Extraction. The News Extraction module is one of the most important parts of the system, as there are very few other systems, commercial or academic, that have this feature implemented.

Many news aggregators use RSS feeds as their source of information. Although this might be a better choice than developing a crawler and an extraction algorithm from scratch, it has two disadvantages:

1. Not all the news websites provide RSS feeds for their stories.
2. Some RSS feeds don't provide the whole text of the news.

Our algorithm extracts news directly from web pages, overcoming the disadvantages of using RSS feeds. It tries to determine the title and body, as well as links to images located in the article by analyzing the HTML of each candidate page. In order to come up with an efficient algorithm to extract news stories from raw HTML, an heuristic approach was used. Although it is impossible to cover all the different layouts that exist for web pages, the module manages to successfully analyze a fair share of them.

After cleaning out irrelevant parts such as JavaScript blocks or forms, contiguous blocks of text are extracted. Next, these blocks are analyzed to see if they are part of a possible news story. If a block has a high percentage of text within link anchors, it is discarded. Finally, each block of text is given a score that represents how likely it is to be the title of the article. This score is calculated taking into account 16 factors, including the position and length of the block in the text, and what HTML tags it is enclosed in. If the highest score passes a certain threshold, the block of text is taken as the title, and the following blocks of text are taken as the body of the article. The HTML extraction algorithm has some issues that need to be addressed in the future. One of them, for example, is the inability to treat a news story spanning multiple web pages as a single item. Another one is atypical HTML layouts that need to be analyzed individually to see if the algorithm can be particularized for each case.

Categorization. The algorithm that we used for categorization is K-Nearest Neighbor, or kNN [2]. In [3], Yang compared it with other methods: Support Vector Machines (SVM), Linear Least Squares Fit (LLSF), Neural Networks (NNet) and Naive Bayes (NB). kNN was the second most efficient only to SVM, which is more computation-intensive.

5 Clustering

The Clustering module groups related articles. It receives all the news items extracted by the Anazlyer, and clusters them periodically. Each new set of clusters is sent to the Web Server for processing. Additionally, this module is also responsible for detecting duplicate articles.

The clustering function is run at a fixed interval, currently set at 10 minutes. It clusters the news gathered by the system in the last 24 hours. This amounts to between 100,000 and 200,000 news.

The clustering algorithm that we implemented is called QT (Quality Threshold). It was proposed by Heyer et al. in [4] for gene clustering. From what we know, our implementation is the first one used for text clustering.

The following steps explain the QT clustering algorithm:

1. For a document D that we want to cluster, compute the cosine similarity between D and all the other documents;
2. All the documents that are closer than a certain threshold to D are added to a possible cluster with the center in D.
3. Repeat steps 1 and 2 for all the documents in the dataset. This results in a set of possible clusters. The number of possible clusters is equal to the number of documents in the dataset. Each document is the center of one of the clusters. One document can be assigned to multiple possible clusters.
4. Choose the cluster with the highest number of documents as an actual cluster, and remove its documents from the dataset.
5. Repeat steps 1 - 4 on the remaining dataset until there are no more documents to cluster or until all the possible clusters have a single document.

We added duplicate detection at the first step of the QT algorithm. At that point, documents that have a high similarity with D are considered duplicates, and marked accordingly.

QT derives its name from the threshold that was introduced at step 2. This threshold ensures that all the documents in one cluster have a certain similarity between them, allowing the clusters to satisfy a quality criteria. A low threshold results in clusters with less documents, but closely related. Increasing the threshold implies larger clusters, but less homogenous.

QT clustering has several advantages over the most widely used clustering algorithm, k-means [1]. First, all the clusters returned by QT pass the quality threshold. QT leaves the documents that don't fit in any cluster unclassified, while k-means must cluster all the documents in the dataset. Second, QT produces the same clusters every time it is run, while k-means may yield different results at each run. Third, QT does not need to know the number of clusters to be produced a priori.

The major disadvantage of QT clustering is its high processing capacity requirement. Having $O(n^2)$ complexity, QT is more computationally intensive than other clustering algorithms, specifically k-means. This makes it rather unfeasible for large input data.

However, our implementation of the algorithm includes some optimizations that makes it significantly less time consuming. Instead of computing the cosines between each document, only a small percentage of them are actually calculated. This is done by examining the words that any two documents have in common, and their weights. Additionally, we have managed to make a part of the algorithm incremental. The cosine similarity between two documents is not computed at each run, but instead is computed once and stored in a cache.

These optimizations allow QT clustering to scale to the number of news items that we currently process. But, in the future, if we want to increase the number of sources or the size of the covered time span, this implementation might reach its limits. A subject of future research is designing and developing a distributed algorithm for QT clustering.

6 Results

Although successful tests have been made with multiple instances of the same component run on different machines, the tests presented in this section were made using

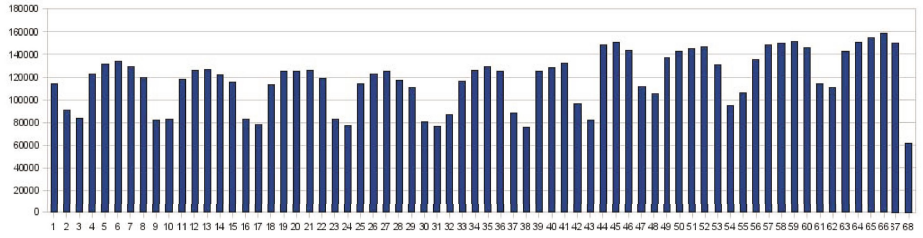


Fig. 2. News extracted over a 68-day period

one instance of each component. This decision was made due to the reduced number of machines available for testing.

6.1 General Results

The current implementation of the system uses 5000 English news websites as the Hub list. The system has been running for approximately 2 months. In this time, more than 8 million articles were gathered. Figure 2 shows the number of news articles found each day. It can be seen that the chart has a certain cyclicity, caused by the fact that less news are crawled in weekends than during weekdays. The average amount of news crawled each day is around 110,000, but in certain days it peaks at over 150,000.

From the chart, it can be observed that the system gathers between 4000 and 5000 news/hour, or an average of 1.25 items/sec. We will show that this value is limited by the number of sources available, and not by the throughput capacity of Newistic.

To obtain the throughput of the system, we conducted another test, in lab conditions, as opposed to the previous one, which was performed in real-life conditions. A random news article was chosen from a website and stored locally on the computer where the Hub Crawler component is installed. Afterwards, 5000 hubs were created artificially, each containing URLs that were different, but linked to the same article downloaded previously. This way, bandwidth and number of sources don't represent an issue. The system gathered around 18,000 news each hour, resulting in an average speed of 5 items/sec. This proves that currently the system is limited by the number of news sources, and not by its processing speed.

Table 1 shows some statistics for the news gathered by Newistic over a period of 26 days, between June 21st and July 17th 2008. It shows the total number of new links found, the number of HTML pages actually downloaded, the number of pages that were

Table 1. Statistics over 26 days

New links found	7,735,581
Pages downloaded	6,600,534
News found	3,135,729
Not news	3,302,859
Not English	161,946

Table 2. Clustering results

Clustered News	**200,000**	**150,000**	**100,000**
Clustering (seconds)	743.59	410.99	175.05
Cosines normal QT	20 billion	11,25 billion	5 billion
Cosines modified QT	75,7 million	42,6 million	18,8 million
Cosines normal/modified	264.0048	263.9848	264.6371
Clustered News	**50,000**	**25,000**	**10,000**
Clustering (seconds)	40.92	10.879	2.729
Cosines normal QT	1,2 billion	312,5 million	50 million
Cosines modified QT	4,736,080	1,177,779	187,695
Cosines normal/modified	263.94	265.34	266.42

considered news articles, and the number of pages that were found as not being in the English language. It can be seen that less than 50.

6.2 Clustering results

For clustering, we have developed a series of tests performed on the Reuters RCV1 [10] corpus. We have run the process several times, varying the number of news articles to be clustered. The tests have been executed on a machine with a 2.4 GHz quad-core processor and 4 GB RAM. Results are shown in Table 3. The table shows the duration of the clustering starting from the moment when all the documents have been read, and the Term Frequency matrix has been constructed. We have decided to exclude the reading process because it depends on factors that have nothing in common with the clustering process itself. An important aspect that can be observed from Table 3 is that the number of cosines computed by our optimized implementation is significant less than the number of cosines that should otherwise be computed, by a factor of around 260.

7 Future Work

Newistic is designed to be a basic platform on which more advanced data mining research can be carried out. After solving remaining issues such as index sharding and distributed clustering, we want to tackle more high level functions like disambiguation, personalization and sentiment analysis.

The only component that we need to scale out is the text clustering module. The algorithm implemented by us is difficult to fully parallelize and distribute but we are currently working on this issue.

References

1. Forgy, E.: Cluster Analysis of Multivariate Data: Efficiency versus Interpretability of Classification. Biometrics, 768–780 (1965)
2. Masand, B., Linoff, G., Waltz, D.: Classifying news stories using memory based reasoning. In: Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval, Copenhagen, Denmark, pp. 59–65 (1992)

3. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, Berkeley, California, USA, pp. 42–49 (1999)
4. Heyer, L., Kruglyak, S., Yooseph, S.: Exploring Expression Data: Identification and analysis of Coexpressed Genes. Genome Research 9, 1106–1115 (1999)
5. Radev, D.R., Blair-Goldensohn, S., Zhang, Z., Raghavan, R.S.: Revathi Sundara Raghavan, Interactive, Domain-Independent Identification and Summarization of Topically Related News Articles. In: Proceedings of the 5th European Conference on Research and Advanced Technology for Digital Libraries, September 04-09, pp. 225–238 (2001)
6. Radev, D.R., Blair-Goldensohn, S., Zhang, Z., Raghavan, R.S.: NewsInEssence: a system for domain-independent, real-time news clustering and multi-document summarization. In: Proceedings of the first international conference on Human Language Technology research, San Diego, pp. 1–4 (2001)
7. McKeown, K., Barzilay, R., Evans, D., Hatzivassiloglou, V., Klavans, J.L., Nenkova, A., Sable, C., Schiffman, B., Sigelman, S.: Tracking and Summarizing News on a Daily Basis with Columbia's Newsblaster. In: Proceedings of Human Language Technology Conference 2002, San Diego, CA (2002)
8. Dunlavy, D.M., Conroy, J., O'Leary, D.P.: QCS: a tool for querying, clustering, and summarizing documents. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Demonstrations, Edmonton, Canada, pp. 11–12 (2003)
9. Gabrilovich, E., Dumais, S., Horvitz, E.: Newsjunkie: Providing Personalized Newsfeeds via Analysis of Information Novelty. In: Proceedings of 13th International World Wide Web Conference, New York, NY, USA (2004)
10. Lewis, D., Yang, Y., Rose, T.G., Li, F.: RCV1: A New Benchmark Collection for Text Categorization Research. The Journal of Machine Learning Research 5, 361–397 (2004)
11. Gulli, A.: The anatomy of a news search engine. In: Proceedings of 14th International World Wide Web Conference, Chiba, Japan, pp. 880–881 (2005)
12. Del Corso, G.M., Gulli, A., Romani, F.: Ranking a stream of news. In: Proceedings of 14th International World Wide Web Conference, Chiba, Japan (2005)
13. Steinberger, R., Pouliquen, B., Ignat, C.: Navigating multilingual news collection using automatically extracted information. In: Proceeding of the 27th International Conference on Information Technology Interfaces (ITI 2005) (2005)
14. Das, A.S., Datar, M., Garg, A., Rajaram, S.: Google news personalization: scalable online collaborative filtering. In: Proceedings of 16th International World Wide Web Conference, Banff, Alberta, Canada, pp. 271–280 (2007)
15. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: "GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In: Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL 2002), Philadelphia (July 2002)

Building a Text Classifier by a Keyword and Unlabeled Documents*

Qiang Qiu, Yang Zhang, and Junping Zhu

College of Information Engineering, Northwest A&F University,
Yangling, Shaanxi Province, P.R. China, 712100
qiuqiang@nwsuaf.edu.cn, zhangyang@nwsuaf.edu.cn, zhujunping@nwsuaf.edu.cn

Abstract. Traditional approaches for building text classifiers usually require a lot of labeled documents, which are expensive to obtain. In this paper, we study the problem of building a text classifier from a keyword and unlabeled documents, so as to avoid labeling documents manually. Firstly, we expand the keyword into a set of query terms and retrieve a set of documents from the set of unlabeled documents. Then, from the documents retrieved, we mine a set of positive documents. Thirdly, with the help of these positive documents, more positive documents could be extracted from the unlabeled documents. And finally, we train a *PU* text classifier with these positive documents and unlabeled documents. Our experiment result on 20Newsgroup dataset shows that the proposed approach could help to build excellent text classifiers.

1 Introduction

For real-life text classification applications, it is usually very expensive to label text documents manually. In order to reduce the amount of labeled documents for training, semi-supervised learning algorithms, which utilize both labeled and unlabeled data, have been used for text classification [1,2,3]. Another approach is to develop classification algorithm based on a set of labeled positive examples and a large amount of unlabeled documents [4,5,6,7,8,9,10,11]. Recently, some researchers have proposed labeling keywords to build a classifier instead of labeling documents manually [12,13].

However, these approaches require either a set of labeled documents or a set of labeled keywords. In this paper, we consider the problem of training a text classifier from a keyword and unlabeled documents. The proposed approach consists of 4 steps. Firstly, we expand the query from a single keyword to a set of query terms, and retrieve an initial set of documents from the set of unlabeled documents, U. Generally speaking, the retrieved documents could include both relevant and irrelevant documents. Then, an association rule based algorithm is used to discriminate the relevant documents from the irrelevant ones. We

* This work is supported by Young Cadreman Supporting Program of Northwest A&F University (01140301). Corresponding author: Yang Zhang.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 564–571, 2009.
© Springer-Verlag Berlin Heidelberg 2009

consider these relevant documents as positive documents P. Thirdly, we enlarge the positive documents P from $U - P$. And finally, an state-of-art PU [11] learning technique is employed to learn from P and $U - P$.

In order to measure the performance of the proposed approach, we made experiments on the 20Newsgroup dataset, with promising experiment result. The experiment result shows that our proposed approach could help to build good classifiers, and be more applicable to real-life text classification applications.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 presents our proposed approach. Section 4 gives our experiment results; and finally, section 5 concludes this paper and gives our further work.

2 Related Work

Due to the problem of manual labeling, with a small set of labeled documents, co-training method is proposed to build text classifiers in [1]. *Nigam et al.* used an algorithm for learning from labeled and unlabeled documents based on the combination of Expectation Maximization (EM) and Naive Bayes algorithm [3].

For learning from large set of positive and unlabeled examples, a number of practical PU learning algorithms were proposed, e.g. S-EM [4], ROC-SVM [5], PEBL [6]. Furthermore, there is a few successful algorithms proposed by the research community to cope with the situation when there is only a small set of positive documents [10,11]. The PNLH-SVM [10] and LPLP [11] methods focus on enlarging positive examples.

The work in [12,13,14] are more related to our research. With unlabeled training data, the approaches have been proposed to build a classifier by labeling keywords instead of labeling training examples [12,13]. The approach in [12] use many keywords to label raw training examples. However, it is too difficult for a user to provide sufficient keywords for accurate learning. So, the approach in [13] utilizes clustering and feature selection technique to label some words for each class. And then, the training examples are extracted based on the labeled words and NB-EM algorithm is used to build classifiers. In [14], an initial set of documents are retrieved from information resource, from which, *DocMine* [15] algorithm is used to mine a set of positive documents. And then, the reliable negative examples N are mined to build a SVM classifier.

However, the approaches in [12,13] require labeling keywords manually, which is difficult to provide sufficient keywords. The method in [14] needs more than one search engine. Only a small part of positive documents could be extracted by the search engines. Hence, the positive examples mined by the method in [14] would be incompetent for recognizing all the hidden positives in unlabeled documents U as well as those in the test sets.

The approach proposed in this paper builds a classifier based on only a topic keyword w and a large of unlabeled documents U. The proposed approach releases users from heavy burden of labeling documents or labeling keywords manually, which makes it more applicable to real-life applications.

3 Building a Classifier by a Keyword and Unlabeled Documents

The approach proposed in this paper consists of four steps.

Step 1: retrieving documents D with respect to query w from the unlabeled documents U;

Step 2: applying association rule based algorithm to mine a set of positive documents P_1, from D;

Step 3: enlarging the positive examples P_1 from $U - P_1$ with the help of P_1;

Step 4: employing the state-of-art PU learning techniques to build a text classifier.

3.1 Retrieving Documents D by a Keyword w

we use the method proposed in [16] to retrieve documents. The method consists of two steps: (1) term expansion; (2) document retrieval.

Term Expansion [16]: Firstly, the query w is augmented by synonym terms (the most common sense is taken). Then, the query is augmented by its nearest hypernyms which are semantically similar to terms already in the query. The set of expanded query term Q could be formulated as:

$$T = \{w\} \cup \{w's\ sysnonyms\} \tag{1}$$

$$Q = T \cup \bigcup_{t \in T} \{t's\ nearest\ hypernyms\} \tag{2}$$

Document Retrieval: According to the default Lucene[1] similarity, the similarity between the expanded query Q and a document d could be formulated as:

$$sim(Q, d) = \sum_{t \in Q} (frequency^{\frac{1}{2}} * [1 + log(\frac{numDocs}{docFreq + 1})]^2) \tag{3}$$

$frequency$: the number of observation of term t in document d
$numDocs$: total number of documents
$docFreq$: the total number of documents in which term t could be observed
The set of retrieved documents D could be defined as:

$$D = \{d | sim(Q, d) > 0; d \in U\} \tag{4}$$

3.2 Mining Positive Documents P_1 from D

In last sub-section, a set of documents D could be retrieved from U. Usually, more documents could be retrieved by the expanded query Q than by the keyword w. However many documents in D might be irrelevant to our query. In this paper, it was assumed that the relevant ones are more than irrelevant ones. We employ the *DocMine* [15] algorithm to label positive examples P_1 from D.

[1] http://lucene.apache.org/

The retrieved documents D are sorted into a descending sequence D_{seq} according to the similarity between the expanded query Q and the document d. And then, D_{seq} is divided into two equal parts, D_1 and D_2. D_1 is made up of the top 50% documents in D_{seq} and D_2 is the rest of documents in D_{seq}. We select h% documents without replacement from D_2 randomly, and use these selected documents together with all the documents in D_1 to form one bucket of documents. In this way, we create l buckets of documents. Then, *DocMine* algorithm is employed to mine the set of positive documents, P_1.

3.3 Enlarging the Set of Positive Examples P_1

However, the set P_1 mined by *DocMine* is only a small part of U with a low *recall* index. The small positive set P_1 could not adequately represent the whole positive category.

In order to improve the performance of the text classifier, we enlarge the set of positive examples based on the method proposed in [10]. This method includes two steps: (1) extracting reliable negative examples; (2) enlarging the sets of positive examples.

Extracting reliable negative examples: In this paper, we employ the algorithm *ExtractReliableNegative*(P_1,$U - P_1$) [10] to extract the negative examples. And then, the negative examples are sorted descendingly according to their similarity. The set of reliable negative examples, N, is made up of the last 40% of the negative examples in the sequence.

Enlarging the positive examples [10]: Firstly, we partition N, the set of reliable negative examples, into k clusters, $(N_1, N_2, \ldots, N_k)$, $k = \sqrt{N/P_1}$. By doing so, each partition focuses on a smaller set of more related features. We measure the similarity of the document $d \in U - P_1 - N$ with the centroid of P_1, cid_{P_1} and the centroid of N_i, cid_{N_i}, $i = 1, 2, \ldots, k$. $Centroid(P_1, N_1, N_2, \cdots, N_k)$[10] is used to get the centroid of the clusters. Here, the cosine coefficient is used for measuring the similarity between a document d and a centroid cid:

$$cos(d, cid) = \frac{d \cdot cid}{\|d\| \cdot \|cid\|} \tag{5}$$

According to this similarity, negative examples are extracted from $U - P_1 - N$ repeatedly until no more documents in $U - P_1 - N$ could be extracted. Then, we start to extract positive documents from $U - P_1 - N$.

3.4 Building Text Classifiers

In the previous section, we have mined the positive examples P from the unlabeled documents U by the keyword w. Next, *PU* learning algorithms are employed to classify the documents in testing data set. In this paper, ROC-SVM [17], SPY-SVM [17] and NB-SVM [17] are used to build a text classifier. In these three algorithms, ROC, SPY and NB algorithms are used respectively to identify a set of reliable negative documents from the unlabeled documents. And then, SVM learning algorithm is used iteratively to build the final classifier.

4 Evaluation Experiments

In this section, we evaluate our proposed approach and compare it with the method based on the positive and unlabeled documents. The LPU[2] system is publically available online and contains ROC-SVM [17], SPY-SVM [17] and NB-SVM [17].

4.1 DataSet and Experiment Settings

We made our experiments on 20Newsgroup[3] dataset. There are 20 categories in the 20Newsgroups dataset, with l0 of them being selected randomly in our experiment. Each category has approximately 1000 articles. For each category, 70% of the documents are used as unlabeled data for training, and the remaining 30% of the documents as the testing documents.

The preprocessing includes stop word removing and stemming. For the document retrieval, the JWNL[4] is used to expand the query keyword w and the Lucene is used to retrieve the documents. For Mining positive documents from retrieved documents, according to [15], we set $l = 5$. For enlarging positive documents, according to [10], for feature selection, we select 3000 features. The classical k-means clustering algorithm[18] is used to cluster the reliable negative examples N. For document classification, the three learning algorithms (ROC-SVM, SPY-SVM, NB-SVM) of the LPU system are employed to build text classifiers.

We randomly select 300 positive examples from the training set for each category and compare our approach with *PU* classifier, which is trained from 300 positive documents and unlabeled documents.

4.2 Experimental Results

In this paper, *Recall*, *Precision* and F_1 are used to measure the performance of our text classifier, as they are widely used by the research community of text classification. Table 1, table 2 and table 3 give the classification performance of ROC-SVM, SPY-SVM and NB-SVM, respectively. In each table, column 1 lists the category name, column 2 lists the corresponding query keyword, column 3, 4 show the performance of our approach based on keyword and *PU* classifier based on positive examples.

It is obviously from table 1, table 2, and table 3 that the approach proposed in this paper outperforms the *PU* classifiers, which are trained on 300 positive examples and unlabled examples. Based on a keyword and unlabeled documents, our approach could help to build a good text classifier. We believe that our proposed approach requires less user effort to label a sufficient set of documents, even in case of a small positive examples.

[2] http://www.cs.uic.edu/∼liub/LPU/LPU-download.html
[3] http://people.csail.mit.edu/jrennie/20Newsgroups/
[4] http://sourceforge.net/projects/jwordnet

Table 1. The Classification Performance of ROC-SVM

Category	Query Word	one keyword(%)			300 Positives(%)		
		Recall	Precision	F_1	Recall	Precision	F_1
alt.atheism	atheism	52.51	98.13	**68.41**	43.14	100.00	60.28
rec.sport.baseball	baseball	79.93	90.87	**85.05**	33.11	98.02	49.50
soc.religion.christian	christian	98.31	93.27	**95.72**	41.89	100.00	59.05
sci.crypt	encrypt	90.64	48.39	**63.10**	30.10	98.90	46.15
sci.electronics	electronics	21.07	98.44	**34.71**	20.40	100.00	33.89
comp.graphics	graphics	95.99	62.66	**75.83**	38.46	99.14	55.42
talk.politics.guns	guns	80.94	98.37	**88.81**	21.40	100.00	35.26
rec.sport.hockey	hockey	90.97	75.35	**82.42**	29.43	100.00	45.48
talk.politics.mideast	mideast	75.59	97.41	**85.12**	51.51	100.00	67.99
rec.motorcycles	motorcycles	75.59	95.36	**84.33**	37.12	100.00	54.15

Table 2. The Classification Performance of SPY-SVM

Category	Query Word	one keyword(%)			300 Positives(%)		
		Recall	Precision	F_1	Recall	Precision	F_1
alt.atheism	atheism	51.17	98.08	**67.25**	43.14	100.00	60.28
rec.sport.baseball	baseball	71.57	94.69	**81.52**	32.44	97.98	48.74
soc.religion.christian	christian	98.31	93.57	**95.88**	89.19	99.62	94.12
sci.crypt	encrypt	88.63	49.17	**63.25**	28.76	98.85	44.56
sci.electronics	electronics	20.07	98.36	33.33	22.74	98.55	**36.96**
comp.graphics	graphics	95.32	62.91	**75.80**	43.81	100.00	60.93
talk.politics.guns	guns	74.25	98.23	**84.57**	24.75	98.67	39.57
rec.sport.hockey	hockey	87.63	78.68	**82.91**	28.76	100.00	44.68
talk.politics.mideast	mideast	74.92	96.97	**84.53**	50.5	100.00	67.11
rec.motorcycles	motorcycles	71.24	95.52	**81.61**	37.12	100.00	54.15

Table 3. The Classification Performance of NB-SVM

Category	Query Word	one keyword(%)			300 Positives(%)		
		Recall	Precision	F_1	Recall	Precision	F_1
alt.atheism	atheism	52.17	98.11	**68.12**	42.81	100.00	59.95
rec.sport.baseball	baseball	70.57	94.62	**80.84**	34.45	98.10	50.99
soc.religion.christian	christian	98.31	93.27	**95.72**	50.34	100.00	66.97
sci.crypt	encrypt	88.96	48.90	**63.11**	33.78	99.02	50.37
sci.electronics	electronics	23.41	97.22	**37.74**	21.74	100.00	35.71
comp.graphics	graphics	95.32	64.04	**76.61**	38.46	99.14	55.42
talk.politics.guns	guns	71.24	98.16	**82.56**	20.40	100.00	33.89
rec.sport.hockey	hockey	90.97	75.14	**82.30**	30.43	100.00	46.67
talk.politics.mideast	mideast	67.22	97.57	**79.60**	47.16	100.00	64.09
rec.motorcycles	motorcycles	72.58	95.18	**82.35**	35.79	100.00	52.71

Here, we analyze the number of true positive examples in each step of our proposed approach, which is shown in Table 4. In table 4, column 1 lists the category name, column 2,3 and 4 gives the information of true positives in step 1, 2 and

Table 4. The True Positives in Each Step

Category	True Positives								
	Step 1			Step 2			Step 3		
	$\|D\|$	$\|p\|$	Precision	$\|P_1\|$	$\|p\|$	Precision	$\|P\|$	$\|p\|$	Precision
alt.atheism	108	91	84.26	89	81	91.01	426	414	97.18
rec.sport.baseball	508	335	65.94	128	95	74.22	363	323	88.98
soc.religion.christian	1145	698	60.96	214	155	72.43	737	629	85.35
sci.crypt	377	332	88.06	176	162	92.61	1200	635	52.92
sci.electronics	103	68	66.02	42	31	73.81	247	236	95.55
comp.graphics	442	255	57.69	91	72	79.12	1055	622	58.96
talk.politics.guns	570	362	63.51	120	112	93.33	528	479	90.72
rec.sport.hockey	262	256	97.71	147	146	99.32	835	595	71.26
talk.politics.mideast	73	62	84.93	47	42	89.36	367	314	85.56
rec.motorcycles	392	382	96.71	209	204	97.61	474	465	98.1

3, respectively. Here $|p|$ shows the number of true positive documents; *Precision* shows the precision index in the corresponding step positive documents; $|D|$ shows the number of documents retrieved in Step 1; $|P_1|$ shows the number of documents mined in Step 2; $|P|$ shows the number of documents mined after Step 3.

From table 4, it is obviously that *DocMine* could improve the relevance of the retrieved documents, and the method of enlarging the positive examples could mine more relevant documents from the unlabeled documents. From the category *sci.electronics*, it could be concluded the performance of a classifier is relevant to the number of the positive examples mined, as a small set of positive examples could not represent all the hidden positive examples in unlabeled documents. Meanwhile, the more irrelevant documents mined in step 3, the worse a classifier performs, eg, *sci.crypt*, *comp.graphics*. The more relevant documents and the less irrelevant documents mined in these three steps, the better a classifier performs, eg, *soc.religon.christian*.

5 Conclusion and Future Work

In many real-world text classification applications, it is often expensive to obtain enough training examples for building good text classifiers. In this paper, a new approach has been proposed to learn from a keyword w and the unlabeled data U. To cope with lacking of training examples, firstly, term expansion and association rule based algorithm are used to label some positive examples P_1 from U. Then, with the help of P_1, more positive examples P_2 are extracted from $U-P_1$. Finally, the state-of-art PU learning algorithm is applied to build a classifier based on $P_1 \cup P_2$ and $U - P_1 - P_2$. The experiment results show our approach can build a good classifier from a keyword and unlabeled documents. The proposed approach could help to reduce the process of labeling documents manually, and could be more applicable to real-life text classification applications.

In our future work, we plan to study how to identify more positive examples from unlabeled documents U with the help of only a small positive examples P, so as to further improve the performance of our classifier.

References

1. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: COLT (1998)
2. Ghnai, R.: Combining labeled and unlabeled data for multiclass text categorization. In: ICML (2002)
3. Nigam, K., McCallum, A., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using EM. Machine learning 39(2-3), 103–134 (2000)
4. Liu, B., Lee, W., Yu, P., Li, X.: Partially Supervised Classification of Text Documents. In: ICML, pp. 387–394 (2002)
5. Li, X., Liu, B.: Learning to Classify Texts Using Positive and Unlabeled Data. In: IJCAI, pp. 587–594 (2003)
6. Yu, H., Han, J., Chang, K.C.-C.: PEBL: Positive Example Based Learning for Web Page Classification Using SVM. In: KDD, pp. 239–248 (2002)
7. Fung, G.P.C., Yu, J.X., Lu, H., Yu, P.S.: Text Classification without Negative Examples. In: Proc. 21st Int'l Conf. Data Eng. (2005)
8. Yu, H., Han, J., Chang, K.C.-C.: PEBL: Web Page Classification without Negative Examples. IEEE Trans. Knowledge and Data Eng. (2004)
9. Li, X., Liu, B.: Learning from Positive and Unlabeled Examples with Different Data Distributions. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS, vol. 3720, pp. 218–229. Springer, Heidelberg (2005)
10. Fung, G.P.C., et al.: Text Classification without Negative Examples Revisit. IEEE Transactions on Knowledge and Data Engineering 18(1), 6–20 (2006)
11. Li, X., Liu, B., Ng, S.-K.: Learning to Classify Documents with Only a Small Positive Training Set. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS, vol. 4701, pp. 201–213. Springer, Heidelberg (2007)
12. McCallum, A.: Nigam. K.: Text classification by bootstrapping with keywords, EM and shrinkage. In: ACL Workshop on Unsupervised Learning in Natural Language Processing (1999)
13. Liu, B., Li, X., Lee, W., Yu, P.: Text Classification by Labeling Words. In: Proc. 19th Nat'l Conf. Artificial Intelligence (2004)
14. Barbara, D., Domeniconi, C., Kang, N.: Classifying Document Without Labels. In: Proceedings of the SIAM International Conference on Data Mining (2004)
15. Barbara, D., Domeniconi, C., Kang, N.: Mining Relevant Text from Unlabeled Documents. In: Proceedings of the Third IEEE International Conference on Data Mining (2003)
16. Varelas, G., Voutsakis, E., Raftopoulou, P., Petrakis, E., Milios, E.: Semantic Similarity Methods in Wordnet and their Application Information Retrieval on the Web. In: 7th ACM International Workshop on Web Information and Data Management (2005)
17. Liu, B., Dai, Y., Li, X., Lee, W.S., Yu, P.S.: Building Text Classifiers Using Positive and Unlabeled Examples. In: Proc. Third Int'l Conf. Data Mining (2003)
18. Bradley, P.S., Fayyad, U.: Refining Initial Points for k-Means Clustering. In: Proc. 15th Int'l Conf. Machine Learning (1998)

A Discriminative Approach to Topic-Based Citation Recommendation*

Jie Tang and Jing Zhang

Department of Computer Science and Technology,
Tsinghua University, Beijing, 100084, China
jietang@tsinghua.edu.cn, zhangjing@keg.cs.tsinghua.edu.cn

Abstract. In this paper, we present a study of a novel problem, i.e. topic-based citation recommendation, which involves recommending papers to be referred to. Traditionally, this problem is usually treated as an engineering issue and dealt with using heuristics. This paper gives a formalization of topic-based citation recommendation and proposes a discriminative approach to this problem. Specifically, it proposes a two-layer Restricted Boltzmann Machine model, called RBM-CS, which can discover topic distributions of paper content and citation relationship simultaneously. Experimental results demonstrate that RBM-CS can significantly outperform baseline methods for citation recommendation.

1 Introduction

Citation recommendation is concerned with recommending papers that should be referred to. When starting a work in a new research topic or brainstorming for novel ideas, a researcher usually wants to have a quick understanding of the exiting literatures in this field, including which papers are the most relevant papers and what sub-topics are presented in these papers. Two common ways to find reference papers are: (1) search documents on search engines such as Google and (2) trace the cited references by starting with a small number of initial papers (seed-papers). Unfortunately, for (1) it would be difficult to find a comprehensive keyword list to cover all papers, especially for beginning researchers. It is very possible to miss important developments in areas outside a researcher's specialty. For (2), an average paper may cite more than twenty papers. It would be quite time consuming to analyze each of the cited reference to see whether it is useful or not, especially with the increase of the tracing depth. Additionally, even a well organized paper may miss some important "related work", due to space limitation or other reasons.

Previously, papers recommendation has been studied, for example, by exploring collaborative filtering [7]. Our problem is relevant, but different, from this kind of work. Firstly, in citation recommendation, the user is interested in not only a list of recommended papers, but also the sub-topics presented in these papers. Secondly, conventional methods can only recommend papers; but cannot suggest the citation position (i.e., which sentences should refer to the citation).

* The work is supported by the National Natural Science Foundation of China (60703059), Chinese National Key Foundation Research and Development Plan (2007CB310803), and Chinese Young Faculty Research Funding (20070003093).

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 572–579, 2009.
© Springer-Verlag Berlin Heidelberg 2009

In this paper, we formalize citation recommendation as that of topic discovery, topic-based recommendation, and matching citation sentences with the recommended papers. We propose a unified and discriminative approach to citation recommendation. This approach can automatically discover topical aspects of each paper and recommend papers based on the discovered topic distribution. Experimental results show that the proposed approach significantly outperforms the baseline methods.

2 Problem Formulation

We define notations used throughout this paper. Assuming that a paper d contains a vector $\mathbf{w}_d$ of N_d words, in which each word w_{di} is chosen from a vocabulary of size V; and a list $\mathbf{l}_d$ of L_d references. Then a collection of D papers can be represented as $\mathbf{D} = \{(\mathbf{w}_1, \mathbf{l}_1), \cdots, (\mathbf{w}_D, \mathbf{l}_D)\}$. We only consider references that appear in the paper collection $\mathbf{D}$. Thus the size L of the vocabulary of references is D. Further, we consider that each paper is associated with a distribution of T topics, so is the citation.

Definition 1. ***(Citation Context and Citation Sentence)*** *Citation context is defined by the context words occurring in, for instance, the user written proposal. For an example, the words "... We use Cosine computation [x] evaluate the similarity ..." would be a citation context. One reference paper is expected to be cited at the position "[x]". We use c to denote a citation context. Each sentence in the citation context is called citation sentence. The position "[x]" to cite the reference paper is called citation position.*

Figure 1 shows an example of citation recommendation. The left part of Figure 1 includes a citation context provided by the user and a paper collection. The right part shows the recommended result that we expect a citation recommendation algorithm outputs. For instance, two topics, i.e., "text summarization" and "information retrieval", have been extracted from the citation context. For the first topic "text summarization", two papers have been recommended and for the second topic "information retrieval", three papers have been recommended. Further, the recommended papers are matched with the citation sentences and the corresponding citation positions have been identified.

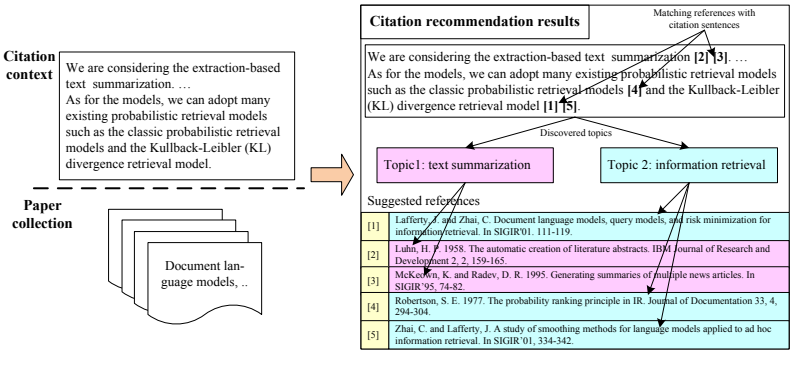


Fig. 1. Example of citation recommendation

We see that the recommended papers are topic dependent. By nature, the problem of citation recommendation can be formalized as topic discovery, reference papers recommendation, and matching of the recommended papers with the citation sentences.

3 Our Approach

At a high level, our approach primarily consists of three steps:

1. We propose a two-layer Restricted Boltzmann Machine (RBM) model, referred to as RBM-CS. Given a collection of papers with citation relationship, the model learns a mixture of topic distribution over paper contents and citation relationships.
2. We present a method to rank papers for a given citation context, based on the learned topic model. We take the top ranked papers as the recommended papers.
3. We describe a method to find the correspondence between the recommended papers and the citation sentences.

3.1 The RBM-CS Model

Restricted Boltzmann Machines (RBMs) [8] are undirected graphical models that use a layer of hidden variables to model a (topic) distribution over visible variables. In this work, we propose a two-layer RBM model, called RBM-CS, to jointly model papers and citations. Graphical representation of the RBM-CS model is shown in Figure 2. We see that in RBM-CS, the hidden layer **h** is associated with two visible layers: words **w** and citation relationships **l**, respectively coupling with an interaction matrix **M** and **U**. The basic idea in RBM-CS is to capture the topic distribution of papers with a hidden topic layer, which is conditioned on both words and citation relationships. Words and citation relationship are considered to be generated from the hidden topics independently.

To train a graphical model, we can consider maximization of the generative log-likelihood $\log p(\mathbf{w}, \mathbf{l})$. However, we are dealing with a predictive problem, our interests ultimately only lie in correct prediction $p(\mathbf{l}|\mathbf{w})$, not necessarily to have a good $p(\mathbf{w})$. Therefore, we define a discriminative objective function by a conditional log-likelihood:

$$L = \sum_{d}^{D} \log p(\mathbf{l}_d|\mathbf{w}_d) = \sum_{d}^{D} \log \left(\prod_{j=1}^{L} p(l_j|\mathbf{w}_d) \right) \tag{1}$$

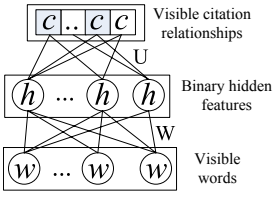


Fig. 2. Graphical representation of the RBM-CS model

The probability $p(l_j|\mathbf{w}_d)$ can be defined as:

$$p(l_j|\mathbf{w}) = \sigma\left(\sum_{k=1}^{T} U_{jk} f(h_k) + e_j\right), f(h_k) = \sigma\left(\sum_{i=1}^{V} M_{ij} f(w_i) + \sum_{j} U_{kj} f(l_j) + a_k\right) \tag{2}$$

where $\sigma(.)$ is a sigmoid function, defined as $\sigma(x) = 1/(1+exp(-x))$; e are bias terms for citation relationships; $f(h_k)$ is the feature function for hidden variable h_k; $f(l_j)$ and $f(w_i)$ are feature functions for citation relationship l_j and word w_i, respectively; a are bias terms for hidden variables. For simplicity, we define $f(w_i)$ as the count of word w_i in document d. We define binary value for the feature function of citation relationship l. For example, for document d, $f(l_j) = 1$ denotes that the document d has a citation relationship with another paper d_j.

Now, the task is to learn the model parameters $\Theta = (\mathbf{M}, \mathbf{U}, \mathbf{a}, \mathbf{b}, \mathbf{e})$ given a training set $\mathbf{D}$. Maximum-likelihood (ML) learning of the parameters can be done by gradient ascent with respect to the model parameters ($\mathbf{b}$ are bias terms for words). The exact gradient, for any parameter $\theta \in \Theta$ can be written as follows:

$$\frac{\partial \log p(\mathbf{l}|\mathbf{w})}{\partial \theta} = E_{P_0}[\mathbf{l}|\mathbf{w}] - E_{P_M}[\mathbf{l}|\mathbf{w}] \tag{3}$$

where $E_{P_0}[.]$ denotes an expectation with respect to the data distribution and E_{P_M} is an expectation with respect to the distribution defined by the model. Computation of the expectation E_{P_M} is intractable. In practice, we use a stochastic approximation of this gradient, called the contrastive divergence gradient [4]. The algorithm cycles through the training data and updates the model parameters according to Algorithm 1, where the probabilities $p(h_k|\mathbf{w}, \mathbf{l})$, $p(w_i|\mathbf{h})$ and $p(l_j|\mathbf{h})$ are defined as:

$$p(h_k|\mathbf{w}, \mathbf{l}) = \sigma(\sum_{i=1}^{V} M_{ik} f(w_i) + \sum_{j=1}^{L} U_{jk} f(l_j) + a_k) \tag{4}$$

$$p(w_i|\mathbf{h}) = \sigma(\sum_{k=1}^{T} M_{ik} f(h_k) + b_i) \tag{5}$$

$$p(l_j|\mathbf{h}) = \sigma(\sum_{k=1}^{T} U_{jk} f(h_k) + e_j) \tag{6}$$

where b are bias terms for words; $f(l_j)$ is the feature function for citation relationship.

Algorithm 1. Parameter learning via contrastive divergence

Input: training data $\mathbf{D} = \{(\mathbf{w}_d, \mathbf{l}_d)\}$, topic number T, and learning rate λ

1. repeat
   (a) for each document d:
      i. sampling each topic h_k according to (4);
      ii. sampling each word w_i according to (5);
      iii. sampling each citation relationship l_j according to (6);
   (b) end for
   (c) update each model parameter $\theta \in \Theta$ by

$$\theta = \theta + \lambda(\frac{\partial \log p(\mathbf{l}|\mathbf{w})}{\partial \theta})$$

2. until all model parameters Θ converge

3.2 Ranking and Recommendation

The objective of citation recommendation is to rank the recommended papers for a given citation context. Specifically, we apply the same modeling procedure to the citation context. Hence, we can obtain a topic representation $\{\mathbf{h}_c\}$ of the citation context c. Based on the topic representation and the modeling results, we can calculate the probability of each paper being the reference paper for the citation context according to Equation (6). Finally, the papers are ranked in terms of the probabilities and the top K ranked papers are returned as the recommended papers. It is hard to specify an accurate value of K for each citation context. A simple way is to set it as the average number of citations in a paper (i.e., 11 in our data set).

3.3 Matching Recommended Papers with Citation Sentences

The purpose of matching the recommended papers with citation sentences is to align the recommended papers with sentences in the citation context. This can be done by using each recommended paper as a keyword query to retrieve relevant citation sentences. In general, we may use any retrieval method. In this paper, we used KL-divergence to measure the relevance between the recommended paper and the citation sentence:

$$KL(d, s_{ci}) = \sum_{k=1}^{T} p(h_k|d) \log \frac{p(h_k|d)}{p(h_k|s_{ci})} \tag{7}$$

where d is a recommended paper and s_{ci} is the ith sentence in the citation context c; the probabilities $p(h_k|d)$ and $p(h_k|s_{ci})$, which can be obtained by (4).

4 Experiments

4.1 Experimental Setting

Data Set. We conducted experiments on two data sets, NIPS[1] and Citeseer[2]. The NIPS data set consists of 12 volumes of NIPS papers (1,605 papers and 10,472 citation relationships). Each paper contains full text and its citations. We removed some citations with incomplete information, e.g., consisting of only authors and publication venue, but no title. We also removed citations that do not appear in the data set. The Citeseer data set consists of $3,335$ papers (with 32,558 citation relationships) downloaded from the Citeseer web site. As well, we removed citations that do not appear in the data set.

Each paper was preprocessed by (a) removing stopwords and numbers; (b) removing words appearing less than three times in the corpus; and (c) downcasing the obtained words. Finally, we obtained $V = 26,723$ unique words and a total of $350,361$ words in NIPS and $V = 44,548$ unique words and $634,875$ words in Citeseer.

[1] `http://www.cs.toronto.edu/~roweis/data.html`
[2] `http://citeseer.ist.psu.edu/oai.html`

Evaluation Measure and Baseline Methods. We used P@1, P@3, P@5, P@10, Rpec, MAP, Bpref, and MRR as the evaluation measures. For the details of the measures, please refer to [1] [2]. We conducted the evaluation on both paper-level (without considering the citation position) and sentence-level (considering the citation position).

We defined two baseline methods. One is based on language model (LM). Given a citation context c, we computed the score of each paper d by $p(c|d) = \prod_{w \in c} p(w|d)$, where $p(w|d)$ is the maximum likelihood of word w in document d. We ranked papers according to this score and recommended the top K ranked papers.

The other baseline is based on RBM, which learns a generative model for papers and the citation context. Then we use KL-divergence to calculate a score for each paper (by a similar equation to Equation (7)). For both RBM and RBM-CS, we set the number of topic as $T = 200$ and the number of recommended references as the average number of the data set, i.e. $K = 7$ for NIPS and $K = 11$ for Citeseer. The weights were updated using a learning rate of 0.01/batch-size, momentum of 0.9, and a weight decay of 0.001.

4.2 Experimental Results

Estimated Topics. Table 1 shows two example topics discovered by RBM-CS from the NIPS data. We can see that our model can capture the topic distribution very well.

Performance of Citation recommendation. Table 2 shows the result of citation recommendation. We see that our proposed model clearly outperforms the two baseline models. The language model suffers from the fact that it is based on only keyword matching. The RBM uses a hidden topic layer to alleviate the problem. However, it is aimed at optimize $p(\mathbf{w})$, which might be inappropriate for citation recommendation. In addition, RBM cannot capture the dependencies between paper contents and citation relationships. Our proposed RBM-CS can be advantageous to optimize $p(l|w)$ directly and to model the dependencies between paper contents and citation relationships.

We can also see from Table 2 that the recommendation performance is much better on the Citeseer data than that on the NIPS data. This means that on the sparse data, the recommendation tasks would be more difficult. How to improve the recommendation performance on the sparse data is also one of our ongoing work.

Table 1. Two topics discovered by RBM-CS from the NIPS data

"Topic 12: Markov Model"			
Words		Citation	
hmm	0.091	links between Markov models and multilayer perceptrons	0.0347
state	0.063	a tutorial on hidden Markov models and selected applications in speech recognition	0.0221
markov	0.058	connectionist speech recognition a hybrid approach	0.0169
probability	0.057	global optimization of a neural network hidden Markov model hybrid	0.0169
field	0.018	neural network classifiers estimate Bayesian a posteriori probabilities	0.0169
"Topic 97: Support Vector Machines"			
Words		Citation	
kernel	0.083	the nature of statistical learning	0.036363
margin	0.079	a training algorithm for optimal margin classifiers	0.026984
support	0.075	a tutorial on support vector machines for pattern recognition	0.026763
svm	0.075	statistical learning theory	0.020220
machine	0.069	support vector networks	0.015117

Table 2. Performance of citation recommendation on the two data sets

Data	Method	P@1	P@2	P@3	P@5	P@10	Rprec	MAP	Bpref	MRR
NIPS	LM	0.0195	0.0164	0.0132	0.0125	0.0148	0.0161	0.0445	0.0108	0.0132
	RBM	0.0289	0.0313	0.0263	0.0224	0.0164	0.0245	0.0652	0.0176	0.0162
	RBM-CS	0.2402	0.2628	0.2349	0.1792	0.1170	0.1676	0.3499	0.1626	0.1082
Citeseer	LM	0.0496	0.0492	0.0454	0.0439	0.0274	0.0259	0.1103	0.0311	0.0243
	RBM	0.1684	0.1884	0.1780	0.1519	0.0776	0.1510	0.2804	0.1189	0.0639
	RBM-CS	0.3337	0.3791	0.3501	0.2800	0.1768	0.2375	0.4237	0.2501	0.1564

Table 3. Performance of sentence-level citation recommendation on the NIPS data set

Model	P@1	P@2	P@3	P@5	P@10	Rpec	MAP	Bpref	MRR
LM	0.0783	0.0642	0.0582	0.0629	0.00503	0.0607	0.1178	0.0483	0.0502
RBM	0.1081	0.1061	0.1061	0.1000	0.0727	0.0914	0.2089	0.0761	0.0851
RBM-CS	0.2005	0.2136	0.2010	0.1788	0.1561	0.1782	0.2854	0.1565	0.1657

Table 3 shows the performance of citation recommendation by RBM and RBM-CS in terms of sentence-level evaluation. (As the Citeseer data contains a lot OCR errors and it is difficult to accurately extract the citation position, we conducted sentence-level evaluation on the NIPS data only.) We can again see that our proposed model significantly outperforms the method of using LM and that of using RBM.

5 Related Work

We review scientific literatures about citation analysis and related topic models. Citation analysis usually employs a graphical model to represent papers and their relationships, for example Science Citation Index [3]. This index links authors and their corresponding papers. Bibliographical Coupling (BC) [6] and co-citation analysis are proposed for citation analysis, for example to measure the quality of an academic paper [3].

Recommending citations for scientific papers is a task which has not been studied exhaustively before. Strohman et al. [9] investigated this task using a graphical framework. Each paper is represented by a node and the citation relationship is represented as the link between nodes. A new paper is a node without in and out links. Citation recommendation is then cast as link prediction. McNee et al. [7] employed collaborative filtering in citation network to recommend citations to papers. Both of them use the graphical framework. We look at citation recommendation from a different perspective. We take advantages of the dependencies between paper contents and citation relationships by using a hidden topic layer to joint model them.

Restricted Boltzmann Machines (RBMs) [8] are generative models based on latent (usually binary) variables to model an input distribution, and have been applied in a large variety of problems in the past few years. Many extensions of the RBM model have been proposed, for example dual wing RBM [12], modeling various types of input distribution [5] [11]. In this paper, we propose a two-layer Restricted Boltzmann

Machine model, called RBM-CS, which can jointly model topic distribution of papers and citation relationships.

6 Conclusion

In this paper, we formally define the problems of topic-based citation recommendation and propose a discriminative approach to this problem. Specifically, we proposes a two-layer Restricted Boltzmann Machine model, called RBM-CS, to model paper contents and citation relationships simultaneously. Experimental results show that the proposed RBM-CS can significantly improve the recommendation performance.

There are many potential future directions of this work. It would be interesting to include other information for citation recommendation, such as conference and author information. We are going to integrate the citation recommendation as a new feature into our academic search system ArnetMiner [10] (http://arnetminer.org).

References

1. Buckley, C., Voorhees, E.M.: Retrieval evaluation with incomplete information. In: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2004), pp. 25–32 (2004)
2. Craswell, N., de Vries, A.P., Soboroff, I.: Overview of the trec-2005 enterprise track. In: TREC 2005 Conference Notebook, pp. 199–205 (2005)
3. Garfield, E.: Citation analysis as a tool in journal evaluation. Science 178(4060), 471–479 (1972)
4. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation 14, 1771–1800 (2002)
5. Hinton, G.E.: A fast learning algorithm for deep belief nets. Neural Computation 18, 1527–1554 (2006)
6. Kessler, M.M.: Bibliographic coupling between scientific papers. American Documentation 14, 10–25 (1963)
7. McNee, S.M., Albert, I., Cosley, D., Gopalkrishnan, P., Lam, S.K., Rashid, A.M., Konstan, J.A., Riedl, J.: On the recommending of citations for research papers. In: CSCW 2002, pp. 116–125 (2002)
8. Smolensky, P.: Information processing in dynamical systems: foundations of harmony theory. pp. 194–281 (1986)
9. Strohman, T., Croft, W.B., Jensen, D.: Recommending citations for academic papers. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007), pp. 705–706 (2007)
10. Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., Su, Z.: Arnetminer: Extraction and mining of academic social networks. In: KDD 2008, pp. 990–998 (2008)
11. Welling, M., Rosen-Zvi, M., Hinton, G.E.: Exponential family harmoniums with an application to infomration retrieval. In: Proceedings of the 17th Neural Information Processing Systems (NIPS 2005) (2005)
12. Xing, E.P., Yan, R., Hauptmann, A.G.: Mining associated text and images with dual-wing harmoniums. In: Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI 2005), pp. 633–641 (2005)

Romanization of Thai Proper Names Based on Popularity of Usages

Akegapon Tangverapong, Atiwong Suchato, and Proadpran Punyabukkana

Spoken Language Systems Research Group,
Department of Computer Engineering,
Faculty of Engineering, Chulalongkorn University, Bangkok, 10330, Thailand
Akegapon.T@student.chula.ac.th, atiwong@cp.eng.chula.ac.th,
Proadpran.P@Chula.ac.th

Abstract. The lack of standards for Romanization of Thai proper names makes searching activity a challenging task. This is particularly important when searching for people-related documents based on orthographic representation of their names using either solely Thai or English alphabets. Romanization based directly on the names' pronunciations often fails to deliver exact English spellings due to the non-1-to-1 mapping from Thai to English spelling and personal preferences. This paper proposes a Romanization approach where popularity of usages is taken into consideration. Thai names are parsed into sequences of grams, units of syllable-sized or larger governed by pronunciation and spelling constraints in both Thai and English writing systems. A Gram lexicon is constructed from a corpus of more than 130,000 names. Statistical models are trained accordingly based on the Gram lexicon. The proposed method significantly outperformed the current Romanization approach. Approximately 46% to 75% of the correct English spellings are covered when the number of proposed hypotheses increases from 1 to 15.

Keywords: Thai Romanization, Statistical Language Processing, Machine Translation.

1 Introduction

Thai *Romanization* refers to the method of writing Thai word with English alphabet, which somehow is not standardized. The problem is more prominent for Thai proper names. Personal preferences affect how ones spell names using English alphabets significantly. In some cases, name pronunciations are strictly preserved (transcription) while in many cases spellings reflecting the roots of the names are preferred over how they would sound (transliteration). In this work, we propose a Thai Romanization approach where popularity of Romanization patterns is taken into consideration. A Romanization approach that is capable of producing a list of probable hypotheses of Romanized string given a Thai proper name would be beneficial in searching activities relating to retrieving people-related documents published in either Thai or English when names are used as the search keywords.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 580–587, 2009.
© Springer-Verlag Berlin Heidelberg 2009

2 Literature Review

Romanization is typically approached in two steps: text segmentation and writing system translation. The first one segments the original text string into sequences of method-specific units such as words, syllables, or some other orthographic-based units. The latter proposes the spelling of the original string in the destination writing system.

For Thai, there have been some researches related to Thai text segmentation. Poovarawan [1], Sornlertlamvanich [2] proposed a dictionary-based. Unregistered words will not be recognized. Theeramunkong [3] proposed a method based on a decision tree model without using dictionaries. This method can solve the unregistered word problem. Still, it cannot handle cases with ambiguituies. Aroonmanakun [4] proposed a segmentation method based on syllable trigram models. Romanization based on this segmentation method yields reasonable result. However, the method is not designed to handle the transliteration type of Romanization where resulting English strings aim at preserving some original linguistic information rather than retaining the closest pronunciations. An example of such names includes "โชติ", originally pronounced as a single syllable, while one of the most popular transliteration is a double-syllable Romanized string "choti".

Apart from transliteration, Romanization could be done via transcription, where Thai graphemes are converted to Thai phoneme sequences before they are then mapped to their closest matched English phoneme sequences. Finally, the best sequence of English graphemes is then hypothesized based on the phonemes. Charoenporn et al. [5] and Aroonmanakun et al [6] both deployed corpus-based statistical segmentation methods to segment Thai words into syllables. However, the former chose to perform the writing system translation step by relying on a set of hand-written rules based on the Romanization guide for Thai script defined by the Royal Institute in 1984 together with some deterministic mapping tables.

Obviously, it is difficult and time consuming for such heuristic approaches to handle the unsystematic character of Thai proper name Romanization. Therefore, we propose a data-driven approach that automatically learns the character mapping between Thai and English via a set of newly proposed units generically called "Gram". These units are distinct from typical syllable-based units in the aspect that each Gram contains attributes related to the writing systems (as well as the indirectly embedded pronunciations) of both languages at the same time.

3 Background on Thai Writing System

44 alphabets are used to represent 21 Thai consonant sounds. All of them can be used for consonantal phonemes in the syllable-initial position. Three of them can be combined with other consonants to form true consonant clusters. Although, theoretically, all 44 consonantal alphabets can be used to represent 8 syllable-final consonantal phonemes, some are more popular than the others. In many occasions, a string of multiple consonants could represent only one phoneme, and many occurrences are governed by exceptions rather than precise rules. 34 symbols, some of which are constructed from multiple alphabets, together with three of the alphabets used for the

consonants, are used to represent Thai vowel phonemes including monophthongs as well as diphthongs. 5 Tone symbols are superimposed to Thai syllabic representation to govern the tonal aspect of each syllable, which is also affects by the base consonantal alphabets representing the phoneme in the syllable-initial position.

4 Proposed Methods

4.1 Name Decomposition and Gram Accumulation

In this work, the spelling of each name is looked as if it is the concatenation of a sequence of basic units. The attributes of each one of these units are their alphabet sequences in both the original and the destination systems, which in this case are Thai and English respectively. These arrangements of these units will constrain the spelling of the names in both languages. Although resulting units with meaning are preferable, it is not a requirement for units to be meaningful. Therefore, these units are referred to by using a generic name called *Gram*s, in contrary to the name *Morph* adopted in many linguistic literatures. The string **A**:**B** will be used in this paper to refer to a Gram whose Thai spelling is **A** and its English spelling is **B**. Figure 1 shows some examples of the name decomposition.

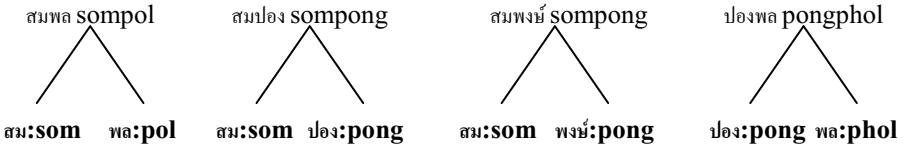


Fig. 1. Some examples of the name decomposition

Items in the Gram lexicon were added while names in the training corpus were analyzed. Each Gram must at least consist of alphabets for the corresponding syllable-initial phoneme and alphabets for the vowel phoneme. Although single-syllable Grams are preferable, multi-syllable Grams are also allowed to ensure that the decomposition of names do not produce any Grams that their Thai spellings only contain a single alphabet apart from two exceptions to be mentioned later. Such cases would have made the decomposition generate too many possible hypotheses. Furthermore, if the vowel alphabet of a syllable is a type of vowel that requires mandatory coda consonants, the alphabets for the consonant and the vowel cannot be separated. This is, again, to prevent the over-generation problem.

Building Gram lexicon is a tedious task if the decomposition is performed manually on a large name corpus. Here, we semi-automatically process names in our training/development set (Details about the name corpus used in this work will be described later in this paper.) in several batches of a couple thousands names. Grams obtained from one batch are added to the lexicon which will be used for the automatic decomposition of the names in the next batch. Manual adjustment is performed on the result of the automatic decomposition of each batch before new Grams are added to the lexicon. The process can be illustrated in Figure 2.

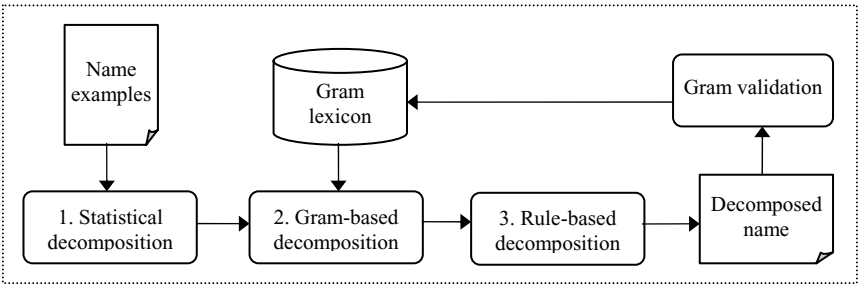


Fig. 2. Gram lexicon construction process

At the beginning of each batch, the Thai spelling of each name is decomposed into a syllable sequences using the method proposed by Aroonmanakun et al. [4]. Then, given the accumulated Grams from previous batches, a right-to-left searching is performed in the English alphabet string associated with each name in order to match each Thai syllable with English alphabets and at the same time, each resulting pair must replicates a Gram in the accumulated Gram lexicon. This search attempt can lead to three possible results: 1) A complete matching, 2) A single failure, and 3) A multiple failure.

A complete matching is when the name can be decomposed entirely into a sequence of Grams in the accumulated Gram lexicon. No manual adjustments are needed and no new Grams are added into the Gram lexicon.

From the result of the right-to-left searching, if the name can be partially decomposed into a sequence of Grams in the lexicon except for the first Thai syllable and some unmatched English alphabets at the beginning of the name. An expert judgment is required whether to pair up the Thai syllable with the remaining English alphabets and add a new Gram into the Gram lexicon. Otherwise, a manual adjustment has to be made on the entire name.

The last case is when more than one syllables cannot be represented using existing Grams. In this case, an alphabet-mapping decomposition is tried. This method of decomposition first tokenized the associated sequence of English alphabets so that each resulting token is either a group of alphabets representing the English vowel, a group of alphabets representing syllable-initial consonants, or a group of alphabets representing consonants in the coda positions. A simple algorithm based on a Thai-English alphabet-wise mapping is then used to combine some English tokens together so that it results in a one-to-one mapping between the Thai syllables and the English tokens. An expert judgment is then needed to verify the validity of the resulting Grams. Manual adjustments are sometimes needed before the new Grams are added to the Gram lexicon.

Although manual adjustments are still required in many cases, such manual burdens are less severe when the Gram lexicon grows larger. Note that in the first batch when the Gram lexicon is empty, the decomposition of every name will result in the multiple failure case and will be treated accordingly.

4.2 Gram-Based Romanization

With the Gram lexicon, A Thai name to be romanized can be decomposed into multiple sequences of Grams constrained by its corresponding Thai spelling. Statistical language and phase transition models are used to give a popularity score to each possible Gram sequence. The system picks the sequence with the highest score to be the hypothesis of choice. Then, Romanization can be completed simply by concatenating the English alphabets of the Grams in the corresponding sequence together.

Given a Thai name $\boldsymbol{R} = r_0 r_1 \ldots r_N$ where each r_i is a Thai alphabet and N is the number of alphabets in the Thai spelling of that name, the system first converts $\boldsymbol{R}$ to a sequence of K Thai alphabet strings, $\boldsymbol{T} = t_0 t_1 \ldots t_K$ where each t_i is the Thai spelling (string of Thai alphabets) of a Gram appearing in our Gram lexicon using a right-to-left longest matching approach.

Then, for the sequence $\boldsymbol{T}$ corresponding with the name $\boldsymbol{R}$, an associated popularity score is computed for each possible Romanization hypothesis, $\boldsymbol{E} = e_0 e_1 \ldots e_K$ where each e_i is the English spelling (string of English alphabets) of a Gram. Note that the length of a possible $\boldsymbol{E}$ must also be K since the mapping between each term in both sequences must be one-to-one. This also means that each of $t_i{:}e_i$ for $i = 0, 1, \ldots, K$ must be a Gram in the lexicon. Here, we deploy the conditional probability of the sequence $\boldsymbol{E}$ given $\boldsymbol{T}$. The MAP criterion is deployed such that we will choose the best hypothesis $\boldsymbol{E}^*$ for the given $\boldsymbol{T}$ such that:

$$\boldsymbol{E}^* = \arg\max_{\boldsymbol{E}} p(\boldsymbol{E} \mid \boldsymbol{T}) = \arg\max_{\boldsymbol{E}} p(\boldsymbol{T} \mid \boldsymbol{E})\, p(\boldsymbol{E})$$

We will refer to the term $p(\boldsymbol{E})$ as the "N-Gram score" which can be computed using the typical N-gram models of English alphabet strings. In this work, bigram models are created from the name decomposition examples in the training/development set. However, only the English alphabets of each Gram are considered. In other words, when counting the number of Gram pairs to obtain the bigram models, two different Grams considered the same unit if their English spellings are the same.

Similar to the problem of the machine translation problems in the sentence level, we view the term $p(\boldsymbol{T} \mid \boldsymbol{E})$ as the translation score. Assuming independency among the terms in $\boldsymbol{T}$ and that t_i only depends on e_i we have:

$$p(\boldsymbol{T} \mid \boldsymbol{E}) = p(t_0 t_1 \ldots t_K \mid e_0 e_1 \ldots e_K) = \prod_{i=0}^{K} p(t_i \mid e_i)\,,$$

and

$$p(t_i \mid e_i) = \frac{N(t_i : e_i)}{\sum_{\text{all } \tau} N(\tau : e_i)}\,,$$

where $N(t_i{:}e_i)$ is the number of the Gram $t_i{:}e_i$ in the training/development set and $\sum_{\text{all } \tau} N(\tau : e_i)$ is the total number of any Grams τ that their associated English spelling is e_i.

5 Experiment Details

5.1 Name Corpus

A corpus of Thai first names and surnames were used for the training of all statistical models as well as for evaluating the proposed method. The corpus was constructed from a database of the names of students registered at Chulalongkorn University, mostly, in the last 10 years. Foreign names were excluded from the corpus. The corpus contains Thai spellings of 178,612 names together with their English counterparts. 20% of the names were randomly picked as the test data set while the rest 80% were put in the training/development set, in which names were used for the building of our Gram lexicon, as well as to train statistical models for the N-gram scores as well as the translation scores mentioned earlier. The names in this set were also observed in the process of introducing our Grams.

6 Results and Discussion

6.1 Construction of the Gram Lexicon

Figure 3 compares the coverage percentages and the number of accumulated Grams (as the percentage of the total Grams in the final Gram lexicon) with the number of names analyzed.

As we can observe from the table, after analyzing the 22 batches of data in the training/development set, including manual adjustments that handle decomposition failures, 24,385 Grams cover 139,500 names in the set. Note that each of these Grams is constrained by both the Thai and English spelling. Our analysis also showed that, among these 24,385 Grams, there were around eight thousands unique tokens if only Thai alphabet sequences were considered, while the number is around twelve thousand if only English alphabet sequences were considered.

From figure 3, we can see that the number of Grams grows rather linearly with the increasing of the number of names after some Grams (which, in this case, are around 35% of the total Grams) are already in the lexicon. The coverage percentage also grows linearly with the increasing of the number of names already analyzed.

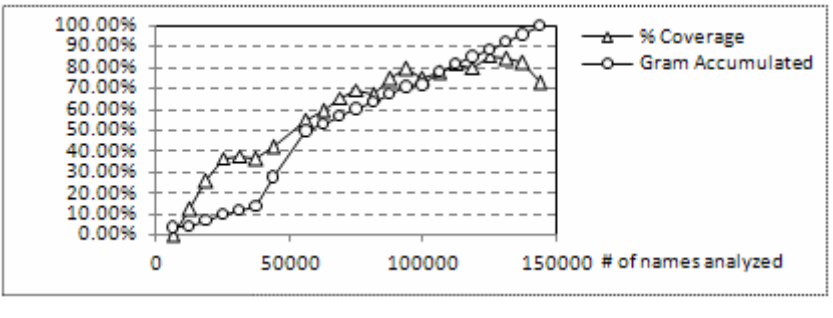


Fig. 3. The tendency of %Coverage and the number of accumulated Grams when more names are analyzed in batches

6.2 Romanization Results

Table 1 compares the Romanization accuracies of the baseline and the proposed methods evaluated on the 35,722 names in the test data set.

Table 1. Romanization accuracies evaluated on the test data set

	Baseline	Proposed Method	
		1st Variation	2nd Variation
Romanization Accuracy	18.20%	46.13%	43.66%

The Romanization accuracy percentage of the baseline method is around 18%. The fact that more than 80% of the names are romanized differently from what are listed in the test data set reflects the degree of complexity of Thai name Romanization. However, the highly-varied nature of the Thai name Romanization also exposes the weakness of the baseline method that it only focuses on the closest matching of the sounds between the two sound systems and it lacks the flexibility to handle real-life preferences. The proposed method yield the best performance of almost 50% evaluated on the same data set.

Figure 4 shows the recall rate percentages when *N*-best hypotheses are used. We can see from the figure that the 1st variation of our proposed method yields higher recall rates for every values of *N* tested. Both variations show a similar tendency where the recall rate grows from the values shown in table 4 (≈44% – 46%) to around 75% to 80% when 15-best hypotheses are used. Although the recall rate keeps increasing as the number of hypothesis increases, we can observe that the curves tend to saturate. Consequently, we can expect that using more hypotheses is not likely to push the recall rate any higher. It is worth mentioning that even though using *N*-best hypotheses does not single out the correct Romanization of a name, we can reasonably expect it to recall the correct one almost 80% of the time when applied to a search application in which not too many Romanization hypotheses from the proposed method are used as the search queries.

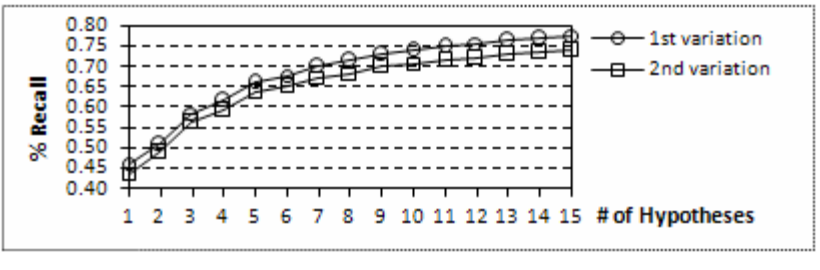


Fig. 4. Recall rate percentages of the proposed method when N-best hypotheses are used

7 Conclusion

This paper presents a Romanization method whose flexibility is more suitable for the Romanization of Thai names than a conventional Romanization method. Although it

might not be surprising that taking into consideration real-life preferences in the forms of a large number of Romanized name examples yield a better Romanization performance, practical statistical models have to be formulated in order for an automatic method to hypothesize the most likely results instead of having to rely on the burden of creating sets of heuristics from such a large name database. In this work, the proposed Gram-based models yield a satisfactory finding in terms of the Romanization results as well as the framework adopted in the construction of the Gram lexicon from examples of more than one-hundred thousand names.

References

[1] Poowarawan, Y.: Dictionary-based Thai Syllable Separation. In: Proceedings of the Ninth Electronics Engineering Conference (1986)
[2] Sornlertlamvanich, V.: Word Segmentation for Thai in a Machine Translation system (in Thai), Papers on Natural Language processing, NECTEC, Thailand (1995)
[3] Thanaruk, T., Thanasan, T., Duangrumol, P., Arunthep, S.: Non-Dictionary-Based Word Segmentation Using Local Context Statistics. In: Proceedings of the 5th Symposium on Natural Language Processing and Oriental COCOSDA Workshop, Hua Hin, Thailand, pp. 81–88 (May 2002)
[4] Aroonmanakun, W.: Collocation and Thai Word Segmentation. In: Proceedings of the Fifth Symposium on Natural Language Processing & The Fifth Oriental COCOSDA Workshop, pp. 68–75. Sirindhorn International Institute of Technology, Pathumthani (2002)
[5] Charoenporn, T., Chotimongkol, A., Sornlertlamvanich, V.: Automatic Romanization For Thai, Bangkok, Thailand (1999)
[6] Aroonmanakun, W., Rivepiboon, W.: A Unified Model of Thai Word Segmentation and Romanization. In: Proceedings of The 18th PACLIC, Tokyo, Japan (2004)
[7] Karoonboonyanan, T.: Standardization and Implementations of Thai Language. In: The Seminar on Enhancement of the International Standardization Activities in Asia Pacific Region (AHTS-1) held on at CICC, Japan (March 1999)

Budget Semi-supervised Learning

Zhi-Hua Zhou[1], Michael Ng[2], Qiao-Qiao She[1], and Yuan Jiang[1]

[1] National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing 210093, China
[2] Institute for Computational Mathematics
Hong Kong Baptist University, Hong Kong, China
{zhouzh,sheqq,jiangy}@lamda.nju.edu.cn, mng@math.hkbu.edu.hk

Abstract. In this paper we propose to study *budget semi-supervised learning*, i.e., semi-supervised learning with a resource budget, such as a limited memory insufficient to accommodate and/or process all available unlabeled data. This setting is with practical importance because in most real scenarios although there may exist abundant unlabeled data, the computational resource that can be used is generally not unlimited. Effective budget semi-supervised learning algorithms should be able to adjust behaviors considering the given resource budget. Roughly, the more resource, the more exploitation on unlabeled data. As an example, in this paper we show that this is achievable by a simple yet effective method.

1 Introduction

Previous studies on semi-supervised learning almost neglect the fact that although there exist abundant or even unlimited unlabeled data, the computational resource that can be used is generally not unlimited. In this paper, we propose to study *budget semi-supervised learning*, that is, semi-supervised learning with a resource budget. Roughly, given a labeled data set L and a large unlabeled data set U, for a concerned semi-supervised learning algorithm $Algo$, a computational resource Δ is needed to exploit all the data in U, yet the available resource is only Δ_a which is much smaller than Δ; try to enable $Algo$ to adapt to Δ_a.

Here, the budget can be memory storage, computational time cost, etc. For example, assume that a storage of $10^6 \times 10^6$ matrix is required for $Algo$ to exploit all available unlabeled data, yet the memory storage available is only able to accommodate a $10^5 \times 10^5$ matrix. Ideally, effective budget semi-supervised learning algorithms should be able to adjust their behaviors considering the given resource budget. Roughly, the more resource, the more exploitation on unlabeled data.

There were some consideration on limited budget of computational resource in some other fields, but in the area of semi-supervised learning, to the best of our knowledge, this is the first study on this topic. In this paper, using a kind of spectral analysis based algorithm, *cluster kernel* [1], as an example, we will show that the problem of budget semi-supervised learning can be tackled by a simple yet effective method which exploits advantages of known techniques and is validated in experiments.

The rest of this paper is organized as follows. Section 2 briefly introduces the cluster kernel algorithm [1]. Section 3 presents the Lank method. Section 4 reports on experimental results. Finally, Section 5 concludes.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 588–595, 2009.
© Springer-Verlag Berlin Heidelberg 2009

2 Background

Given labeled data $\{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_l, y_l)\}$ and unlabeled data $\{\boldsymbol{x}_{l+1}, \cdots, \boldsymbol{x}_{l+u}\}$, $l \ll u$, $n = l + u$. For simplicity, assume that $\boldsymbol{x} \in \mathbb{R}^m$ and $y \in \{0, 1\}$. We can construct a Gaussian kernel matrix K by considering the pairwise affinity of the data points, where nearby data points are assigned with relatively large edge weights. Now we consider the following cluster kernel algorithm [1]:

1. Let D denote the diagonal matrix whose elements are $D_{ii} = \sum_j K_{ij}$, and construct the matrix $L = D^{-1/2} K D^{-1/2}$.
2. Compute the eigen-decomposition $L = U \Lambda U^{\mathrm{T}}$, and assume that the eigenvalues are ordered as $\lambda_1 \geq \lambda_2 \geq \cdots$.
3. Apply a transfer function φ on λ. Let $\tilde{\lambda}_i = \varphi(\lambda_i)$, and construct $\tilde{L} = U \tilde{\Lambda} U^{\mathrm{T}}$.
4. Let $\tilde{D}$ denote the diagonal matrix whose elements are $\tilde{D}_{ii} = 1/\tilde{L}_{ii}$, and compute $\tilde{K} = \tilde{D}^{1/2} \tilde{L} \tilde{D}^{1/2}$.

The transfer function φ can take different forms. Here, the *poly-step* transfer function which has achieved the best performance in [1] is adopted:

$$\tilde{\lambda}_i = \begin{cases} \sqrt{\lambda_i} & i > h \ (= l + 9) \\ \lambda_i^2 & i \leq h \ (= l + 9) \end{cases} \tag{1}$$

For an unseen test example $\boldsymbol{x}$, we can approximate it with a liner combination of the labeled and unlabeled training examples by

$$\boldsymbol{\alpha}^0 = \arg\min_{\boldsymbol{\alpha}} ||\Phi(\boldsymbol{x}) - \sum\nolimits_{i=1}^{n} \alpha_i \Phi(\boldsymbol{x}_i)|| = K^{-1} \boldsymbol{v} \ , \tag{2}$$

where $\boldsymbol{v}_i = K(\boldsymbol{x}, \boldsymbol{x}_i)$ since K is Gaussian, and Φ is the feature map corresponding to K, i.e., $K(\boldsymbol{x}, \boldsymbol{x}^{\mathrm{T}}) = (\Phi(\boldsymbol{x}) \Phi(\boldsymbol{x}^{\mathrm{T}}))$. Thus,

$$\tilde{K}(\boldsymbol{x}, \boldsymbol{x}_i) = (\tilde{K} \boldsymbol{\alpha}^0)_i = (\tilde{K} K^{-1} \boldsymbol{v})_i \ . \tag{3}$$

3 The Lank Method

The storage cost of the cluster kernel algorithm is $O(n^2)$ and the computational cost is roughly $O(n^3)$. Now, suppose we do not have unlimited resource, e.g., we have a storage budget which is much smaller than $O(n^2)$, what can we do?

The large storage is caused by the use of a full-connection graph. If we consider a k-graph where each node is only connected to its k-nearest neighboring nodes, the storage will reduce to $O(kn)$. Actually, using k-graph to reduce the storage cost is not new [2,5]. We call this as *k-approx*. This method is reasonable since in most cases the affinity among neighboring examples is important while that among far examples are not very meaningful. To ensure that the resulting matrix is symmetric, we simply set the symmetric component of any non-zero component to the same non-zero value, and thus, for the worst case the storage is $O(2kn)$.

The 2nd step of the cluster kernel algorithm, however, is still expensive in storage even after using k-approx. In addition, using k-approx alone could not significantly

reduce the computational cost. Note that K is large and symmetric, and assume that the m ($m \ll n$) largest eigenvalues and the corresponding eigenvectors are sufficient. Thus, we can solve the problem through the famous Lanczos method [4] and reduce the computational cost to $O(mn^2)$. Given a symmetric $n \times n$ matrix K and an n-dimensional vector $\hat{\boldsymbol{z}}$ having unit 2-norm, the Lanczos method works as follows:

$\boldsymbol{w}_1 = \hat{\boldsymbol{z}}; \quad a(1) = \boldsymbol{w}_1^{\mathrm{T}} K \boldsymbol{w}_1; \quad b(1) = 0;$
For $i = 1, 2, \cdots, m-1$ do
$\quad \hat{\boldsymbol{w}}_{i+1} = K\boldsymbol{w}_i - a(i)\boldsymbol{w}_i - b(i)\boldsymbol{w}_{i-1}; \quad \boldsymbol{w}_{i+1} = \hat{\boldsymbol{w}}_{i+1}/\|\hat{\boldsymbol{w}}_{i+1}\|_2;$
$\quad b(i+1) = \|\hat{\boldsymbol{w}}_{i+1}\|_2; \quad a(i+1) = \boldsymbol{w}_{i+1}^{\mathrm{T}} K \boldsymbol{w}_{i+1}$
End

The method computes an $m \times m$ symmetric tridiagonal matrix T_m with the property that the eigenvalues of K contains the eigenvalues of T_m. The diagonal and subdiagonal elements of T_m are stored in $a(1:m)$ and $b(1:m-1)$ respectively. Note that K is not altered during the entire process. If K has an average of about k non-zero values per row then approximately $(2k+8)n$ flops are involved in a single Lanczos step. Upon termination the eigenvalues of T_m can be found using the symmetric tridiagonal QR algorithm [3]. The main computational task is the matrix-vector multiplication $K\boldsymbol{u}$ with cost $O(n^2)$. When k-approx is used, each matrix-vector product is $O(kn)$ operations. So, the computational cost can be further reduced to $O(mkn)$. It is noteworthy that the whole Lanczos process can be implemented with just two n-vectors of storage.

3.1 Analysis

The k-approx. First, by using k-approx we get a matrix $K^{(k)}$ which is different from K. If from $K^{(k)}$ and K we can solve the same eigenvectors, the remaining steps of the cluster kernel algorithm are almost untouched. So, we can analyze the influence of the use of k-approx by studying the eigenvectors solved from $K^{(k)}$ and K.

Once K is got, D is determined and $L = D^{-1/2} K D^{-1/2}$ can be derived. Assume that the eigen-decomposition of L is given by $L = U \Lambda U^{\mathrm{T}} = \sum_{i=1}^{n} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^{\mathrm{T}}$, where $\Lambda = \mathrm{diag}(\lambda_i)$ is a diagonal matrix whose diagonal elements are the eigenvalues of L, and $U = [\boldsymbol{u}_1 \boldsymbol{u}_2 \cdots \boldsymbol{u}_n]$ is an orthogonal matrix whose ith column is the eigenvector corresponding to λ_i. Assume that the eigenvalues are ordered as $\lambda_1 \geq \cdots \geq \lambda_n$.

Suppose we get $K^{(k)}$ by applying k-approx with a specific k value, the eigenvalues of $L^{(k)} = D^{-1/2} K^{(k)} D^{-1/2}$ are ordered as $\lambda_1^{(k)} \geq \cdots \geq \lambda_n^{(k)}$, with corresponding eigenvectors $\boldsymbol{u}_1^{(k)}, \cdots, \boldsymbol{u}_n^{(k)}$. That is, $L^{(k)} = U^{(k)} \Lambda^{(k)} (U^{(k)})^{\mathrm{T}} = \sum_{i=1}^{n} \lambda_i^{(k)} \boldsymbol{u}_i^{(k)} (\boldsymbol{u}_i^{(k)})^{\mathrm{T}}$.

We can measure the difference between L and $L^{(k)}$ by considering $E^{(k)} = K - K^{(k)}$. It is obvious that when $k = n$, $E^{(k)}$ is a zero matrix, i.e., $K^{(k)} = K$. When k is large, $\|E^{(k)}\|_F$ is small. We have the following results [7] about the relative perturbation bounds for eigenvalues between L and $L^{(k)}$:

$$\sqrt{\sum_{i=1}^{n} \frac{|\lambda_i - \lambda_i^{(k)}|}{|\lambda_i \lambda_i^{(k)}|}} \leq \|V\|_2 \|V^{(k)}\|_2 \frac{\|K^{-1}\|_2 \|E^{(k)}\|_F}{\sqrt{1 - \|K^{-1}\|_2 \|E^{(k)}\|_2}}, \tag{4}$$

where the following matrix decompositions are used:

$$V^{\mathrm{T}}G^{\mathrm{T}}GV = |\Lambda|; \quad V^{\mathrm{T}}JV = I; \quad K = U|\Lambda|^{1/2}J|\Lambda|^{1/2}U^{\mathrm{T}}; \quad G = D^{-1/2}U|\Lambda|^{1/2};$$

$$(V^{(k)})^{\mathrm{T}}(G^{(k)})^{\mathrm{T}}G^{(k)}V^{(k)} = |\Lambda^{(k)}|; \quad (V^{(k)})^{\mathrm{T}}J^{(k)}V^{(k)} = I;$$

$$K^{(k)} = U^{(k)}|\Lambda^{(k)}|^{1/2}J^{(k)}|\Lambda^{(k)}|^{1/2}(U^{(k)})^{\mathrm{T}}; \quad G^{(k)} = D^{-1/2}U^{(k)}|\Lambda^{(k)}|^{1/2}\,.$$

When k is large ($\|E^{(k)}\|_F$ is small), the left-hand of Eq. 4 is small, and thus the distances between the eigenvalues of L and $L^{(k)}$ is small. We have the following results about the relative perturbation bounds for eigenvectors between L and $L^{(k)}$:

$$\|\sin\Theta^{(k)}\|_F \leq \frac{\|K^{-1}\|_2\|E^{(k)}\|_F}{\sqrt{1-\|K^{-1}\|_2\|E^{(k)}\|_2}} \cdot \frac{\|V_1\|_2\|V_1^{(k)}\|_2}{\mathrm{rg}(\Lambda_1^{(k)},\Lambda_2^{(k)})}\,, \tag{5}$$

where $\mathrm{rg}(\cdot)$ expresses the relative gap, $\Theta^{(k)}$ denotes diagonal matrix with canonical angles between the subspace generated by $Q_1 = [\boldsymbol{u}_{i_1}\cdots\boldsymbol{u}_{i_l}]$ and that generated by $Q_1^{(k)} = [\boldsymbol{u}_{i_1}^{(k)}\cdots\boldsymbol{u}_{i_l}^{(k)}]$ with

$$L = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}\begin{pmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{pmatrix}\begin{pmatrix} Q_1 \\ Q_2 \end{pmatrix} \text{ and } L^{(k)} = \begin{pmatrix} Q_1^{(k)} & Q_2^{(k)} \end{pmatrix}\begin{pmatrix} \Lambda_1^{(k)} & 0 \\ 0 & \Lambda_2^{(k)} \end{pmatrix}\begin{pmatrix} Q_1^{(k)} \\ Q_2^{(k)} \end{pmatrix}.$$

The matrices $V = [V_1\ V_2]$ and $V^{(k)} = [V_1^{(k)}\ V_2^{(k)}]$ are defined similarly based on the corresponding partitions [7]. When k is large, the left-hand of Eq. 5 is small, and thus the canonical angles between the subspaces Q_1 and $Q_1^{(k)}$ is small.

The Lanczos Process. Given a starting vector $\hat{\boldsymbol{z}}$, the Lanczos process will generate successively an orthonormal basis $\boldsymbol{w}_1,\cdots,\boldsymbol{w}_m$ of a subspace. The vectors $\boldsymbol{w}_i$'s form an orthonormal basis of the Krylov subspace $E_m = \text{ span }\{\hat{\boldsymbol{z}}, K\hat{\boldsymbol{z}},\cdots,K^{m-1}\hat{\boldsymbol{z}}\}$. We have $W_m^{\mathrm{T}}KW_m = T_m$, so that T_m is the matrix representation of the projection of K onto the Krylov subspace E_m with respect to the basis $W_m = [\boldsymbol{w}_1, \boldsymbol{w}_2,\cdots,\boldsymbol{w}_m]$.

Specifically, the approximate solution $\boldsymbol{z}_m$ of the linear system $K\boldsymbol{z} = \boldsymbol{v}$ (or $\boldsymbol{z} = K^{-1}\boldsymbol{v}$) in Eq. 2 can be found by noting that it belongs to the affine subspace $\hat{\boldsymbol{z}}+E_m$ and its residual vector $\boldsymbol{v}-K\boldsymbol{z}_m$ is orthogonal to E_m. If we denote by J_m the orthogonal projector onto E_m, this means that the Lanczos process solves the approximate problem: To find $\boldsymbol{z}_m \in \hat{\boldsymbol{z}}+E_m$ such that $J_m(\boldsymbol{v}-K\boldsymbol{z}_m) = 0$, i.e., to find $\boldsymbol{z}_m = \hat{\boldsymbol{z}}+\boldsymbol{g}$, $\boldsymbol{g} \in E_m$. The desired approximation can be written as $\boldsymbol{z}_m = \hat{\boldsymbol{z}} + W_mT_m^{-1}W_m^{\mathrm{T}}(\boldsymbol{v}-K\hat{\boldsymbol{z}})$. Here we state how well the result approximates the $K^{-1}\boldsymbol{v}$ in Eq. 2 [6]:

Theorem 1. *Assume that K is symmetric positive definite with the smallest eigenvalue λ_n and largest eigenvalue λ_1. Let the projected residual $J_m(\boldsymbol{v}-K\hat{\boldsymbol{z}})$ be expressed in the Krylov basis as $J_m(\boldsymbol{v}-K\hat{\boldsymbol{z}}) = \sum_{i=1}^{n}\eta_iK^{i-1}\hat{\boldsymbol{z}}$. Then the approximation $\boldsymbol{z}_m$ obtained from the Lanczos-Galerkin projection process is such that $\|\boldsymbol{v}-K\boldsymbol{z}_m\|_{K^{-1}} = \|(I-J_m)(\boldsymbol{v}-K\hat{\boldsymbol{z}})\|_{K^{-1}} + \epsilon$, where $|\epsilon| \leq \frac{\|\eta_1\boldsymbol{w}_1\|_{K^{-1}}}{j_m(\gamma)}$, in which $\gamma = \frac{\lambda_1+\lambda_n}{\lambda_1-\lambda_n}$ and j_m represents the Chebyshev polynomial of the first kind of degree m.*

If $\boldsymbol{v}-K\hat{\boldsymbol{z}}$ belongs to the Krylov subspace E_m, $\|(I-J_m)(\boldsymbol{v}-K\hat{\boldsymbol{z}})\|_{K^{-1}}$ vanishes. Theorem 1 tells us that in this case the method will provide a good accuracy when η_1

is not too large. It is clear that when m becomes larger, the term $|\epsilon|$ becomes smaller as $|j_m(\gamma)|$ increases. On the other hand, the Lanczos process can compute the eigenvalues $\lambda_{1(m)} \geq \cdots \geq \lambda_{m(m)}$ of T_m and the corresponding eigenvectors $\boldsymbol{u}_{1(m)}, \cdots, \boldsymbol{u}_{m(m)}$. Theorem 2 [3] indicates that $\lambda_{i(m)}$ and $\boldsymbol{u}_{i(m)}$ converge to λ_i and $\boldsymbol{u}_i$, respectively.

Theorem 2. *The Rayleigh-Ritz approximations* $(\lambda_{i(m)}, q_{i(m)})$ *to* (λ_i, q_i) *satisfy* $0 \leq \lambda_i - \lambda_{i(m)} \leq (\lambda_i - \lambda_n)\left(\frac{c_{i(m)}}{J_{n-i}(\gamma_i)}\right)^2 \tan^2\theta(\boldsymbol{q}_i, \hat{\boldsymbol{z}})$, *where* $c_{i(m)} = \prod_{l=1}^{i-1} \frac{\lambda_{l(m)}-\lambda_n}{\lambda_{l(m)}-\lambda_i}$, $c_{1(m)} = 1$, $\gamma_i = 1 + \frac{2(\lambda_i-\lambda_{i+1})}{\lambda_{i+1}-\lambda_n}$, $\sin^2\theta(\boldsymbol{q}_i, \boldsymbol{q}_{i(m)}) \leq \frac{\lambda_i-\lambda_{i(m)}+\sum_{l=1}^{i-1}(\lambda_l-\lambda_{l(m)})\sin^2\theta(\boldsymbol{q}_l,\boldsymbol{q}_{l(m)})}{\lambda_i-\lambda_{i+1}}$.

According to the theorem, when m is large, both the approximation of eigenvalues and eigenvectors are close to the original eigenvalues and eigenvectors.

The Combination. It can be found from Eq. 3 that the final classification is made by $\tilde{K}K^{-1}\boldsymbol{v}$, so we analyze the bound on $\tilde{K}K^{-1}\boldsymbol{v}$ when the approximate eigenvalues and eigenvectors obtained by the Lanczos process are used in calculation. For simplicity, the normalization is not considered in the analysis. Assume that the eigen-decomposition of T_m is given by $T_m = S_m\Omega_m S_m^{\mathrm{T}}$ where Ω_m is an $m \times m$ diagonal matrix with entries given by $\lambda_{i(m)}$. We define $\tilde{K}_m = W_m^{\mathrm{T}} S_m \varphi(\Omega_m) S_m^{\mathrm{T}} W_m$ and $K_m^{-1} = W_m T_m^{-1} W_m^{\mathrm{T}}$. Here $\varphi(\Omega_m)$ is also an $m \times m$ diagonal matrix whose entries are given by $\varphi(\lambda_{i(m)})$, see Eq. 1. By using the Lanczos process, we know that K_m and $\tilde{K}_m$ are close to K and $\tilde{K}$ respectively when m is large.

Overall, we would like to know the difference between $\tilde{K}K^{-1}\boldsymbol{v}$ and $\tilde{K}_m^{(k)}\left(K_m^{(k)}\right)^{-1}\boldsymbol{v}$. Here $\tilde{K}_m^{(k)}$ is the kernel matrix obtained by the k-approx and Lanczos processes. We find that when k (the kernel matrix density) and m (the number of Lanczos steps) are large, the difference between $\tilde{K}K^{-1}\boldsymbol{v}$ and $\tilde{K}_m^{(k)}(K_m^{(k)})^{-1}\boldsymbol{v}$ is small. More precisely, assume that $\|K^{-1}\|_2$, $\|\tilde{K}_m\|_2$, $\|K_m^{-1}\|_2$, $\|\tilde{K}_m^{(k)}\|$ and $\|(\tilde{K}_m^{(k)})^{-1}\|$ are bounded, then we note that $\tilde{K}K^{-1}\boldsymbol{v} - \tilde{K}_m K_m^{-1}\boldsymbol{v} = (\tilde{K} - \tilde{K}_m)K^{-1}\boldsymbol{v} + \tilde{K}_m K^{-1}(K_m - K)K_m^{-1}\boldsymbol{v}$, and $\tilde{K}_m K_m^{-1}\boldsymbol{v} - \tilde{K}_m^{(k)}(K_m^{(k)})^{-1}\boldsymbol{v} = (\tilde{K}_m - \tilde{K}_m^{(k)})K_m^{-1}\boldsymbol{v} + \tilde{K}_m^{(k)}K_m^{-1}(K_m^{(k)} - K_m)(K_m^{(k)})^{-1}\boldsymbol{v}$. According to the results in Sections 3.2.1 and 3.2.2, we have

$$\begin{aligned}\|\tilde{K}K^{-1}\boldsymbol{v} - \tilde{K}_m^{(k)}(K_m^{(k)})^{-1}\boldsymbol{v}\| \leq{}& \|\tilde{K} - \tilde{K}_m\|\|K^{-1}\boldsymbol{v}\| + \|\tilde{K}_m K^{-1}\|\|K_m - K\|\|K_m^{-1}\boldsymbol{v}\| \\ &+ \|\tilde{K}_m - \tilde{K}_m^{(k)}\|\|K_m^{-1}\boldsymbol{v}\| + \|\tilde{K}_m^{(k)}K_m^{-1}\|\|K_m^{(k)} - K_m\|\|(K_m^{(k)})^{-1}\boldsymbol{v}\| .\end{aligned}$$

The two terms $\|K_m - K\|$ and $\|\tilde{K} - \tilde{K}_m\|$ is small when m is large. Similarly, $\|K_m^{(k)} - K_m\|$ and $\|\tilde{K}_m - \tilde{K}_m^{(k)})\|$ is small when k is large. Thus, the difference between $\tilde{K}K^{-1}\boldsymbol{v}$ $\tilde{K}_m^{(k)}(K_m^{(k)})^{-1}\boldsymbol{v}$ is small when both m and k are large.

Note that Eqs. 4 and 5 imply that by using k-approx, the differences between the eigenvalues/eigenvectors and those of using the full-connection graph are small when k is large. We must note that, however, classification error is not necessarily proportional to eigen-system approximation error. On the other hand, using the full-connection graph does not necessarily lead to the optimal classification performance. In practice, if there are sufficient labeled training examples, cross validation can be used to select the optimal k value from those tolerated by the resource budget; otherwise it is not bad to use the largest k value tolerated by the budget.

3.2 The Algorithm

From the above analysis we have the Lank (Lanczos with k-approx) algorithm:

1. Decide the largest k and m tolerated by the given budget.
2. Obtain $K^{(k)}$ by constructing the k-nearest neighbor affinity matrix and setting the symmetric component of non-zero components to the same non-zero value.
3. Apply the Lanczos process to $K^{(k)}$ to obtain the largest m eigenvalues $\lambda_1^{(k)}, \cdots, \lambda_m^{(k)}$ and corresponding eigenvectors $\boldsymbol{q}_1^{(k)}, \cdots, \boldsymbol{q}_m^{(k)}$.
4. Apply the transfer function (e.g., Eq. 1) to the eigenvalues, and use the transferred eigenvalues $\tilde{\lambda}_1^{(k)}, \cdots, \tilde{\lambda}_m^{(k)}$ and corresponding eigenvectors to construct $\tilde{L}^{(k,m)} = \sum_{i=1}^{m} \tilde{\lambda}_i^{(k)} \boldsymbol{q}_i^{(k)} (\boldsymbol{q}_i^{(k)})^{\mathrm{T}}$.
5. Let $\tilde{D}$ denote the diagonal matrix whose elements are $\tilde{D}^{ii} = 1/\tilde{L}_{ii}^{(k,m)}$, compute $\tilde{K}^{(k,m)} = \tilde{D}^{1/2} \tilde{L}^{(k,m)} \tilde{D}^{1/2}$.

For test example $\boldsymbol{x}$, similar to Eq. 3, we have

$$\tilde{K}^{(k,m)}(\boldsymbol{x}, \boldsymbol{x}_i) = (\tilde{K}^{(k,m)} \boldsymbol{\alpha}^0)_i = (\tilde{K}^{(k,m)} (K^{(k)})^{-1} \boldsymbol{v})_i \ .$$

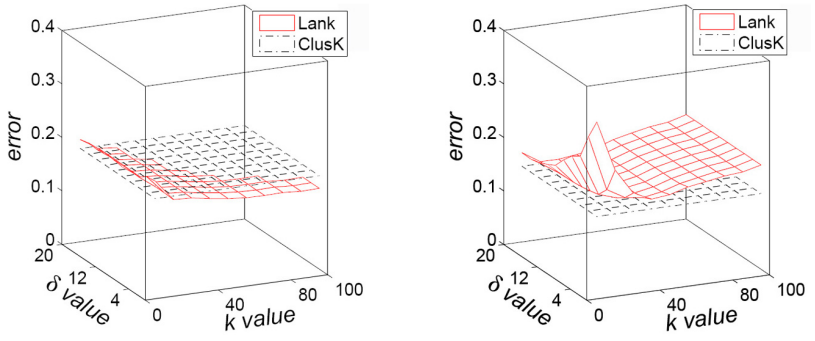
Suppose each integer costs 4 bytes and each double float costs 8 bytes in storage; this is popular in current machines. Given n examples with dimensionality d, the storage for these examples is $dn \times 8$ bytes. After applying k-approx, the matrix $K^{(k)}$ which contains $2kn$ number of non-zero entries is generated. Considering both the storage for the non-zero values and their indices, the required storage is roughly $2kn \times 8 + 2 \times 2kn \times 4 = 4kn \times 8$ bytes. So, up to now the storage required is $(d + 4k)n \times 8$ bytes. Now considering the Lancozs process, for storing the m largest eigenvalues and corresponding eigenvectors we need $(m + mn) \times 8$ bytes. Since we use $K^{(k)}$, the entire storage for the Lanczos process is $(m + mn + 4kn) \times 8$ bytes. When $m \ll n$, the storage required is roughly $((m + 4k)n) \times 8$ bytes. So, overall the storage required by the Lank method for exploiting all the labeled and unlabeled examples is $(\max(m, d) + 4k)\, n \times 8$ bytes. Since d and n are known, by assuming $m = \delta k$ (δ is a parameter), we can get the estimate of the largest k tolerated by the budget.

4 Experiments

4.1 When Original Cluster Kernel Can Work

First, we study that when there are sufficient resource for the original cluster kernel algorithm [1] (abbreviated as ClusK in the following) to exploit all available unlabeled examples, how well the Lank method approximates the performance of ClusK.

For this purpose, we run experiments on two small-scale UCI data sets, *australian* and *kr-vs-kp*. On each data set we randomly pick 10% examples to use as labeled training examples and regard the remaining ones as unlabeled examples. On both data sets the resource are sufficient for ClusK to utilize all examples, and we compare ClusK and Lank in a transductive setting. According to [1], σ of Gaussian is set to 0.55 for both algorithms. We repeat the experiments for ten times with random labeled/unlabeled partitions, and the average results are shown in Fig. 1.



(a) On *australian* (b) On *kr-vs-kp*

Fig. 1. Comparing Lank and the original cluster kernel (denoted by ClusK) when there are sufficient resource for the original cluster kernel to use all unlabeled data

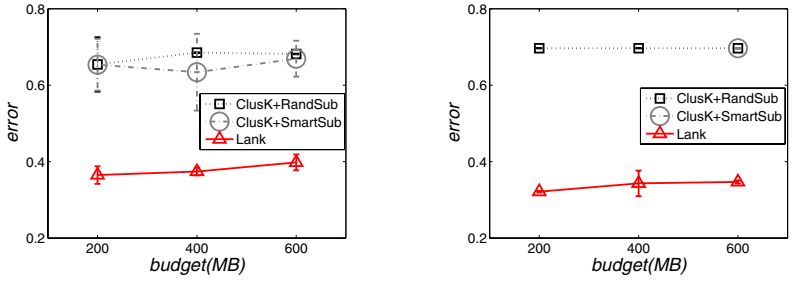
To study how well the Lank method approximates ClusK, we present in Fig. 1 the performance of Lank under different k and m ($m = \delta k$) configurations. Note that the number of examples in *kr-vs-kp* is about five times of that in *australian*, and so a k value being large for the latter is not necessarily large for the former. We can see that except when both k and m are small on *kr-vs-kp*, Lank approximates ClusK well on both data sets and on most parameter configurations.

4.2 When Original Cluster Kernel Cannot Work

Next, we study how the algorithms work given a resource budget. We run experiments on the SRAA data set (http://www.cs.umass.edu/∼mccallum/code-data.html) which contains 70K examples each described by 300 features. The original data set has four classes; we merge the two *real* classes and the two *simulate* classes, resulting in a two-class classification task. We randomly pick 1K or 5K examples to use as labeled training examples and regard the remaining ones as unlabeled examples. In the experiments we evaluate the performances under three storage budgets, 200MB, 400MB and 600MB.

The original ClusK cannot deal with such a large data set even when the largest budget (i.e., 600MB) were allocated. So, we facilitate it with the two sampling methods described in [2]. Briefly, *RandSub* randomly samples some unlabeled examples to use; *SmartSub* first uniformly samples some unlabeled examples to get an estimate of the decision surface, and then chooses the examples near that estimated surface to use.

We use the calculation described at the end of Section 3.3 to estimate k and m. Here, we simply set $\delta = 0.5k$. ClusK facilitated with RandSub or SmartSub is denoted by *ClusK+RandSub* and *ClusK+SmartSub*, respectively; both use the largest number of unlabeled examples that can make the algorithms executable under the given storage budget. The parameters of the sampling methods are set to the values recommended in [2]. The experiments are repeated for ten times and the results are shown in Fig. 2. Note that when there are 5K labeled training examples, ClusK+SmartSub cannot work given the budget of 200MB or 400MB, because the budget is not sufficient for running the SmartSub method. Fig. 2 shows that Lank is a better choice under all budgets.



(a) 1K labeled training examples (b) 5K labeled training examples

Fig. 2. Comparing Lank, ClusK+RandSub and ClusK+SmartSub. The original ClusK cannot deal with such a large data set even when the largest budget (i.e., 600MB) were allocated.

5 Conclusion

This paper proposes to study *budget semi-supervised learning*. The key is that, given different budgets, even for the same data, the behaviors of the algorithm should be different. Roughly speaking, the more resource, the more exploitation on unlabeled data. Considering that algorithms relying on spectral analysis suffer seriously from budget resource, we present a simple yet effective method which is able to adapt such kind of algorithms to a given budget. In order to show that the goal of budget semi-supervised learning is tractable, the presented method utilizes some well-known simple techniques. It is for sure that new elaborate methods will be attractive and lead to a better performance. This will become a fruitful topic in future research.

Acknowledgements. This research was supported by NSFC (60635030, 60721002), JiangsuSF (BK2008018), HKRGC (201508) and HKBU FRGs.

References

1. Chapelle, O., Weston, J., Schölkopf, B.: Cluster kernels for semi-supervised learning. In: NIPS 15, pp. 585–592 (2003)
2. Delalleau, O., Bengio, Y., Roux, N.L.: Large-scale algorithms. In: Semi-Supervised Learning, pp. 87–96. MIT Press, Cambridge (2006)
3. Golub, G.H., Van Loan, C.F.: Matrix Computations, 3rd edn. Johns Hopkins University Press, Baltimore (1996)
4. Lanczos, C.: An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. Journal of Research of the National Bureau of Standards 45(4), 255–282 (1950)
5. Pfahringer, B., Leschi, C., Reutemann, P.: Scaling up semi-supervised learning: An efficient and effective LLGC variant. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS, vol. 4426, pp. 236–247. Springer, Heidelberg (2007)
6. Saad, Y.: On the rates of convergence of the lanczos and the block-Lanczos methods. SIAM Journal of Numerical Analysis 17(5), 687–706 (1980)
7. Stewart, G.W., Sun, J.-G. (eds.): Matrix Perturbation Theory. Academic Press, New York (1990)

When does Co-training Work in Real Data?

Charles X. Ling[1], Jun Du[1], and Zhi-Hua Zhou[2]

[1] Department of Computer Science
The University of Western Ontario, London, Ontario, N6A 5B7, Canada
[2] National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing, 210093, China
cling@csd.uwo.ca, jdu42@csd.uwo.ca, zhouzh@lamda.nju.edu.cn

Abstract. Co-training, a paradigm of semi-supervised learning, may alleviate effectively the data scarcity problem (i.e., the lack of labeled examples) in supervised learning. The standard two-view co-training requires the dataset be described by two views of attributes, and previous theoretical studies proved that if the two views satisfy the sufficiency and independence assumptions, co-training is guaranteed to work well. However, little work has been done on how these assumptions can be empirically verified given datasets. In this paper, we first propose novel approaches to verify empirically the two assumptions of co-training based on datasets. We then propose simple heuristic to split a single view of attributes into two views, and discover regularity on the sufficiency and independence thresholds for the standard two-view co-training to work well. Our empirical results not only coincide well with the previous theoretical findings, but also provide a practical guideline to decide when co-training should work well based on datasets.

1 Introduction

Co-training, a paradigm of semi-supervised learning, has drawn considerable attentions and interests recently (see, for example, [1,2] for review). The standard two-view co-training [3] assumes that there exist two disjoint sets of features or views that describe the data.[1] The standard co-training utilizes an initial (small) labeled training dataset and a (large) set of unlabeled data from the same distribution, and it works roughly as follows [3]. Two separate classifiers are first trained on the initial labeled training dataset using the two views respectively. Then, alternately, each classifier classifies the unlabeled data, chooses the few unlabeled examples whose labels it predicts most confidently, and adds those examples and the predicted labels to the training dataset of the other classifier. The classifiers are retrained, and the process repeats, until some stopping criterion is met. That is, the two classifiers "teach" each other with the additional

[1] Another form of co-training, called the single-view co-training in our paper, generates diverse learners on the single view of attributes [4,5]. In this paper, we mainly study the standard two-view co-training [3], which will be referred to as two-view co-training, or simply co-training in the paper.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 596–603, 2009.
© Springer-Verlag Berlin Heidelberg 2009

examples whose labels are given by the other classifier to improve the classification accuracy, compared to a classifier learned only from the initial labeled training data.

Two assumptions are proposed for co-training to work well [3]. The first one assumes that the views are sufficient; that is, each view (thus also the combined view) is sufficient to predict the class perfectly. We call it the *sufficiency assumption*. The second assumption requires that the two views be conditionally independent; that is, the two views are independent given the class. We call it the *independence assumption*. Theoretical results have shown that if the sufficiency and independence assumptions are satisfied, co-training is guaranteed to work well. (The assumptions can be relaxed for co-training to still work well [6,7]. Nevertheless, the sufficiency and independence assumptions are a "sufficient condition" for co-training to work well). In addition, the two-view co-training has been applied quite successfully to many real-world tasks, such as statistical parsing [8], noun phrase identification [9], and image retrieval [10].

However, the two assumptions that guarantee co-training to work well may not be true in most real-world applications. Given a real-world dataset with two views of attributes, how can we judge if the two-view co-training would work well? How can we verify if the sufficiency and independence assumptions are satisfied to guarantee co-training to work well? If the real-world dataset has only one view, can the two-view co-training still work? This paper is our first attempt to answer these questions.

2 Verifying Co-training Assumptions Empirically

Given a whole dataset (with labels) and two views of attributes ($X = x_1, \ldots, x_m$ and $Y = y_1, \ldots, y_n$), how can we verify if the two assumptions on sufficiency and independence for the standard co-training are satisfied? If the assumptions are satisfied, co-training is guaranteed to work well, and thus can be applied. Note that sometimes the domain knowledge can ensure the satisfaction of the two assumptions, but in most real-world applications, such assumptions cannot be guaranteed. Thus it is important that these assumptions be empirically verified based on the dataset given. Here we will use the whole labeled dataset (or a very large training set) that *represents the learning task* to verify the two assumptions. This is because the theoretical assumptions on sufficiency and independence are based on the whole domain (for example, it is assumed that there exist target functions that map from the single view, the X view, and the Y view perfectly [3]).

The sufficiency assumption is relatively easy to verify. Sufficiency means that $X \times Y$ can accurately predict the class, so can X and Y individually. We can build a classifier to estimate the accuracy on the whole dataset D using $X \times Y$ with the 10-fold cross-validation. We denote this accuracy as p. The sufficiency says that p should be close to 1. Note that the theoretical results assume that there exist (target) functions that map from $X \times Y$, X, and Y to the class label perfectly. As we are verifying the assumption empirically, we use learning algorithms on the whole dataset to establish if such functions exist or not. Similarly, we build

a classifier using attributes in X to estimate the accuracy (call it p_x) of X predicting the class, and accuracy (call it p_y) of Y predicting the class.

Thus, the sufficiency assumption of co-training can be defined as: there exists a small positive number δ_1 (such as 0.1) such that

$p > 1 - \delta_1$,
$p_x > 1 - \delta_1$, and
$p_y > 1 - \delta_1$.

We call δ_1 the *sufficiency threshold*. In Section 3.2, we will discover ranges of δ_1 that make co-training work well.

Conditional independence is a bit harder to verify. It means that given the class, the two views are independent. One way to verify this is, for each class label, if each x_i is independent of Y, and each y_i is independent of X. To verify if x_i is independent of Y empirically, we build a classifier (or many classifiers) to predict x_i using Y on the whole dataset. If x_i is independent of Y, then Y cannot predict x_i well — not better than the default accuracy of x_i. Again we establish empirically if Y can predict x_i better than its default accuracy on the whole dataset. Assume that the 10-fold cross-validated accuracy of Y predicting x_i on D is p_{x_i}, then it should not be much larger than the default accuracy of x_i — the accuracy (denoted as p'_{x_i}) of the majority value of the class. The same is true for using X to predict y_j. Thus, the independence assumption can be defined as: there exists a small positive number δ_2 (such as 0.1) such that for each class value

$p_{x_i} < p'_{x_i} + \delta_2$ for all $1 \leq i \leq m$, and
$p_{y_i} < p'_{y_i} + \delta_2$ for all $1 \leq i \leq n$.

We call δ_2 the *independence threshold*. We will establish the ranges of δ_2 to make co-training work in Section 3.2.

3 Splitting Single Views to Two Views

In the previous section we describe an empirical approach to verify, when given the whole dataset and two views, if the two views satisfy the sufficiency and independence assumptions for co-training to work well. However, the standard two-view co-training has limited success in most real-world datasets with single views, such as most UCI datasets [11]. (One could also apply directly the single-view co-training on the datasets with single views, but other complications may be entailed.)

In this section we propose a simple heuristic to split single views into two views such that if the two views satisfy the sufficiency and independence assumptions, the two-view co-training is guaranteed to work well. The heuristic works as follows. We first calculate the entropy of each attribute in the single view based on the whole dataset D, similar to the entropy calculation for all attributes when deciding which attribute should be chosen as the root of the decision tree [12]. Intuitively, the larger the entropy, the more predictive of the class that the attribute would be. In order to distribute high-entropy attributes evenly in the two views, we simply assign attributes with the first, third, and so on (the

odd number of), highest entropy to the first view. We then assign attributes with the second, fourth, and so on (the even number of), highest entropy to the second view. Our proposed method is closely related to [13], which also splits single-views into two views. However, it simply splits the attributes randomly into two views. Later in this section, we make a comparision between our entropy splitting approach and the random splitting approach. After the two views are formed, the two assumptions (sufficiency and independence) for co-training are verified using the approaches described in the previous section.

We choose 32 UCI datasets coming with the WEKA package [14] to see if we can split the single view into two views for co-training to work. The continuous attributes are discredited into 10 equal-width bins in order to utilize naive Bayes [15] for checking the sufficiency and independence assumptions. As most previous co-training researches are based on binary classification problem, datasets with multiple classes are converted to binary by using the majority class value as one class, and the rest of the other values as the other class. These datasets are named with "_new" appended on the end of their original names.

In order to study the range of the sufficiency and independence thresholds for co-training to work well, we set $\delta_1 = 0.5$ for now (a very relaxed value, as any weak binary classifier should predict better than 50%). We apply both entropy splitting and random splitting on these 32 datasets for comparison. Entropy splitting yields smaller δ_1 on most datasets (31 out of 32) and smaller δ_2 on about half datasets (15 out of 32) compared to random splitting, thus we utilize it to verify the working of co-training in the rest of the paper. The cross-validated accuracies on the single view, the X view, and the Y view using naive Bayes on the whole datasets are listed in Table 1. We use "Acc(X,Y)", "Acc(X)", and "Acc(Y)" to denote them respectively in the table.

Our experiments of applying co-training on these UCI datasets are conducted in the following two high-level steps. In the first step, we run the standard co-training on these datasets to see if co-training would work. For each dataset we also obtain the tightest (smallest) sufficiency and independence thresholds that would make it pass the verification. In the second step, we apply a meta-learning algorithm [16] on the results of the first step to discover proper ranges of the thresholds that can predict when co-training works well. We describe these two steps in details below.

3.1 Applying Co-training on UCI Datasets

To apply co-training on these 32 datasets, the whole datasets D are first split randomly into three disjoint subsets: the training set (R), unlabeled set (U), and test set (T). The test set T is always 25% of D. To make sure that co-training can possibly show improvement when the unlabeled data are added, we choose a small training set for each dataset such that the "optimal gain" in accuracy when using the unlabeled data optimally is large enough (greater than 10%). The "optimal gain", denoted as "OptGain" in Table 1, is thus the difference between the accuracy on the initial training set R plus all unlabeled data with correct labels and the accuracy on R alone (without any benefit of unlabeled

examples). The "optimal gain" reflects the upper bound that co-training can achieve in accuracy. The unlabeled set is the whole dataset taking away the test set and the training set. The proper training set size (with the optimal gain greater than 10%) is also listed in Table 1. The standard co-training [3] is then applied. The process is repeated 20 times with different split of R, U, and T.

The average accuracy before applying co-training (test accuracy of applying naive Bayes on the initial training set; denoted as "IniAcc"), and the average accuracy after applying co-training (denoted as "CtAcc") are recorded in the table. A significance test, a paired t-test with 95% confidence, is applied to see if the test accuracy after co-training is significantly better than the test accuracy before co-training (i.e., if co-training really works or not). If it is, then co-training wins, denoted by W in the "CtWorks?" column; if it is significantly worse, then co-training loses (L); else co-training ties (T) with no co-training. These results are presented together in Table 1 for easy viewing.

From Table 1, we can see that overall, co-training wins in 6 datasets, loses in 3, and ties in the rest 23 datasets. Of course this does not imply that co-training does not work well for most single-view real-world datasets, as the sufficiency and independence thresholds (δ_1 and δ_2) are set very relaxed ($\delta_1 = 0.5$), thus the two views of these datasets may not be sufficient or independent. For each dataset, we can obtain the tightest (smallest) threshold values for the sufficiency and independence assumptions to pass. These threshold values (δ_1 and δ_2) are also listed in Table 1. These values provide us with an opportunity to discover the hidden regularity of these thresholds that make co-training win.

3.2 Meta-learning Co-training Thresholds

Results in Table 1 do seem to indicate that co-training would win when δ_1 and δ_2 are relatively small. In order to obtain a more precise range of δ_1 and δ_2, we use the idea of meta-learning to find hidden regularity of δ_1 and δ_2 that makes co-training work (win). We simply take, from Table 1, the numerical values in columns δ_1 and δ_2 as attributes, and W, L or T from "CtWorks?" as the class label. We obtain 32 training examples on which we can apply meta-learning.

We first assign W (win) as one class, and group L (lose) and T (tie) as the "others" class, to discover when co-training would win (W). As we expect simple rules for the thresholds, we apply WEKA's j48, the standard decision-tree algorithm [12] on the 32 training examples with pruning. The decision tree found is surprisingly simple:

```
d1 <= 0.23
|   d2 <= 0.15: W (7.0/2.0)
|   d2 > 0.15: others (8.0/1.0)
d1 > 0.23: others (17.0)
```

The tree discovered by j48 clearly indicates that co-training would win if the sufficiency threshold (δ_1) is less than or equals to 0.23, and the independence threshold (δ_2) is less than or equals to 0.15. There are 3 exceptions to the simple

Table 1. Applying the standard co-training on UCI datasets after view split. The cross-validated accuracies on the single view, the X view, and the Y view using naive Bayes on the whole datasets are listed as "Acc(X,Y)", "Acc(X)", and "Acc(Y)" respectively. "Training" indicates the size of training set, the test set is always 25% of the whole dataset, and the rest is the unlabeled set. "OptGain" is the difference between the accuracy on the initial training set plus all unlabeled data with correct labels and the accuracy on initial training set alone (without any benefit of unlabeled examples). "IniAcc" and "CtAcc" indicate the average test accuracy before and after applying co-training respectively. "CtWorks?" shows if co-training wins, ties or loses based on a paired t-test with 95% confidence.

Dataset	Acc(X,Y)	Acc(X)	Acc(Y)	δ_1	δ_2	Training	OptGain	IniAcc	CtAcc	CtWorks?
breast-cancer	75.5%	73.3%	74.0%	0.27	0.32	1/50	28.5%	0.44	0.48	T
breast-w	97.3%	96.6%	95.4%	0.05	0.13	1/100	17.2%	0.80	0.90	W
colic	78.8%	75.0%	81.8%	0.25	0.13	1/50	21.0%	0.60	0.63	T
credit-a	84.8%	84.6%	74.1%	0.26	0.31	1/50	23.0%	0.62	0.60	T
credit-g	76.3%	72.0%	73.4%	0.28	0.18	1/100	16.2%	0.58	0.58	T
diabetes	75.4%	69.7%	74.9%	0.30	0.12	1/50	15.1%	0.62	0.64	T
heart-statlog	83.7%	79.6%	76.7%	0.23	0.06	1/50	26.6%	0.59	0.73	W
hepatitis	83.9%	83.2%	82.6%	0.17	0.19	1/50	17.6%	0.69	0.70	T
ionosphere	90.9%	89.7%	90.1%	0.10	0.35	1/50	25.2%	0.66	0.69	T
kr-vs-kp	87.9%	70.4%	80.1%	0.30	0.27	1/200	24.0%	0.64	0.50	L
mushroom	95.8%	92.9%	98.5%	0.07	0.54	1/500	15.9%	0.80	0.82	T
sonar	77.4%	76.4%	76.0%	0.24	0.23	1/50	24.1%	0.53	0.52	T
tic-tac-toe	69.6%	70.5%	65.3%	0.35	0.17	1/100	15.5%	0.58	0.56	T
vote	90.1%	86.9%	91.0%	0.13	0.26	1/50	23.0%	0.68	0.60	T
anneal_new	92.4%	82.9%	92.5%	0.17	0.49	1/50	13.9%	0.77	0.71	L
arrhythmia_new	75.2%	74.6%	72.8%	0.27	0.34	1/50	21.9%	0.52	0.56	T
autos_new	79.0%	81.0%	72.2%	0.28	0.62	1/50	24.9%	0.57	0.55	T
cmc_new	65.4%	66.1%	61.0%	0.39	0.21	1/100	14.5%	0.52	0.50	T
cylinder-bands_new	73.9%	67.4%	73.8%	0.33	0.51	1/50	16.8%	0.57	0.51	L
dermatology_new	99.7%	99.5%	99.7%	0.01	0.20	1/50	27.1%	0.73	0.86	W
ecoli_new	97.7%	85.4%	83.9%	0.16	0.11	1/50	40.5%	0.57	0.70	W
flags_new	75.3%	73.7%	74.2%	0.26	0.32	1/50	20.0%	0.54	0.52	T
glass_new	72.4%	69.2%	68.2%	0.32	0.24	1/50	21.5%	0.53	0.55	T
haberman_new	75.8%	75.2%	74.2%	0.26	0.02	1/50	29.0%	0.46	0.49	T
heart-c_new	84.2%	80.2%	76.9%	0.23	0.10	1/50	20.0%	0.65	0.69	T
heart-h_new	84.0%	78.6%	80.3%	0.21	0.12	1/50	20.2%	0.66	0.75	W
liver-disorders_new	64.9%	60.0%	64.9%	0.40	0.03	1/50	18.0%	0.47	0.49	T
primary-tumor_new	85.0%	81.7%	79.1%	0.21	0.12	1/50	17.4%	0.67	0.66	T
solar-flare_1_new	74.0%	73.4%	74.9%	0.27	0.27	1/50	27.5%	0.47	0.50	T
solar-flare_2_new	100%	100%	81.4%	0.19	0.18	1/50	24.7%	0.75	0.69	T
spambase_new	84.5%	79.9%	78.6%	0.21	0.15	1/300	28.8%	0.57	0.75	W
splice_new	92.3%	84.1%	86.3%	0.16	0.19	1/200	25.9%	0.66	0.67	T

tree, but the overall accuracy of this tree is quite high at 91%, much higher than the default accuracy of 81% (26/32).

It would also be interesting to find out when co-training would lose (L) so it should be avoided. We use L (lose) as one class, and group win (W) and tie (T) as the "others" class, and run j48 again.[2] The result is also surprisingly simple:

[2] We tried to run j48 on the training examples with three classes: W, L and T, but it always returns a one-node tree, with or without pruning.

```
d2 <= 0.26: others (21.0)
d2 > 0.26
|   d1 <= 0.28: others (9.0/1.0)
|   d1 > 0.28: L (2.0)
```

This indicates that co-training would lose (so it should not be used) if the sufficiency threshold (δ_1) is greater than 0.28, and the independence threshold (δ_2) is greater than 0.26. Clearly our empirical results coincide well with theoretical findings that if the two views are sufficient and independent, co-training must work well (win). However, theoretical guarantee on sufficiency and independence is often impossible to obtain. What is more important is that the actual range of the sufficiency and independence thresholds, though discovered empirically here, provides a simple guideline for deciding and applying the standard co-training in real-world datasets.

4 Conclusions

To summarize, in this paper we propose empirical verification of the sufficiency and independence assumptions of the standard two-view co-training algorithm. We design heuristic to split datasets with a single view into two views, and if the two views pass the sufficiency and independence verification discovered by meta-learning, co-training is highly likely to work well. Our conclusions coincide well with the previous theoretical results, but our work provides a practical guide as to when co-training can work in datasets with two views. Our current work is based on the whole dataset. In our future work, we will study co-training verification on small training data.

Acknowledgments

Z.-H. Zhou was supported by NSFC (60635030, 60721002), JiangsuSF (BK2008018) and 863 Program (2007AA01Z169).

References

1. Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-Supervised Learning. MIT Press, Cambridge (2006)
2. Zhu, X.: Semi-supervised learning literature survey. Technical Report 1530, Department of Computer Sciences, University of Wisconsin at Madison, Madison, WI (2006)
3. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the 11th Annual Conference on Computational Learning Theory, Madison, WI, pp. 92–100 (1998)
4. Goldman, S., Zhou, Y.: Enhancing supervised learning with unlabeled data. In: Proceedings of the 17th International Conference on Machine Learning, San Francisco, CA, pp. 327–334 (2000)

5. Zhou, Z.H., Li, M.: Tri-training: exploiting unlabeled data using three classifiers. IEEE Transactions on Knowledge and Data Engineering 17, 1529–1541 (2005)
6. Abney, S.: Bootstrapping. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, pp. 360–367 (2002)
7. Balcan, M.F., Blum, A., Yang, K.: Co-training and expansion: Towards bridging theory and practice. In: Saul, L.K., Weiss, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems 17, pp. 89–96. MIT Press, Cambridge (2005)
8. Sarkar, A.: Applying co-training methods to statistical parsing. In: Proceedings of the 2nd Annual Meeting of the North American Chapter of the Association for Computational Linguistics, Pittsburgh, PA, pp. 95–102 (2001)
9. Pierce, D., Cardie, C.: Limitations of co-training for natural language learning from large data sets. In: Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing, Pittsburgh, PA, pp. 1–9 (2001)
10. Zhou, Z.H., Chen, K.J., Jiang, Y.: Exploiting unlabeled data in content-based image retrieval. In: Proceedings of the 15th European Conference on Machine Learning, Pisa, Italy, pp. 525–536 (2004)
11. Blake, C., Keogh, E., Merz, C.J.: UCI repository of machine learning databases. Department of Information and Computer Science, University of California, Irvine, CA (1998), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
12. Quinlan, R.J.: C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco (1993)
13. Nigam, K., Ghani, R.: Analyzing the effectiveness and applicability of co-training. In: Proceedings of the 9th ACM International Conference on Information and Knowledge Management, Washington, DC, pp. 86–93 (2000)
14. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, San Francisco (2005)
15. Langley, P., Iba, W., Thompson, K.: An analysis of bayesian classifiers. In: National Conference on Artificial Intelligence, pp. 223–228 (1992)
16. Vilalta, R., Drissi, Y.: A perspective view and survey of meta-learning. Artificial Intelligence Review 18(2), 77–95 (2002)

Classification of Audio Signals Using a Bhattacharyya Kernel-Based Centroid Neural Network

Dong-Chul Park[1], Yunsik Lee[2], and Dong-Min Woo[1]

[1] Dept. of Information Engineering, Myong Ji University, Korea
{parkd,dmwoo}@mju.ac.kr
[2] Korea Electronics Technology Institute, Seongnam, Korea
leeys@keti.re.kr

Abstract. A novel approach for the classification of audio signals using a Bhattacharyya Kernel-based Centroid Neural Network (BK-CNN) is proposed and presented in this paper. The proposed classifier is based on Centroid Neural Network (CNN) and also exploits advantages of the kernel method for mapping input data into a higher dimensional feature space. Furthermore, since the feature vectors of audio signals are modelled by Gaussian Probability Density Function (GPDF), the classification procedure is performed by considering Bhattacharyya distance as the distance measure of the proposed classifier. Experiments and results on various audio data sets demonstrate that the proposed classification scheme based on BK-CNN outperforms conventional algorithms including Self-Organizing Map(SOM) and CNN.

1 Introduction

With the large number of audio signals such as speech and music data that can be retrieved through internet services, automatic classification and retrieval methods have received much attention recently from researchers in this field. In a content-based retrieval system for audio signals, audio signal classification normally involves the processing of two main tasks: audio feature extraction and a classification task that uses the extracted features to discriminate the classes. Audio data descriptions used for the similarity measure of audio signals in feature extraction method include Mel-Frequency Cepstrum Coefficients (MFCC), the timbre, the rhythm, pitch-related features, FFT coefficients, Linear Prediction (LP) coefficients, MPEG filterbank components, the spectral centroid, the spectral flux, the zero crossing rate, the spectral roll-off, low order statistics, and delta coefficients [1,2]. Among these features, MFCC, which was originally developed for speech recognition systems[3,4], has been widely used in various audio information retrieval tasks of late. Recent studies indicate that classification schemes with MFCC features outperform other existing features at a similar time level [5,6].

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 604–611, 2009.
© Springer-Verlag Berlin Heidelberg 2009

After the features are determined and extracted from audio data sets, a classifier is employed to classify the genre of a given audio data. In recent years, different clustering methods have been proposed for automatic discrimination of speech and music signals. Among these, the Gaussian Mixture Model (GMM) provides notably enhanced performance over conventional models while having a major impact on the computational complexity[1,7]. A GMM is obtained from training data by considering audio signals as mixtures of Gaussian Probability Density Functions (GPDFs). The GPDF model contains a mean vector and a covariance matrix. From the trained GMM, the classification decision is based on the likelihoods of the audio feature vectors and given statistical models.

Among various clustering algorithms, the k-means algorithm[8] is one of the most widely utilized unsupervised learning algorithms. The strengths of the k-means algorithm include its convergence speed and its applicability to large data. Meanwhile, the Self-Organizing Map (SOM)[9], another conventional algorithm, finds the winner neuron, which is the closest to a given datum, and updates the synaptic weights of the winner and its neighbors. The clustering quality of SOM heavily depends on the initial weight values, the sequence of data, the learning coefficient and the number of total iterations [10]. Pedro et al. successfully used SOM for a music genre classification problem[11]. Grimaldi et al. [12] used a set of features based on a discrete wavelet packet transform (DWPT) to represent a music track. The classification performance can be evaluated by four alternative classifiers: simple k-nearest neighbor, one-against-all, Round-Robin, and feature subspace based ensembles of nearest neighbor classifiers. One of the disadvantages of these algorithms is an inability to deal with data in which boundaries are non-linear, in which case the mixtures of the GMM are inefficient.

In order to improve the stability of clustering results from the k-means algorithm and SOM, the Centroid Neural Network (CNN) algorithm introduces the concept of reward and punishment to the winner and the loser neurons [10]. Furthermore, in order to obtain a closer solution to the best clustering result, CNN starts by setting the number of groups at 2 and increases the number of groups one by one until reaching the predetermined number of groups. In most experiments, CNN has shown superior performance to conventional algorithms including SOM and Differential Competitive Learning [13].

The approach presented in this paper is based on CNN and the Bhattacharyya kernel[14,15] for clustering GPDF data of a GMM. Since a Gaussian distribution captures only a limited part of the statistics in the empirical distribution of the vectors, those vectors are first mapped in a feature space via the minor kernel. The Bhattacharyya affinity is computed in the feature space, thereby making it possible to capture more information of the empirical distribution [16].

The remainder of this paper is organized as follows: Section 2 briefly summarizes the CNN. MFCC, the extracted feature vector, is presented in Section 3. A summary of the Bhattacharyya distance and the Bhattacharyya kernel is given in Section 4 and BK-CNN is outlined in Section 5. Section 6 provides the experiments involving practical audio data set and presents the results. Finally, conclusions are given in Section 7.

2 Centroid Neural Network(CNN)

The CNN algorithm [10] is an unsupervised competitive learning algorithm based on the classical k-means clustering algorithm [8]. It finds the centroids of clusters at each presentation of the data vector. The CNN first introduces definitions of the winner neuron and the loser neuron. When a data $\mathbf{x}_i$ is given to the network at the epoch (k), the winner neuron at the epoch (k) is the neuron with the minimum distance to $\mathbf{x}_i$. The loser neuron at the epoch (k) to $\mathbf{x}_i$ is the neuron that was the winner of $\mathbf{x}_i$ at the epoch (k-1) but is not the winner of $\mathbf{x}_i$ at the epoch (k). The CNN updates its weights only when the status of the output neuron for the presenting data has changed when compared to the status from the previous epoch.

When an input vector $\mathbf{x}$ is presented to the network at epoch n, the weight update equations for winner neuron j and loser neuron i in CNN can be summarized as follows:

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \frac{1}{N_j + 1}[\mathbf{x}(n) - \mathbf{w}_j(n)] \tag{1}$$

$$\mathbf{w}_i(n+1) = \mathbf{w}_i(n) - \frac{1}{N_i - 1}[\mathbf{x}(n) - \mathbf{w}_i(n)] \tag{2}$$

where $\mathbf{w}_j(n)$and $\mathbf{w}_i(n)$ represent the weight vectors of the winner neuron and the loser neuron, iteration, respectively.

The CNN has several advantages over conventional algorithms such as SOM or k-means algorithm when used for clustering and unsupervised competitive learning. The CNN requires neither a predetermined schedule for learning gain nor the total number of iterations for clustering. It always converges to sub-optimal solutions while conventional algorithms such as SOM may give unstable results depending on the initial learning gains and the total number of iterations. More detailed description on the CNN can be found in [10,17].

3 Feature Extraction

In this work, the problem of automatic classification of audio signals is viewed as a pattern classification problem where a music sample, considered as an audio signal, is represented in terms of feature vectors. The aim of feature extraction is to represent audio data in a compact and descriptive manner such that it is efficient to deal with when applying learning algorithms. MFCC has been widely used for speech recognition due to its ability to capture the perceptually most important parts of a spectral envelope of audio signals[3]. It has been proven that MFCC is an effective tool in automatic speech recognition and in modelling the subjective frequency content of audio signals. The input signal is divided into the number of frames and the MFCC is computed at each frame. In this paper, in order to represent the characteristics of the audio signal, the mean and deviation of the MFCC are adopted as the feature vectors.

4 Clustering using Bhattacharyya Distance and Bhattacharyya Kernel

4.1 Bhattacharyya Distance

The Bhattacharyya distance has been widely used in the statistical pattern recognition[16]. This measure of divergence,$D\left(G_i, G_j\right)$ is a measure between two Gaussian distributions and it defined as follows:

$$D\left(G_i, G_j\right) = \frac{1}{8}\left(\mu_i - \mu_j\right)^T \left[\frac{\Sigma_i + \Sigma_j}{2}\right]^{-1} \left(\mu_i - \mu_j\right) + \frac{1}{2}\ln \frac{\left|\frac{\Sigma_i + \Sigma_j}{2}\right|}{\sqrt{\left|\Sigma_i\right|\left|\Sigma_j\right|}} \tag{3}$$

where μ_i and $\sum_i$ denote the mean vector and covariance matrix of Gaussian distribution G_i, respectively. T denotes the transpose matrix.

4.2 Bhattacharyya Kernel

Kernel-based methods are well-established tools that are useful in a variety of contexts including classification, regression, density estimation, and clustering problems. They are known to represent complex decision boundaries very efficiently and generalize well to unseen data during the training stage. The Bhattacharyya kernel for GPDF data is defined as follows[14]:

$$BK\left(\mathbf{x}\left(n\right), \mathbf{w}_j\left(n\right)\right) = \exp\left(-\alpha D\left(\mathbf{x}\left(n\right), \mathbf{w}_j\left(n\right)\right) + b\right) \tag{4}$$

where $BK\left(\mathbf{x}\left(n\right), \mathbf{w}_j\left(n\right)\right)$ is the Bhattacharyya with a kernel distance between two Gaussian distributions, $\mathbf{x}$(n) and $\mathbf{w}_j(n)$.

Each set of vectors was suggested to fit a Gaussian distribution. The kernel value between the two sets of vectors is then defined as the Bhattacharyya affinity between the two corresponding Gaussian distributions, which can be computed in a closed form.

5 CNN with the Bhattacharyya Kernel (BK-CNN)

The energy function with a kernel can be written in the feature space with the mapping function ϕ:

$$E_i^{\phi} = \sum_{j=1}^{N_i} \left\|\phi\left(\mathbf{x}_i\left(j\right)\right) - \phi\left(\mathbf{w}_i\right)\right\|^2 \tag{5}$$

$\mathbf{x}_i\left(j\right)$ denotes the data j in the cluster i.

In order to find the minimum of the objective function with a kernel, the steepest gradient descent algorithm can be adopted and the update equations for BK-CNN can be derived. As is the case of CNN, the BK-CNN updates its weights only when the status of the output neuron for the presented data has changed when compared to the status from the previous epoch. When an input

vector $\mathbf{x}$ is presented to the network at epoch n, the weight update equations for the winner neuron j and the loser neuron i in the BK-CNN can be summarized as follows:

$$l\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \frac{\alpha BK(\mathbf{x}(n), \mathbf{w}_j(n))}{(N_j+1) * \left(\sum_{\mathbf{x}(n)} + \sum_{\mathbf{w}_j(n)}\right)} [\mathbf{x}(n) - \mathbf{w}_j(n)] \quad (6)$$

$$l\mathbf{w}_i(n+1) = \mathbf{w}_i(n) - \frac{\alpha BK(\mathbf{x}(n), \mathbf{w}_i(n))}{(N_i-1) * \left(\sum_{\mathbf{x}(n)} + \sum_{\mathbf{w}_i(n)}\right)} [\mathbf{x}(n) - \mathbf{w}_i(n)] \quad (7)$$

where $\mathbf{w}_j(n)$and $\mathbf{w}_i(n)$ represent the weight vectors of the winner neuron and the loser neuron, iteration, respectively.

More detailed information on CNN with the Bhattacharyya Kernel can be found in [18].

6 Experiments and Results

For the evaluation of the proposed audio signal classifier based on BK-CNN, two data sets, a speech data set and a music data set, consisting of 1,000 audio signals

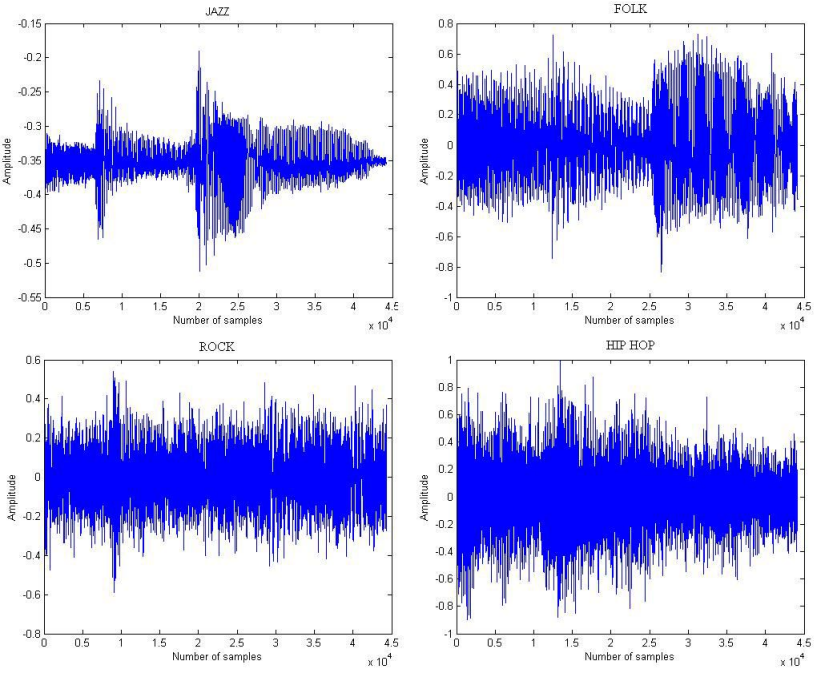


Fig. 1. Amplitudes of 4 kinds of music signals: Jazz, Folk, Rock, Hip hop of 44112 samples (Sampling rate = 44122 Hz)

Table 1. The comparison of classification accuracy (%) of speech/music classifier using different algorithm (SOM,CNN,BK-CNN) and 4 code vectors

	Speech	Music	Overall
SOM	99.92	85.6	92.76
CNN	99.94	87.2	93.57
BK-CNN	99.95	99.98	99.97

Table 2. The comparison of classification accuracy (%) of music genre classifier using different algorithm (SOM,CNN,BK-CNN) and 5 code vectors

	Jazz	Folk	Rock	Hip hop	Overall
SOM	89.2	67.8	63.3	62.7	**70.5**
CNN	89.4	83.6	72.9	48.5	**73.6**
BK-CNN	94.2	93.8	73.6	88.4	**87.5**

are collected and used for experiments. The speech data set, which contains 200 signals of 30 seconds length each, is collected from BBC Radio News. The speech class includes both male and female speakers, recorded both in studio and telephonic environments with a variable amount of background noise. The sampling rate was 44.1k Hz, quantized with 16 bits and only one channel (mono). The second data set consisting of 800 signals of 4 music genres (jazz, folk, rock, and hip hop), as shown in Fig.(1), was downloaded from an internet site [19]. In order to extract multi-dimensional MFCC feature vectors for audio signals, we use jAudio[20], an application software designed for a variety of music information retrieval tasks.

First, the speech/music classification task was performed. In this case, 200 speech data and 200 randomly selected music data from various music genres were employed. Each of the two data sets is partitioned into two subsets: 150 randomly selected samples are used for the training data set and the remaining 50 samples are used for the test data set. Experiments are performed with 50 different combinations of training and test data sets. The classification accuracies shown in Table 1 are the means of the obtained accuracies from the 50 different combinations of the training and test data sets. As can be seen from Table 1 for the speech/music classification task, all three algorithms almost perfectly distinguish speech signals from music signals. However, only the BK-CNN successfully discriminates music signals from speech signals. This implies that the Bhattacharyya kernel for GPDF data is a critical addition to CNN for the speech/music classification task.

Another experiment related to the music genre classification problem was performed. Each of 4 data sets is again partitioned into two subsets: 150 randomly selected samples for the training data set and the remaining 50 samples for the test data set. Experiments are performed with 50 different combinations of training and test data sets. The classification accuracies reported in Table 2

and Table 3 are the means of the obtained accuracies from the 50 different cases of the training and test data sets. The SOM, CNN, and BK-CNN are evaluated through the experiments. As shown in Table 2, overall accuracies of 70.5%, 73.6%, and 87.5% are obtained using the SOM, CNN and BK-CNN, respectively. When compared to the speech/music classification task, the music genre classification problem is considerably more complicated. However, the BK-CNN still outperforms the CNN by 14% in terms of classification accuracy and the usefulness of the Bhattacharyya kernel in the CNN is again demonstrated by this experiment.

7 Conclusion

In this paper, a novel approach for the classification of audio signals using the Bhattacharyya Kernel-based Centroid Neural Network (BK-CNN) is proposed. Experiments were performed for two data sets, a speech data set and a music data set, consisting of 1,000 audio signals. Experiments on the speech/music classification task were first performed. The results show that all three algorithms (SOM, CNN, BK-CNN) almost perfectly identify speech signals from music signals. However, only the BK-CNN successfully discriminates music signals from speech signals. In the subsequent experiments, the music genre classification problem was considered. Four music genres, jazz, folk, rock, and hip hop, were trained and tested using SOM, CNN, and BK-CNN. Results shows that the BK-CNN outperforms the SOM and CNN by 17 % and 14%, respectively, in terms of classification accuracy. The experiments and results clearly show that integration of the CNN and the Bhattacharyya kernel provides an efficient tool for the audio signal classification problem.

Acknowledgement

This work was supported by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korean government (MOST)(Grant No.: R01-2007-000-20330-0).

References

1. Tzanetakis, G., Cook, P.: Music Genre Classification of Audio Signals. IEEE Trans. Speech and Audio Processing 10(5), 293–302 (2002)
2. Aucouturier, J.J., Pachet, F.: Representing Musical Genre:A State of the Art. Journal of New Music Research 32(1), 83–93 (2003)
3. Davis, S.B., Mermelstein, P.: Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Trans. on Acoustics, Speech and Signal Processing 28, 357–366 (1980)
4. Jankowski, C.R., Vo, H.D., Lippmann, R.P.: A comparison of signal processing front ends for automatic word recognition. IEEE Trans. on Speech and Audio Processing 3(4), 286–293 (1995)

5. Gook, H.K., Sikora, T.: Audio spectrum projection based on several basis decomposition algorithms applied to general sound recognition and audio segmentation. In: Proc. of EUSIPCO, pp. 1047–1050 (2004)
6. Ahrendt, P., Meng, A., Larsen, J.: Decision time horizon for music genre classification using short-time features. In: Proc. of EUSIPCO, pp. 1293–1296 (2004)
7. Lu, L., Zhang, H.J., Jiang, H.: Content Analysis for Audio Classification and Segmentation. IEEE Trans. on Speech and Audio Processing 10(7), 504–516 (2002)
8. Darken, C., Moody, J.: Fast adaptive k-means clustering: Some empirical resultes. Proceedings of Int. Jont Conf. Neural Networks 2, 238–242 (1990)
9. Kohonen, T.: Self-Organization Map. Proc. IEEE 78, 1464–1480 (1990)
10. Park, D.C.: Centroid Neural Network for Unsupervised Competitive Learning. IEEE Trans. on Neural Networks 11(2), 520–528 (2000)
11. Leon, P.J.P., Inesta, J.M.: Feature-driven recognition of music style. In: Perales, F.J., Campilho, A.C., Pérez, N., Sanfeliu, A. (eds.) IbPRIA 2003. LNCS, vol. 2652, pp. 773–781. Springer, Heidelberg (2003)
12. Grimaldi, M., Cunningham, P., Kokaram, A.: An evaluation of alternative feature selection strategies and ensemble techniques for classifying music. In: ECML (2003)
13. Kong, S., Kosko, B.: Differential competitive learning for centroid estimation and phoneme recognition. IEEE Trans. Neural Networks 2, 118–124 (1991)
14. Kondor, R., Jebara, T.: A kernel between sets of vectors. In: Proceedings of the ICML (2003)
15. Park, D.-C., Kwon, O.H., Chung, J.: Centroid Neural Network With a Divergence Measure for GPDF Data Clustering. IEEE Trans on Neural Networks 19, 948–957 (2008)
16. Bhattacharyya, A.: A measure of divergence between two statistical populations defined by their probability distributions. Bull. Calcutta Math Soc. 35, 99–110 (1943)
17. Park, D.C., Woo, Y.: Weighted Centroid Neural Network for Edge Reserving Image Compression. IEEE Trans. on Neural Networks 12(5), 1134–1146 (2001)
18. Lee, S.-J., Park, D.-C.: Centroid Neural Network with Bhattacharyya Kernel for GPDF Data Clustering. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 616–622. Springer, Heidelberg (2007)
19. `http://music.download.com/`
20. McEnnis, D., McKay, C., Fujinaga, I., Depalle, P.: jAudio: A feature extraction library. In: Proceedings of the International Conference on Music Information Retrieval, pp. 600–603 (2005)

Sparse Kernel Learning and the Relevance Units Machine

Junbin Gao[1,⋆] and Jun Zhang[1,2]

[1] School of Accounting and Computer Science,
Charles Sturt University, Bathurst, NSW 2795, Australia
{jbgao,jzhang}@csu.edu.au
[2] Institute for Pattern Recognition and Artificial Intelligence,
Huazhong University of Science and Technology,
Wuhan, Hubei 430074, P.R. China
white70121@163.com

Abstract. The relevance vector machine(RVM) is a state-of-the-art constructing sparse regression kernel model [1,2,3,4]. It not only generates a much sparser model but provides better generalization performance than the standard support vector machine (SVM). In RVM and SVM, relevance vectors (RVs) and support vectors (SVs) are both selected from the input vector set. This may limit model flexibility. In this paper we propose a new sparse kernel model called Relevance Units Machine (RUM). RUM follows the idea of RVM under the Bayesian framework but releases the constraint that RVs have to be selected from the input vectors. RUM treats relevance units as part of the parameters of the model. As a result, a RUM maintains all the advantages of RVM and offers superior sparsity. The new algorithm is demonstrated to possess considerable computational advantages over well-known the state-of-the-art algorithms.

1 Introduction

Statistical Learning plays a key role in many areas of science, finance and industry. Kernel methods [5] have demonstrated great successes in solving many machine learning and pattern recognition problems. Many kernel methods produce a model function dependent only on a subset of kernel basis functions associated with some of the training samples. Those samples are called the support vectors (SVs) in support vector machine (SVM) methods[6]. SVM has an excellent generalization properties with a sparse model representation [7]. The SVM and kernel machine models (KMM) [5,8] have attracted considerable interests. Generally speaking, an SVM method often learns a parsimonious model that ensure the simplest possible model that explains the data well. Apart from the obvious computational advantage, practices have demonstrated that simple models often generalize better for the unseen data. Additionally, learning a sparse model has deep connections with problems of selecting regressors in regression [9,10,11,12].

⋆ The author to whom all the correspondences should be addressed.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 612–619, 2009.
© Springer-Verlag Berlin Heidelberg 2009

However, it has been shown that the standard SVM technique is not always able to construct parsimonious models, for example, in system identification [13]. This inadequacy motivates the exploration of new methods for parsimonious models under the framework of both SVM and KMM. Tipping [3] first introduced the relevance vector machine (RVM) method which can be viewed from a Bayesian learning framework of kernel machine and produces an identical functional form to SVM/KMM. The results given by Tipping [3] have demonstrated that the RVM has a comparable generalization performance to the SVM but requires dramatically fewer kernel basis functions or model terms than the SVM. A drawback of the RVM algorithm is a significant increase in computational complexity, compared with the SVM method.

Recently, two separate works [14,15] considered applying the L1 penalty into model regularization. L1 penalty originates from the least absolute selection and shrinkage operator (LASSO) first introduced by Tibshirani [16]. In the classical LASSO, the L1 penalty is applied on the weights of each predictor in a linear model. An earlier attempt, called "generalized LASSO", can be found in [17].

One features shared by all the mentioned approaches is that the sparse model is obtained from a full kernel model defined on the whole dataset and the approach employs an algorithm procedure to trim off unnecessary kernel basis functions associated with some input vectors. The retained input vectors used in the resulting spare model are called such as SVs, relevance vectors (RVs) and critical vectors (CVs) etc. Obviously it is not necessary for these CVs to have to be chosen from the training input vectors. In this paper, we propose a new sparse kernel model called Relevance Units Machine (RUM) in which the CVs will be learnt from data. The idea is not new. It has been around for many years, for example, the direct method for sparse model learning [18], the reduced set (RS) method [5,19], and the sparse pseudo-input Gaussian processes [20]. However in our approach a Bayesian inference framework is adopted so that all the parameters including kernel parameters are to be learnt from the Bayesian inference.

In Section 2 of this paper, the concepts of the RUM are given and the algorithm associated with RUM is presented in Section 3. The experiment results are presented in section 4, followed by our conclusions in Section 5.

2 The RUM Model Description

In supervised learning we are given a set of examples of input vectors set $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\} \subset \mathbb{R}^q$, along with corresponding targets $\mathbf{t} = [t_1, t_2, ..., t_N]^T$ which are independent and identically distributed. Many kernel learning algorithms result in a kernel machine (KM) (such as a kernel classifier), whose output can be calculated as $y = y(\mathbf{x}; \boldsymbol{w}) = w_0 + \sum_{n=1}^{N} w_n k(\mathbf{x}, \mathbf{x}_n)$, where $k(\mathbf{x}, \mathbf{x}_n)$ is a kernel function, effectively defining one basis function for each example in the training set. The learning task becomes finding suitable w_n according to different criteria. Most criteria like those used by SVM and RVM lead to zero values for a large number of w_n so that a sparse model is established.

Inspired by the idea used in the direct method for sparse model learning [18], we consider a model defined as, $t = y(\mathbf{x}, \boldsymbol{w}) + \epsilon = w_0 + \sum_{m=1}^{M} w_m k(\mathbf{x}, \mathbf{u}_m) + \epsilon$,

where $\mathbf{u}_m \in \mathbb{R}^q, m = 1, 2, ..., M$ are unknown units for the model and ϵ is additive noise assumed to be a Gaussian of 0 mean and an unknown variance β^{-1}, denoted by $\mathcal{N}(\epsilon|0, \beta^{-1})$. M controls the sparsity of the model and we assume that M is known in the modeling process. In our experimental setting, we set the minimum of $M = 0.04N$ where N is the total number of training data.

Without loss of generality, we assume $w_0 = 0$ in the sequel. The learning task is to learn the weight vector $\boldsymbol{w} = [w_1, w_2, ..., w_M]^T$, the units matrix $\mathbf{U} = [\mathbf{u}_1, ..., \mathbf{u}_M]$, the noise precision β and hyperparameters in the kernel function k.

To develop an approach for learning all the parameters, we propose to apply Bayesian inference. Let $\mathbf{K_u} = [k_u^T(\mathbf{x}_1), k_u^T(\mathbf{x}_2), ..., k_u^T(\mathbf{x}_N)]^T, k_u(\mathbf{x}) = [k(\mathbf{x}, \mathbf{u}_1), k(\mathbf{x}, \mathbf{u}_2), ..., k(\mathbf{x}, \mathbf{u}_M)]$. Due to the assumption of independence of the data points, the likelihood of the complete training data can be written as

$$p(\mathbf{t}|X, \mathbf{U}, \boldsymbol{w}, \beta, \Theta) = \left(\frac{\beta}{2\pi}\right)^{N/2} \exp\left\{-\frac{\beta}{2}\|\mathbf{t} - \mathbf{K_u}\boldsymbol{w}\|^2\right\} \tag{1}$$

where Θ is the set of kernel hyperparameters.

To make a generative Bayesian model, we further specify a Gaussian prior over the weights as done in the RVM approach[3], that is,

$$p(\boldsymbol{w}|\boldsymbol{\alpha}) = \prod_{m=1}^{M} \mathcal{N}(w_m|0, \alpha_m^{-1}) \tag{2}$$

where $\boldsymbol{\alpha} = [\alpha_1, ..., \alpha_M]^T$ is a vector of M hyperparameters which control precision of the Gaussian over the weights. Similarly the prior over $\mathbf{U}$ is

$$p(\mathbf{U}|r) = \left(\frac{r}{2\pi}\right)^{Mq/2} \exp\left\{-\frac{r}{2}\mathrm{tr}\left[\mathbf{U}^T\mathbf{U}\right]\right\} \tag{3}$$

These hyperparameters $\boldsymbol{\alpha}$, r and the precision β of the Gaussian noise in the target are empowered with hyperpriors given by Gamma distributions

$$p(\boldsymbol{\alpha}) = \prod_{m=1}^{M} \mathrm{Gamma}(\alpha_m|a, b), p(\beta) = \mathrm{Gamma}(\beta|c, d), p(r) = \mathrm{Gamma}(r|e, f).$$

We fixed parameters $a = b = c = d = e = f = 10^{-4}$ in our setting.

Combining (1) - (3), we have,

$$p(\mathbf{t}, \mathbf{U}, \boldsymbol{w}, \boldsymbol{\alpha}, \beta|X, \Theta) = p(\mathbf{t}|X, \mathbf{U}, \boldsymbol{w}, \beta, \Theta)p(\mathbf{U}|r)p(r)p(\boldsymbol{w}|\boldsymbol{\alpha})p(\boldsymbol{\alpha})p(\beta) \tag{4}$$

Let $A = \mathrm{diag}(\alpha_1, \alpha_2, ..., \alpha_M)$, to integrate (4) over $\boldsymbol{w}$, we can see

$$p(\mathbf{t},\mathbf{U}, \boldsymbol{\alpha}, \beta|X, \Theta) = \frac{\exp\left\{-\frac{1}{2}\mathbf{t}^T(\beta^{-1}I_N + \mathbf{K_u}A^{-1}\mathbf{K_u^T})^{-1}\mathbf{t}\right\}}{(2\pi)^{N/2}|\beta^{-1}I_N + \mathbf{K_u}A^{-1}\mathbf{K_u^T}|^{1/2}} p(\mathbf{U}|r)p(r)p(\boldsymbol{\alpha})p(\beta) \tag{5}$$

To cope with the outliers in the targets $\mathbf{t}$, a L1 Laplacian noise model [21] can replace the Gaussian one, in which a robust RUM can be formulated with the variational Bayesian inference procedure. We leave this for another work.

3 Algorithm Design

As the probabilistic model for the relevance units machine only involves Gaussian and Gamma distributions, see (5), the inference can be easily established by using normal maximum likelihood (ML) procedure. The algorithm deduction is straightforward. Let $L = ln[p(\mathbf{t}, \mathbf{U}, \boldsymbol{\alpha}, \beta | X, \Theta)]$, $\Sigma = (A + \beta \mathbf{K}_\mathbf{u}^T \mathbf{K}_\mathbf{u})^{-1}$, $\boldsymbol{\mu} = \beta \Sigma \mathbf{K}_\mathbf{u}^T \mathbf{t}$. By a tedious calculation process we can obtain that

$$\frac{\partial L}{\partial \beta} = \frac{1}{2}\left[\frac{N}{\beta} - \mathrm{tr}(\Sigma \mathbf{K}_\mathbf{u}^T \mathbf{K}_\mathbf{u}) - \|\mathbf{t} - \mathbf{K}_\mathbf{u}\boldsymbol{\mu}\|^2\right] + \frac{c-1}{\beta} - d \tag{6}$$

$$\frac{\partial L}{\partial \alpha_m} = \frac{1}{2}\left[\frac{1}{\alpha_m} - (\Sigma_{mm} + \mu_m^2)\right] + \frac{a-1}{\alpha_m} - b. \tag{7}$$

where Σ_{mm} is the m-th diagonal element of Σ. Similarly we have

$$\frac{\partial L}{\partial r} = \frac{Mq}{2r} - \frac{1}{2}\mathrm{tr}(\mathbf{U}^T\mathbf{U}) + \frac{e-1}{r} - f. \tag{8}$$

$$\frac{\partial L}{\partial \mathbf{U}} = \frac{\partial L}{\partial \mathbf{K}_\mathbf{u}} \otimes \frac{\partial \mathbf{K}_\mathbf{u}}{\partial \mathbf{U}} - r\mathbf{U}, \frac{\partial L}{\partial \Theta} = \frac{\partial L}{\partial \mathbf{K}_\mathbf{u}} \otimes \frac{\partial \mathbf{K}_\mathbf{u}}{\partial \Theta}. \tag{9}$$

$$\frac{\partial L}{\partial \mathbf{K}_\mathbf{u}} = \beta[(\mathbf{t} - \mathbf{K}_\mathbf{u}\boldsymbol{\mu})\boldsymbol{\mu}^T - \mathbf{K}_\mathbf{u}\Sigma]$$

However, once the analytical expression of the kernel function is given, it is easy to find out both $\frac{\partial \mathbf{K}_\mathbf{u}}{\partial \mathbf{U}}$ and $\frac{\partial \mathbf{K}_\mathbf{u}}{\partial \Theta}$. With (6) - (9), a gradient type algorithm for maximizing L can be constructed.

Once all the parameters have been determined, to solve a normal regularized least squares problem, we have $\boldsymbol{w} = \beta \Sigma \mathbf{K}_\mathbf{u}^T \mathbf{t}$. For a new unseen input $\mathbf{x}^*$, let $K_{*u} = [k(\mathbf{x}^*, \mathbf{u}_1), k(\mathbf{x}^*, \mathbf{u}_2), ..., k(\mathbf{x}^*, \mathbf{u}_M)]^T$, the confidence (or error bar) on the prediction outcome $t^* = K_{*u}\boldsymbol{w}$ can be given by

$$\mathrm{var}(t^*) = \beta^{-1} + K_{*u}^T A^{-1} K_{*u} - K_{*u}^T A^{-1} \mathbf{K}_\mathbf{u} (\beta^{-1} I_N + \mathbf{K}_\mathbf{u} A^{-1} \mathbf{K}_\mathbf{u}^T)^{-1} \mathbf{K}_\mathbf{u}^\mathbf{T} A^{-1} K_{*u}$$

Note 1: The above algorithm can be easily extended to the case of multivariate outputs, i.e., t_n is a D-dimensional vector. In some application problems, the dimension D is much greater than the dimension d of the input vectors.

Note 2: The algorithm can be generalized to the unsupervised learning setting in which the input X is unknown. When $D \gg d$, to seek the projection images $\mathbf{x}_n \in \mathbb{R}^d$ of t_n is considered as a dimensionality reduction algorithm in terms of the latent variable model. The closest approach in literature is the so-called Gaussian Process latent variable model [22]. This is our next work.

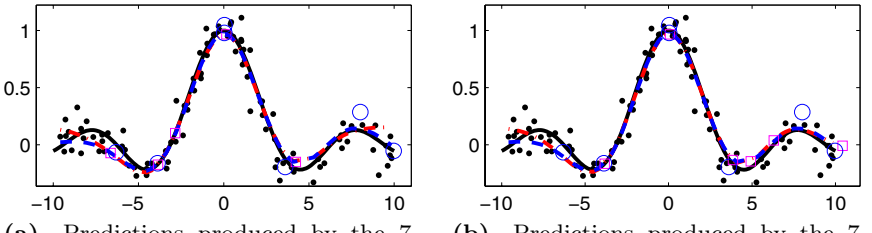
4 Experimentals

In this section, we will test the RUM method for regression problems on both synthetic and real world datasets. The experiments will show the ability of RUM in modeling dataset and learning kernel hyperparameters and also compare RUM to RVM [3]. The experiment of the RVM part was conducted by using Tipping's Matlab code (`http://www.miketipping.com/index.php?page=rvm`).

Example 1: In this example synthetic data were generated from the scalar function, $\mathrm{sinc}(x) = \sin(x)/x, -10 \leq x \leq 10$. The sets of training data $\{(x_n, t_n)\}_{n=1}^{100}$ and test data $\{(xt_i, tt_i)\}_{i=1}^{100}$ are generated for the input x_n and xt_i by drawing from the uniform distribution over $[-10, 10]$ and target Gaussian noise within t_n and tt_i were given with zero mean and variance 0.113. The targets is quite noisy compared to the maximal target values 1.

The RBF kernel function used in this experiment takes the following form, $k(x, x'; \sigma_w) = \exp\left\{-\frac{1}{2\sigma_w^2}(x - x')^2\right\}$, where σ_w is called the width of the RBF kernel. A full kernel model is defined for the RVM algorithm by all the RBF regressors with centers at each input training datum. In the experiment, the width of the RBF kernel function is set to 2.1213 for RVM algorithm where 9.0 was the best value for the kernel variance as chosen for this example in Tipping's Matlab code. For RUM algorithm, the width of the RBF kernel function is treated as a unknown parameter which is automatically estimated by the learning procedure. All the two algorithms produce sparse models. Their predictions on unseen data are presented in Figure 1.

Table 1 compares the mean square error (MSE) values over the training and test sets for the models constructed by the RUM and RVM. The number of iterative loops and the RBF kernel widths are also listed. The numbers of chosen regressors are, respectively, 7 (RVM), 5–8 (RUM).



(a). Predictions produced by the 7-term RVM and 5-term RUM

(b). Predictions produced by the 7-term RVM and 6-term RUM

Fig. 1. The predictions for the simple scalar sinc function modelling problem: dots are the noise testing data, the black solid curve is the underlying function $\mathrm{sinc}(x)$, the red dash-dot curve is prediction generated from RUM, the blue dash curve is prediction generated from RVM, the marker □ indicates the RUs selected by RUM, and the marker ∘ indicates the RVs selected by RVM

Table 1. Mean Square Errors of Example 1

Methods	RVM (7*)	RUM (5)	RUM (6)	RUM (7)	RUM (8)
Training MSE	0.00216	0.00457	0.00351	0.00161	0.00181
Test MSE	0.00228	0.00480	0.00262	0.00173	0.00222
IterativeLoops	1000	36	54	37	34
RBF Width	2.12130	1.59858	2.18004	2.91481	2.10048

*The number of regressors found by the algorithms.

In summary, the results given by the RUM are comparable to the result generated by the RVM algorithm however the performance of the RVM algorithm depends on the choice for the value of the RBF width. In experiment we also find that the RUM has less computational cost than the RVM algorithm.

Example 2: The second example is a practical modeling problem [8][1]. In this example, we are about to construct a model representing the relationship between the fuel rack position (input) and engine speed (output) for a Leyland TL11 turbocharged, direct inject diesel engine operated at low engine speed. A detailed system description and experimental setup can be found in [23]. The data set consists of 410 samples. We use the first 1/3 of the total data points, 140 data points as training data in modeling and all the points in model validation with one-step prediction and iterative prediction.

The input vector $\mathbf{x}_n = [t_{n-1}, v_{n-1}, v_{n-2}]^T$, where t_{n-1} is the engine speed, also considered as outputs in modeling, and v means the fuel input at the last time step. That is, the output t_n at time n depends on the output t_{n-1} at time $n-1$. Actually this is a time series modeling problem. The kernel to be used is an RBF kernel defined on three dimensional vectors,

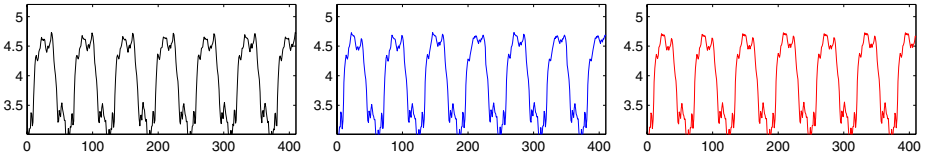
$$k(\mathbf{x}_i, \mathbf{x}_j; \sigma_w) = \exp\left\{-\frac{1}{2\sigma_w^2}\left[(t_{i-1} - t_{j-1})^2 + (v_{i-1} - v_{j-1})^2 + (v_{i-2} - v_{j-2})^2\right]\right\}$$

where σ_w are positive width which is set to 0.9192 for RVM algorithm which is equivalent to 1.69 the best value for the kernel variance as reported in [8] and σ_w is learned in the RUM algorithm. We shall note that, in [8,14], this example fails the RVM algorithm due to numerical instability. After carefully tracking the procedure for the RVM algorithm, we found out what causes the numerical instability. The revised program works well for this example.

To demonstrate the capability of model generalization on unseen data, we use the constructed RBF models by the RUM and RVM algorithms to generate the iterative model output $\widehat{t}_i^d$ by $\mathbf{x}_i^d = [\widehat{t}_{i-1}^d, v_{i-1}, v_{i-2}]^T, \widehat{t}_i^d = t(\mathbf{x}_i^d; \boldsymbol{w})$. The models iterative prediction results are presented in Figures 2.

The one-step prediction and iterative prediction results of 8–11 term RUM model are better than the ones given by 22-term RVM model, see Table 2.

[1] The authors thank Professor Sheng Chen of University of Southampton, UK, who provided us the raw data used in Example 2.



(a). Target of engine data **(b).** Iterative model predictions by the 22-term RVM **(c).** Iterative model predictions by the 8-term RUM

Fig. 2. The target data, the one-step predictions and the iterative model predictions for modeling the relationship between the engine speed and the fuel rack position, where the black solid curve is the target, red solid curve is the one-step or the iterative model predictions generated from RUM while the blue solid curve from RVM.

Table 2. Mean Square Errors of Example 2

Methods	RVM (22*)	RUM (8)	RUM (9)	RUM (10)	RUM (11)
OneStepPrediction MSE	0.00066	0.00051	0.00051	0.00053	0.00052
IterativePrediction MSE	0.00217	0.00188	0.00173	0.00175	0.00180
IterativeLoops	683	89	71	45	46
RBF Width	0.91920	1.54979	1.40733	1.29033	1.47656

*The number of regressors found by the algorithms.

5 Conclusion

The RUM algorithm has been proposed for solving kernel regression modeling problems under the Bayesian inference framework. The overall performance offered by the RUM algorithm is comparable to the results given by RVM algorithm but RUM has superior sparsity, which has been demonstrated on two modeling problems.

References

1. Tipping, M.: The relevance vector machine. In: Solla, S., Leen, T., Müller, K. (eds.) Advances in Neural Information Processing Systems, vol. 12. MIT Press, Cambridge (2000)
2. Bishop, C., Tipping, M.: Variational relevance vector machines. In: Boutilier, C., Goldszmidz, M. (eds.) Uncertainty in Artificial Intelligence 2000, pp. 46–53. Morgan Kaufmann, San Francisco (2000)
3. Tipping, M.: Sparse Bayesian learning and the relevance vector machine. J. Machine Learning Research 1, 211–244 (2001)
4. Tipping, M., Faul, A.: Fast marginal likelihood maximisation for sparse bayesian models. In: Bishop, C., Frey, B. (eds.) Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, Key West, FL (January 2003)
5. Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press, Cambridge (2002)
6. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)

7. Poggio, T., Girosi, F.: A sparse representation for function approximation. Neural Computation 10, 1445–1454 (1998)
8. Chen, S.: Local regularization assisted orthogonal least squares regression. Neuro-Computing 69, 559–585 (2006)
9. Kruif, B., Vries, T.: Support-Vector-based least squares for learning non-linear dynamics. In: Proceedings of 41st IEEE Conference on Decision and Control, Las Vegas, USA, pp. 10–13 (2002)
10. Gestel, T., Espinoza, M., Suykens, J., Brasseur, C., deMoor, B.: Bayesian input selection for nonlinear regression with LS-SVMS. In: Proceedings of 13th IFAC Symposium on System Identification, Totterdam, The Netherlands, pp. 27–29 (2003)
11. Valyon, J., Horváth, G.: A generalized LS-SVM. In: Principe, J., Gile, L., Morgan, N., Wilson, E. (eds.) Proceedings of 13th IFAC Symposium on System Identification, Rotterdam, The Netherlands (2003)
12. Suykens, J., van Gestel, T., DeBrabanter, J., DeMoor, B.: Least Square Support Vector Machines. World Scientific, Singapore (2002)
13. Drezet, P., Harrison, R.: Support vector machines for system identification. In: Proceeding of UKACC Int. Conf. Control 1998, Swansea, U.K, pp. 688–692 (1998)
14. Gao, J., Antolovich, M., Kwan, P.H.: L1 lasso and its Bayesian inference. In: 21st Australasian Joint Conference on Artificial Intelligence, New Zealand (submitted, 2008)
15. Wang, G., Yeung, D.Y., Lochovsky, F.: The kernel path in kernelized LASSO. In: International Conference on Artificial Intelligence and Statistics, pp. 580–587. MIT Press, San Juan (2007)
16. Tibshirani, R.: Regression shrinkage and selection via the LASSO. J. Royal. Statist. Soc. B 58, 267–288 (1996)
17. Roth, V.: The generalized lasso. IEEE Transactions on Neural Networks 15(1), 16–28 (2004)
18. Wu, M., Schölkopf, B., Bakir, G.: A direct method for building sparse kernel learning algorithms. Journal of Machine Learning Research 7, 603–624 (2006)
19. Burges, C.: Simplified support vector decision rules. In: Proc. 13th International Conference on Machine Learning, pp. 71–77. Morgan Kaufman, San Mateo (1996)
20. Snelson, E., Ghahramani, Z.: Sparse gaussian processes using pseudo-inputs. In: Advances in Neural Information Processing Systems 18, pp. 1257–1264. MIT Press, Cambridge (2006)
21. Gao, J.: Robust L1 principal component analysis and its Bayesian variational inference. Neural Computation 20, 555–572 (2008)
22. Lawrence, N.: Probabilistic non-linear principal component analysis with gaussian process latent variable models. Journal of Machine Learning Research 6, 1783–1816 (2005)
23. Billings, S., Chen, S., Backhouse, R.: The identification of linear and nonlinear models of a turbocharged automotive diesel engine. Mech. Syst. Signal Processing 3(2), 123–142 (1989)

Pairwise Constrained Clustering for Sparse and High Dimensional Feature Spaces

Su Yan[1,2], Hai Wang[1,3], Dongwon Lee[1,2], and C. Lee Giles[1,2]

[1] College of Information Sciences and Technology
The Pennsylvania State University
[2] University Park, PA 16802, USA
[3] Dumore, PA 18512, USA
{syan, dongwon, haiwang, giles}@ist.psu.edu

Abstract. Clustering high dimensional data with sparse features is challenging because pairwise distances between data items are *not* informative in high dimensional space. To address this challenge, we propose two novel semi-supervised clustering methods that incorporate prior knowledge in the form of pairwise cluster membership constraints. In particular, we project high-dimensional data onto a much reduced-dimension subspace, where rough clustering structure defined by the prior knowledge is strengthened. Metric learning is then performed on the subspace to construct more informative pairwise distances. We also propose to propagate constraints locally to improve the informativeness of pairwise distances. When the new methods are evaluated using two real benchmark data sets, they show substantial improvement using only limited prior knowledge.

1 Introduction

Clustering is one of the most important and fundamental techniques in data mining, information retrieval, and knowledge management. Most clustering techniques rely on the pairwise distances between data items. However, it is commonly believed that pairwise distances in the high-dimensional space is not informative, and the nearest neighborhood is not meaningful either [2]. As a result, many learning algorithms (including clustering methods) lose their algorithm effectiveness for high dimensional cases.

Recently, semi-supervised clustering has shown effectiveness in improving clustering accuracy by exploring "weak" supervision in the form of pairwise "must-link" or "cannot-link" constraints [1,3,5,9]. That is, if data items a and b are must-linked/cannot-linked, then a and b belong to the same/different cluster(s). Two basic semi-supervised approaches are metric learning that learns a distance measure based on constraints, and constraint enforcement that enforces constraints by modifying the objective function of a clustering method. However, most existing semi-supervised clustering methods have difficulties in handling data with high-dimensional sparse features. For example, in order for a metric-learning method [4,10] to train a distance measure, the number of independent

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 620–627, 2009.
© Springer-Verlag Berlin Heidelberg 2009

variables to be learned is proportional to the dimension of the feature space. For data with hundreds or thousands(e.g., text data) features, metric learning is computationally expensive. Besides, due to the sparseness, only a small portion of total features are covered by constrained data items. Therefore, training a distance measure for sparse features is not effective. For another example, graph-based methods can usually handle high-dimensional data better since they work on the low-dimensional representation (affinity matrix) of the high dimensional data. However, the performance of a graph-based method partially relies on the affinity matrix, which is built upon pairwise distances. Since the pairwise distances in high dimensional space is not informative, the performance of graph-based method is impaired.

In this paper, toward these challenges, we propose two methods to tackle the high-dimensional sparse feature space problem with the help of pairwise constraints. User-provided constraints reflect user's expectation of how the data set should be clustered. Therefore, constraints define a rough clustering structure to a data set. The first method seeks a low-dimensional representation of the data through orthogonal factorizations. The clustering structure defined by prior knowledge is kept and strengthened in the subspace. Metric learning is then performed in the subspace. The second method does space-level generalization of pairwise constraints by local constraints propagation. Both methods can construct more informative pairwise distances for high-dimensional data. Our proposed schemes of exploiting constraints can be applied to any unsupervised clustering model and any high-dimensional data set. We apply the schemes to the widely used Normalized Cut method (NC), and the document data sets to demonstrate the idea and concept. Experimental results base on real data verify the effectiveness and efficiency for our proposals.

2 Main Proposal

2.1 Metric Learning in Structure-Preserving Subspace

This approach is to construct more informative distances between data items through metric learning in the reduced-dimension subspace. The motivation is obvious. In the much reduced-dimension subspace, features are not sparse. Therefore, metric learning is more effective. In addition, since the number of variables to be learned is significantly reduced, metric learning is also more efficient. Most importantly, we argue that user-provided constraints define a clustering structure that best satisfies the user's requirement. If we find a low-dimension representation of data where the clustering structure is more evident, we can expect more informative distances metric to be learned in the subspace.

We now introduce how to find such a structure-preserving subspace. Suppose we have n data items in the full space described by a matrix of column vectors $W \in \mathbb{R}^{f \times n}$, where the feature space is of f dimensions ($f \gg n$). Given pairwise constraints, we first do transitive closure to the constraints and generate d small data groups, where the i-th group with m data items represented by matrix $W_i \in \mathbb{R}^{f \times m}$. Then a matrix $C \in \mathbb{R}^{d \times n}$ is generated by using the centroids of

each groups as column vectors. The above two steps incorporates constraints information into the data representation C. We now seek a data projection by splitting the feature space of C, which is the same as the feature space of W into two parts, attributes and noise. That is, we seek a projection matrix $P = \begin{pmatrix} U & V \end{pmatrix} \in \mathbb{R}^{f \times f}$, $U \in \mathbb{R}^{f \times r}$ and $V \in \mathbb{R}^{f \times s}$, such that $P^T C = \begin{pmatrix} \widehat{C} \\ \widehat{C}^{\perp} \end{pmatrix} = \begin{pmatrix} U_r^T C \\ V_s^T C \end{pmatrix}$, where $r+s = f$. Suppose $\widehat{C}$ is the r-dim attribute part, and $\widehat{C}^{\perp}$ is the s-dim noise, a desired projection satisfies that $\widehat{C}$ is orthogonal to $\widehat{C}^{\perp}$ and $\widehat{C}^{\perp} = V_s^T C = 0$, which means that the structure-irrelevant noise that existed in the full space is now removed by the projection, and only relevant dimensions are kept in the reduced space. Since we only care about the attributes part, all we need to find is the projection U. This projection can be found by computing the orthonormal basis $U \in \mathbb{R}^{f \times r}$ for the column space of C, where $rank(C) = r$. It is easy to see that V is in fact the orthonormal basis of the left null space of C. The subspace data representation is then derived by projecting data using U, that is, $\widehat{W} = U^T W \in \mathbb{R}^{r \times n}$ is the reduced r-dim representation of data. We then do metric learning in the reduced space $\widehat{W}$ for informative distances.

Note that, the idea of using centroid to represent a group of data originates from work [7]. However, [7] solves the classification problem, where the number of data groups is fixed to the number of classes and each data group contains a large amount of training data such that the centroid of a group is the rank-1 approximation with less noise. On the contrary, we solve the clustering problem. The number of data groups generated by transitive closure is not fixed and is usually large. Due to the nature of pairwise constraints and the small amount of available constraints, most of the data groups only contain a very small amount of data items (i.e., 1 or 2). The centroids for such data groups may contain spurious information.

It is easy to see that the sparse-feature problem is solved in the subspace. This is because that U is a full rank matrix and $rank(U) = rank(C) = r$, $rank(\widehat{W}) = r < n \ll f$. The subspace thus provides a more compact data representation than the original full dimensional space. The number of variables to be learned is also greatly reduced. Now, we use the following two Lemmas[1] to show that the clustering structure defined by constraints is more evident in the subspace $\widehat{W}$. Proofs are straightforward and thus skipped.

Lemma 1 (Group Volume Shrinkage Property). *Given any data item w_i and its corresponding centroid c_i, the following holds: $\|\hat{\mathbf{w}}_i - \hat{\mathbf{c}}_i\|_2 \leq \|\mathbf{w}_i - \mathbf{c}_i\|_2$. Since data items in the subspace $\widehat{W}$ get closer to their corresponding centroid than in the full space W, the volume of any given data group shrinks to its centroid.* ∎

Lemma 2 (Constant Center-to-Center Distance). *The pairwise distance between any two given centroids $\mathbf{c}_i$ and $\mathbf{c}_j$ of the full space W is strictly preserved in the subspace $\widehat{W}$: $\|\mathbf{c}_i - \mathbf{c}_j\|_2 = \|\hat{\mathbf{c}}_i - \hat{\mathbf{c}}_j\|_2$.* ∎

[1] We only show the Lemmas in the L_2 norm measure to save space. It is easy to prove that the corresponding theorems still hold for the cosine similarity.

According to Lemma 1, data items in the subspace $\widehat{W}$ move towards their corresponding centroids. According to Lemma 2, any data group W_i keeps constant distance away from any other group W_j in the sense of constant center-to-center distance. Geometrically, the volume of a data group W_i shrinks and groups are still well separated in the subspace. Therefore, the projection in fact strengthens the clustering structure defined by constraints.

2.2 Constraint-Guided Local Propagation

Graph-based methods are well known for clustering high-dimensional data with better accuracy. For example, the representative Normalized Cut (NC) method has been successfully applied to image segmentation and document clustering problems. However, as we mentioned in section 1, the performance of a graph-based method can be impaired by noninformative pairwise distance. We propose a simple yet effective method to directly enforce and propagate constraints on the affinity matrix K. The idea is to do space-level generalization of pairwise constraints based on triangle geometry.

Our idea is the following. Given pairwise constraints that data item x and y must belong to the same cluster (i.e., must-link), we set the distance between the two items as 0, that is $dist(x, y) = 0$. For any other data item z, we set $dist(z, x) = dist(z, y) = \min(dist(z, x), dist(z, y))$. Symmetrically, given pairwise constraints that data item x and y belong to different clusters (i.e., cannot-link), we set $dist(x, y) = 1$, where 1 is the largest value for a normalized distance measure. Since constraints are propagated to at most one-hop neighbors of the constrained data items, we consider this method *local*. After local propagation, the matrix K contains more informative pairwise distances, which enable better clustering performance. The following Lemma justifies our idea. Again, proof is straightforward and thus skipped.

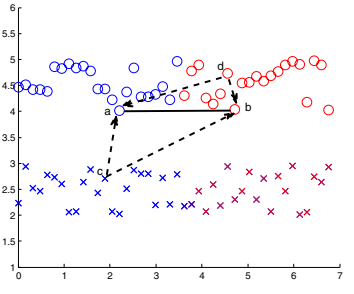


Fig. 1. The two bands data set (The true classification is indicated by marker shape, and the unsupervised K-means clustering results are indicated by marker color. Real line: must-link; Dotted line: cannot-link).

Lemma 3 (Distance propagation property). *Given three data items x, y, and z, suppose $dist(z, x) \leq dist(z, y)$. If data items x and y get closer, the 3rd item z is equally far away from x and y, That is, $dist(x, y) \rightsquigarrow 0$, then $dist(z, y) \rightsquigarrow dist(z, x)$.* ∎

The effectiveness of this method can be illustrated by Figure 1. Unsupervised clustering methods ignore the band structure of the data. If we know that data item a and b belong to the same cluster, and set $dist(d, a) = dist(d, b)$, $dist(c, b) = dist(c, a)$, data items in the upper band will get closer to each other

and the band effect is reduced. That is, constraints on data items a and b are generalized to the whole space. The local propagation method has the time complexity of $O(nq)$, where n is the number of data items and q is the number of constrained data items. It is faster than the related global propagation method [6], which is based on all-pairs-shortest-path and has the $O(n^2q)$ time complexity.

3 Experimental Validation

3.1 Set-Up

Data Sets. We have evaluated the performance of our clustering algorithms using two public available data sets: the *Reuters*-21578 document corpus and the 20-*Newsgroups* 18828 version document corpus. For the Reuters corpus, we included only documents with a *single* label to ensure unambiguous results.We pre-processed each document by tokenization, stop-words removal, and stemming. Terms that appear in only one document are removed.

12 data sets were generated from the two corpora as summarized in Table 1. Without the lose of generality, we first generated data sets of different number of clusters ranging from 2 to 6. For each given cluster number k, 5 test sets were created by first randomly picked k topics from one corpus, and then 40 documents of each of the picked topics were randomly selected and mixed together (Table 1, Reu-2∼6, News-2∼6 report the statistics of the data sets randomly chosen from the pool of 5 data sets for each cluster number k). We also created two challenging data sets from 20-*Newsgroups* corpus. News-Mediocre contains 3 related topics {*talk.politics.misc*, *talk.politics.guns*, and *talk.politics.mideast*}, and News-Difficult contains 3 very similar topics {*comp.windows.x*, *comp.os.ms-windows.misc*, and *comp.graphics*}.

Evaluation Metrics. In addition to the *running time* (RT) as a metric to evaluate the speed of algorithms, to avoid biased accuracy result using a single metric, we evaluate clustering accuracy by employing three widely-used evaluation metrics, which are (1) *Normalized Mutual Information* (NMI), (2) *Rand Index*, and (3) *F-measure*. All the three metrics take values between zero and one, with one meaning best accuracy.

We also implemented four state-of-the-art semi-supervised clustering methods: (1) *L-NC* method does metric learning in the full feature space [10], then uses NC as the unsupervised clustering model; (2) *MPCKmeans* combines metric learning and constraint-enforcement into K-means through an EM process

Table 1. Summary of data sets

20-Newsgroups				*Reuters*			
Data set name	# data items	# features	# cluster	Data set name	# of data items	# features	# cluster
News-2	80	2,422	2	Reu-2	80	1,213	2
News-3	120	3,895	3	Reu-3	120	2,028	3
News-4	160	4,520	4	Reu-4	160	2,347	4
News-5	200	6,203	5	Reu-5	200	2,343	5
News-6	240	5,991	6	Reu-6	240	2,952	6
News-Difficult	300	3,570	3	News-Mediocre	300	4,457	3

Table 2. Summary of all the experimented algorithms

Type	Algorithm	Description
Baseline	NC	Normalized Cut [8]
State-of-the-art	MPCKmeans	learning feature weights & enforce constraint through EM [3]
	L-NC	learning feature weights in the original feature space [10], followed by NC
	C-NC	Constrained Normalized Cut [5]
	Glo-NC	NC with globally adjusted affinity matrix [6]
Our proposals	**RL-NC**	learning feature weights in reduced space (Section 2.1), followed by NC
	RLC-NC	hybrid of **RL-NC** and C-NC
	Lo-NC	NC with locally-adjusted affinity matrix (Section 2.2)

[3]; (3) *C-NC* is a graph-based constraint-enforcement method that has been shown effective in clustering documents [5]; and (4) Glo-NC method *globally* propagates constraints to adjust pairwise distances [6]. Table 2 summarizes the baseline method and seven variations of semi-supervised clustering methods that we have evaluated. Our three proposals – **RL-NC**, **RLC-NC**, and **Lo-NC** – are bold faced.

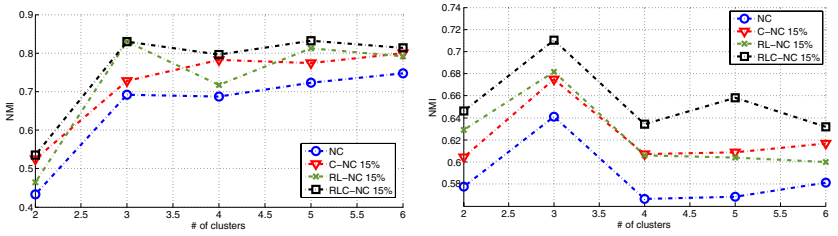
3.2 Experimental Results

Metric Learning. We reported the experimental results based on two challenging data sets News-Mediocre and News-Difficult. We controlled the experiment by varying the amount of constrained data items ranging from 2.5% to 15% of the total documents. Constraints were generated by paring constrained documents based on ground truth.The final performance score was obtained by averaging the scores from 10 test runs.

We compared our sub-space metric learning method (RL-NC) with the full-space learning method (L-NC). Performance comparisons are reported in Table 3. For both data sets and various amount of constraints, RL-NC achieves higher and more stable learning accuracy. Metric learning in the subspace is also much faster than in the full feature space. When the amount of constraints increases, the learning time of both methods increases too, with the subspace learning method scales much better than the full space learning method. Last, note that although the three accuracy metrics (i.e. NMI, RI, and F) show quite different absolute values, they show overall similar patterns for different algorithms. For simple presentation, we will only use NMI as the evaluation metric from here forward.

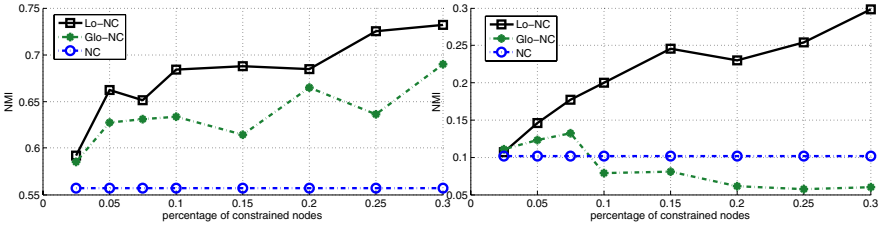
Table 3. Running time and accuracy for subspace metric learning

Algorithm	Metric	*News-Mediocre*					*News-Difficult*				
		2.5%	5%	7.5%	10%	15%	2.5%	5%	7.5%	10%	15%
NC	NMI	0.5568					0.1016				
RL-NC		**0.5865**	**0.6003**	**0.6164**	**0.6405**	**0.6374**	**0.1134**	0.104998	**0.1142**	**0.1151**	**0.1404**
L-NC		0.5220	0.5362	0.4656	0.5246	0.5756	0.1084	**0.1077**	0.1060	0.1060	0.1271
NC	RI	0.7432					0.5098				
RL-NC		**0.7693**	**0.7689**	**0.8029**	**0.8409**	**0.8387**	**0.5427**	0.5242	0.5380	**0.5624**	**0.5550**
L-NC		0.7311	0.7674	0.7112	0.6786	0.8316	0.5398	**0.5478**	**0.5397**	0.5559	0.5363
NC	F	0.6629					0.4401				
RL-NC		**0.6834**	**0.7072**	**0.7222**	**0.7666**	**0.7593**	0.4423	**0.4490**	**0.4424**	0.4436	**0.4636**
L-NC		0.6484	0.6770	0.6455	0.6191	0.7298	**0.4425**	0.4436	0.4418	**0.4443**	0.4555
RL-NC	RT	**0.6129**	**1.0875**	**5.3013**	**9.1093**	**20.4659**	**0.6209**	**2.2153**	**5.2884**	**9.2359**	**17.2411**
L-NC		2.0133	7.9069	18.9226	32.6786	77.0943	0.8736	6.4430	11.2585	26.5236	60.2636



(a) Reuters corpus (b) Newsgroups corpus

Fig. 2. Accuracy of combining *subspace metric learning* (RL-NC) and *constraint enforcement* (C-NC) (% of constraints = 15%)



(a) *News-Mediocre* (b) *News-Difficult*

Fig. 3. Local vs. global propagation

Integrating Metric Learning with Constraint Enforcement. Metric learning and constraint enforcement are two basic schemes of exploiting constraints to improve clustering performance. In this experiment, we combined the two schemes and expected to generate better clustering performance. We compared the subspace metric learning method (RL-NC), with the graph-based constraint enforcement method (C-NC), and with the hybrid approach (**RLC-NC**). Figure 2 shows that both RL-NC and C-NC individually improved upon the regular NC method, and both show similar accuracy given the same amount of constraints. However, the hybrid algorithm, RLC-NC, always significantly outperforms the individual ones. These results empirically validate our hypothesize that the hybrid method, RLC-NC, more comprehensively utilizes available constraints.

Constraint-Guided Local Propagation. We compared the effectiveness of the local propagation method (Lo-NC) with the global adjustment method (Glo-NC). The regular NC approach was adopted as the unsupervised clustering model for both methods. Figure 3 shows the clustering results on the two representative Newsgroups data sets. Lo-NC made sizable improvement over the unconstrained version of NC even with a small amount of constraints. Glo-NC is less effective and its performance for the *News-Difficult* data set is worse than the unconstrained method when the number of constraints increases. Both the local and the global methods exploit the triangle geometry. But the global method also propagates constraints based on the pairwise distances among all the data

Table 4. Lo-NC, Glo-NC on Reuters and Newsgroups corpora (C: % of constraints)

Algorithms	C	Reuters corpus					Newgroup corpus				
		NMI for different # of clusters					NMI for different # of clusters				
		2	3	4	5	6	2	3	4	5	6
Lo-NC	5%	**0.4474**	**0.7462**	**0.7159**	0.7399	**0.742**	0.6179	**0.6724**	**0.5959**	**0.58347**	**0.5972**
Glo-NC		0.4402	0.713	0.7112	**0.7421**	0.7397	**0.7068**	0.6573	0.5789	0.5819	0.5692
Lo-NC	10%	**0.4809**	**0.7202**	**0.7651**	0.7649	**0.7886**	**0.6767**	0.6845	0.6056	**0.6045**	**0.6188**
Glo-NC		0.4557	0.7109	0.7500	**0.7716**	0.7638	0.6169	**0.7031**	**0.6074**	0.5645	0.6014
Lo-NC	15%	**0.5005**	**0.7275**	0.7508	0.777	**0.7817**	**0.667**	**0.694**	**0.6433**	**0.6154**	**0.6384**
Glo-NC		0.4913	0.7038	**0.7564**	**0.7783**	0.7701	0.6371	0.66	0.6033	0.5789	0.6027

points, which may decrease the discriminative power of the constraints. Detailed results on multiple Reuters and Newsgroups data sets are shown in Table 4.

4 Conclusion

Two novel semi-supervised clustering techniques are proposed for high dimensional and sparse data. The first method projects data onto a reduced-dimension subspace such that clustering structure defined by constraints is strengthened. Metric learning is then applied to the subspace to generate informative pairwise distances. The second method exploits the triangle geometry to generalize pairwise constraints by "local" propagation. The validity of our proposals are empirically validated using extensive experiments.

References

1. Basu, S., Bilenko, M., Mooney, R.J.: A probabilistic framework for semi-supervised clustering. In: ACM KDD, Seattle, WA, USA, pp. 59–68 (2004)
2. Beyer, K., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is nearest neighbor meaningful? In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 217–235. Springer, Heidelberg (1998)
3. Bilenko, M., Basu, S., Mooney, R.J.: Integrating constraints and metric learning in semi-supervised clustering. In: ICML, Banff, Alberta, Canada (2004)
4. Cohn, D., Caruana, R., McCallum, A.: Semi-supervised clustering with user feedback. Technical report, Cornell University (2003)
5. Ji, X., Xu, W.: Document clustering with prior knowledge. In: ACM SIGIR, Seattle, WA, USA, pp. 405–412 (2006)
6. Klein, D., Kamvar, S.D., Manning, C.D.: From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In: ICML, Sydney, Australia, pp. 307–314 (2002)
7. Park, H., Jeon, M., Rosen, J.B.: Lower dimensional representation of text data based on centroids and least squares. BIT Numerical Mathematics 43, 427–448 (2003)
8. Shi, J., Malik, J.: Normalized cuts and image segmentation. In: IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI) (2000)
9. Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S.: Constrained k-means clustering with background knowledge. In: ICML, pp. 577–584 (2001)
10. Xing, E., Ng, A., Jordan, M., Russell, S.: Distance metric learning, with application to clustering with side-information. In: NIPS, Vancouver, Canada (2003)

Clustering Documents Using a Wikipedia-Based Concept Representation

Anna Huang, David Milne, Eibe Frank, and Ian H. Witten

Department of Computer Science, University of Waikato, New Zealand
{lh92,dnk2,eibe,ihw}@cs.waikato.ac.nz

Abstract. This paper shows how Wikipedia and the semantic knowledge it contains can be exploited for document clustering. We first create a concept-based document representation by mapping the terms and phrases within documents to their corresponding articles (or concepts) in Wikipedia. We also developed a similarity measure that evaluates the semantic relatedness between concept sets for two documents. We test the concept-based representation and the similarity measure on two standard text document datasets. Empirical results show that although further optimizations could be performed, our approach already improves upon related techniques.

1 Introduction

Clustering is an indispensable data mining technique, particularly for handling large-scale data. Applied to documents, it automatically groups ones with similar themes together while separating those with different topics. Creating a concise representation of a document is a fundamental problem for clustering and for many other applications that involve text documents, such as information retrieval, categorization and information extraction. Redundancy in feature space adds noise and often hurts subsequent tasks. This paper follows our previous work on using Wikipedia to create a *bag of concepts* (BOC) document representation [6]. By *concept* we mean the abstract unit of knowledge represented by a single Wikipedia article. We extend previous work by exploring the semantic relatedness between concepts to calculate the similarity between documents. In the previous work, documents are connected based on the overlap of the concepts that appear in them: this does not take account of the fact that concepts are clearly related to each other. We now explicitly incorporate the semantic connections among concepts into the document similarity measure. This allows us to identify topics that are distinct and yet relate closely to each other—*USA* and *America*, for example—and connect documents at the semantic level regardless of terminological idiosyncrasies. The experiments (Section 4) show that our BOC model together with the semantically enriched document similarity measure outperform related approaches.

Techniques such as *Latent Semantic Indexing* (LSI) [2] and *Independent Component Analysis* (ICA) [7] have been applied to the bag-of-words (BOW) model to find latent semantic word clusters. Representing documents with these clusters also allow subsequent clustering to relate documents that do not overlap in the original word space. In a quest for comparisons with our document similarity measure, we apply LSI and ICA to

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 628–636, 2009.
© Springer-Verlag Berlin Heidelberg 2009

our BOC model, and use the identified latent concept structures as features for clustering. Empirical results show that clustering using these latent structures is outperformed by using the plain BOC model, either with or without the enriched document similarity measure.

The paper proceeds as follows. The next section briefly describes our approach for identifying concepts in a document, each concept being associated with a Wikipedia article. Section 3 extends the semantic relatedness measure between concepts introduced in [10] to compute the semantic similarity of two documents, which forms a basis for clustering. Section 4 presents experiments and discusses results. Related work is reviewed in Section 5; Section 6 concludes the paper.

2 Representing Documents as Wikipedia Articles

In this section we describe our approach for identifying concepts in a document. There are three steps in total: identifying candidate phrases in the document and mapping them to anchor text in Wikipedia; disambiguating anchors that relate to multiple concepts; and pruning the list of concepts to filter out those that do not relate to the document's central thread. The method presented here differs from our previous approach in the way it measures the salience of each concept identified in a document and how it selects the best ones to represent the document.

2.1 Selecting Relevant Wikipedia Concepts

The first step is to map document terms to concepts in Wikipedia. Various approaches have been proposed [3,15,5]. We take the same route as [9], and use Wikipedia's vocabulary of anchor texts to connect words and phrases to Wikipedia articles. Given a plain text document, we first find phrases in it that match Wikipedia's anchor text vocabulary. For example, Wikipedia articles refer to our planet using several anchors, including *Earth*, *the world* and *the globe*. If any of these phrases appear in the document, the article about Earth will be identified as a candidate descriptor. We confine the search for phrases to individual sentences.

2.2 Resolving Ambiguous Terms

Anchors may be ambiguous in that they may refer to different concepts depending on the articles in which they are found. For example, *Pluto* links to 26 different articles, including the celestial body, the Greek god, the Disney character, and a rock band from New Zealand. Disambiguating and selecting the intended concept is essential for creating a correct thematic representation. We use machine learning to identify the correct sense. The input to the classifier is a set of possible targets for a given anchor text and the set of all unambiguous anchors from the surrounding text, which are used as context. The classifier predicts, for each sense, the probability of it being the intended one. The sense with the highest probability is selected. More details about the algorithm can be found in [9].

2.3 Pruning the Concept List

The resulting list of concepts, which together cover the topics mentioned in the input document, is rather long, because phrases are matched against a huge vocabulary; the Wikipedia snapshot we used (dated Nov. 2007) contains just under five million distinct anchors after lower casing. Irrelevant or marginally related concepts must be pruned: they add noise to the representation, which adversely impacts the document similarity calculation and reduces clustering performance. Pruning is based on salience: the average strength of relationship with the other concepts in the document. Let U denote the set of concepts extracted from a document, and salience of concept $c_i \in U$ is defined by:

$$SAL(c_i) = \frac{\sum_{c_j \in U, i \neq j} SIM(c_i, c_j)}{|U|}, \quad (1)$$

where c_j represents the other concepts in U and $|U|$ is the total number of concepts identified in the document. The more concepts c_i relates to and the greater the strength of those relationships, the more salient c_i is. The salience formula depends on $SIM(c_i, c_j)$, the semantic relatedness between two concepts. For this we use Milne and Witten's similarity measure [10]. All concepts in the list are ranked in descending order of SAL, and a fixed proportion t is discarded from the bottom of the list. In our experiments t is set to 0.1 based on empirical observations that this yields the best representation.

It is worth noting that the computational complexity of the above approach is in general linear with the input document length. The disambiguation classifier can be built beforehand, and computing the relatedness between two concepts is a linear operation. The only non-linear calculation is the last step where the averaged relatedness with all the other concepts is computed for each concept in the document. However, this step is restricted to the set of concepts identified from one document and normally the number of concepts per document is moderate. For example, the two datasets used in our experiments have, on average, 24 (OHSUMed) and 20 (Reuters) concepts per document (before pruning) respectively.

3 A Semantically Enriched Document Similarity Measure

Document similarity is typically measured using the cosine of their word vectors, so that matches indicate relatedness and mismatches indicate otherwise. Our new representation allows us to take conceptual relatedness, rather than just lexical overlap, into account. A document d_i is represented by a set of concepts U_i, each with a weight $w(c, d_i)$ (TFIDF value in our experiment). We extend the semantic relatedness between concepts mentioned earlier to the similarity between documents. Given two documents d_i and d_j, their semantic similarity is defined as:

$$Sim_{sem}(d_i, d_j) = \frac{\sum_{\forall c_k \in U_i, \forall c_l \in U_j} w(c_k, d_i) \times w(c_l, d_j) \times SIM(c_k, c_l)}{\sum_{\forall c_k \in U_i, \forall c_l \in U_j} w(c_k, d_i) \times w(c_l, d_j)}. \quad (2)$$

Because $SIM(c_k, c_l)$ is always in [0,1], Sim_{sem} is also bounded within [0,1]. 0 indicates topics in one document are completely unrelated to those in the other, and 1 indicates they are the same topics.

Table 1. Relatedness between the four concepts

	Computer Science (*CS*)	Machine Learning (*ML*)
Data Mining (*DM*)	0.45	0.80
Database (*DB*)	0.51	0.49

We then define the overall similarity between documents d_i and d_j as a linear combination of the cosine similarity Sim_{cos} and Sim_{sem} between two concept vectors:

$$DSim(d_i, d_j) = (1 - \lambda)Sim_{cos}(d_i, d_j) + \lambda Sim_{sem}(d_i, d_j). \quad (3)$$

where λ is a parameter that we set to 0.1 based on preliminary experiments.

In Hu et al.'s approach for semantically enriched document similarity [5], cosine similarity is computed on three aspects: the two document vectors, their category vectors, and their concept vectors enriched with related terms identified from Wikipedia; and the three parts are combined linearly as the final similarity measure. In our approach, the last two parts are unified neatly by a single semantic relatedness measure.

We illustrate our measure with the example used in Hu et al.'s work [5]. Given two concept sets $C_a = \{(CS, 1), (ML, 1)\}$ and $C_b = \{(DM, 1), (DB, 1)\}$, Table 1 shows the relatedness between the four concepts obtained from Milne and Witten's similarity measure. The semantic similarity between document C_a and document C_b is therefore $(0.45\times1+0.51\times1+0.80\times1+0.49\times1)/4 = 0.5625$. This value is close to that obtained by Hu et al. [5], which is 0.57. It is also worth noting that this similarity measure is not only applicable to the BOC model, but also has the potential to be extended to hybrid models where words and concepts are combined, as in [6].

4 Experiments and Results

To focus our investigation on the representation rather than the clustering method, we used the standard k-means algorithm. We created two test sets, following [5], so as to compare our results with theirs[1].

- **Reuters-21578** contains short news articles. The subset created consists of categories in the original Reuters dataset that have at least 20 and at most 200 documents. This results in 1658 documents and 30 categories in total.
- **OHSUMed** contains 23 categories and 18302 documents. Each document is the concatenation of title and abstract of a medical science paper.

4.1 Methodology

Before beginning the experiments we collected all anchor texts in the November 20, 2007 Wikipedia snapshot and lower-cased them. This produced just under five million distinct phrases linking to almost all of the two million articles in the snapshot.

[1] We would like to thank Hu et al. for sharing the OHSUMed dataset.

Documents were preprocessed by selecting only alphabetic sequences and numbers, lower-casing them, and removing concepts that appeared just once across the dataset.

Each document is represented by a vector $\overrightarrow{t_d}$ of *TFIDF* values, each element being a concept. *TFIDF* is defined as $tfidf(d,t) = tf(d,t) \times log(\frac{|D|}{df(t)})$, where t is a concept, $tf(d,t)$ is its frequency in document d, $df(t)$ is its document frequency, and $|D|$ is the total number of documents in the dataset. We set the number of clusters to the number of classes in the data. Each cluster is labeled with its dominant class. Results reported are the average of 5 runs. To compare our results with previous work, we use two evaluation measures: Purity and Inverse Purity. We also use the micro-averaged F-measure [13], weighted by class size, in a separate experiment.

4.2 Evaluation of the Semantic Document Similarity

Table 2 shows how our new document similarity measure performs in clustering on the two datasets. The other rows show the performance of Hu et al.'s algorithm and the baselines to which they compared it to: the traditional BOW, a reimplementation of Hotho et al.'s WordNet-based algorithm [4], and a system that applies Gabrilovich and Markovich's document categorization approach [3] to clustering. Our system and Hu et al.'s achieve comparable results, and are the only two approaches to provide substantial improvements over the baseline. We obtained better inverse purity because classes are more concentrated into clusters rather than dispersed across multiple clusters.

To further explore the differences between these approaches, let us take a closer look at one document that was clustered. Table 3 compares some of the concepts produced when each of the systems is asked to cluster Reuters document #15264 (results for other approaches were taken from [5]). This document discusses ongoing attempts by Teck Cominco—a Canadian mining company—to begin a joint copper-mining venture in Highland Valley, British Columbia. All of the approaches are able to pick up on the different minerals and units—*copper*, *silver*, *ounce*—and will (implicitly or explicitly) relate to synonyms such as *Cu* and *oz*. The first system, by Hotho et al., does so using WordNet, a lexical rather than encyclopedic resource. Thus it fails to pick up specific named entities such as *Teck Cominco*, but will identify terms that do not resolve to Wikipedia articles, such as *complete*. Each of the terms shown in the table can be further expanded with WordNet semantic relations; *copper* can be expanded with the associated term *cupric* and the hypernyms *metallic element*, *metal* and *conductor*.

All of the latter three approaches use Wikipedia. The approach inspired by Gabrilovich and Markovich gathers Wikipedia concepts through term overlap with

Table 2. Comparison with related work in terms of clustering purity

Dataset	Reuters			OHSUMed		
	Purity	Inverse	Impr.	Purity	Inverse	Impr.
Bag of Words	0.603	0.544	-	0.414	0.343	-
Gabrilovich and Markovich	0.605	0.548	0.33%	0.427	0.354	3.17%
Hotho et al.	0.607	0.556	0.66%	0.435	0.358	4.72%
Hu et al.	0.655	0.598	8.62%	0.459	0.388	12%
Ours	**0.678**	**0.750**	**12.4%**	**0.474**	**0.528**	**14.5%**

Table 3. Comparing features generated by different approaches

Hotho	copper; venture; highland; valley; british; columbia; affiliate; mining; negotiation; **complete**; administration; reply; silver; ounces; molybdenum
Gabri	Teck; John Townson; Cominco Arena; Allegheny Lacrosse Officials Association; **Scottish Highlands**; Productivity; Tumbler Ridge, British Columbia; Highland High School; **Economy of Manchukuo**; Silver; **Gold (color)**; **Copper (color)**;
Hu	**Tech Cominico**; British Columbia; Mining; Molybdenum; Joint Venture; Copper
Ours	Mining; Joint venture; Copper; Silver; Gold; Ore; Management; Partnership; Product (business); Ounce; Negotiation; Molybdenum; **Teck Cominco**; Vice president; Consortium; Short ton;

the document. This unfortunately allows tenuously related concepts such as *Scottish Highlands* and the *Economy of Manchukuo* to creep into the representation and cause problems. Additionally this system performs disambiguation only indirectly, which introduces more irrelevant concepts such as *Copper (color)*.

The last two systems have the tightest representation of the document, because they only contain the Wikipedia concepts that are directly discussed. Both are then able to expand out from these concepts to identify related documents regardless of textual overlap. Hu et al.'s system considers broader topics mined from the categories to which each article belongs, and associated topics mined from the links extending out from each article. Thus *Teck Cominco* is expanded with *Mining companies in Canada* and *Con Mine* in [5]. Our system, in comparison, does not need to expand concepts beforehand. Instead it can compare any two Wikipedia concepts as required, based on the relatedness measure introduced in [10]. *Teck Cominco* is essentially expanded on demand with a huge pool of possibilities, such as different mining companies (*Codelco*, *De Beers*, and about a hundred others), tools (*Drilling rig*, *Excavator*, etc.) and locations (the *Pebble Mine* in Alaska, for example). All of these new concepts are weighted with a proven relatedness measure [10], and only the concepts that are necessary to connect two related documents are ever considered.

4.3 Latent Semantic Indexing with Concepts

As an additional experiment, we apply Latent Semantic Indexing (LSI) and Independent Component Analysis (ICA) on the BOC representation (concept vectors with TFIDF values). LSI and ICA find latent structures/independent components respectively by analyzing the *concept-document* matrix. The purpose is to use the identified latent concept clusters as features for clustering and compare its effectiveness in connecting documents that do not overlap in the original concept space with using the semantically enriched document similarity measure defined in Section 3.

The only work to our knowledge so far that uses LSI with features extracted using Wikipedia is [11], where LSI is used to reduce dimensionality and Wikipedia is used to enrich text models for text categorization. Instead we use Wikipedia to extract

Table 4. Performance of LSI and ICA on BOC model on Reuters dataset

Approach	Dimensionality	Purity	Inverse Purity	FMeasure
BOC with Sim_{cos}	2186	0.667	0.750	0.561
BOC with $DSim$	2186	0.678	0.750	0.575
BOC + LSI	546	0.353	0.450	0.195
BOC + ICA	546	0.414	0.649	0.201

concepts from the input document and apply LSI/ICA directly to the BOC model that is generated. ICA has been applied to text documents in [8] and found to produce better group structures in feature space than LSI. We used the FastICA program[2] with its default settings. For a fair comparison, the number of independent components in ICA is set to the number of eigenvalues retained in LSI. The cosine measure (Sim_{cos}) is used throughout this experiment.

Table 4 shows the performance of using latent concept groups as features for clustering on the Reuters dataset. The OHSUMed dataset could not be processed because it is computationally prohibitive. The results show that the latent concept groups are not as effective as the original concepts: using cosine similarity on the BOC model (ie. based on overlaps between concept sets) still outperforms. This could be explained by the fact that ICA and LSI are applied globally and do not use any knowledge about the categories in the datasets, so the latent semantic structures that are found do not retain sufficient discriminative information to differentiate the classes [14]. Local alternatives for LSI and ICA may be better choices; but are beyond the scope of this paper.

5 Related Work

Document representation is a fundamental issue for clustering, and methods such as BOW, bags of phrases and n-grams have been widely investigated. Explicitly using external knowledge bases can assist generating concise representations of documents. Related work in this area includes Hotho et al. [4] and Recupero [12]; both use relations defined in WordNet to enrich BOW. Techniques such as Latent Semantic Indexing and Independent Component Analysis have been used to find latent semantic structures in dataset [2,8]; each structure is a linear combination of the original features (typically words). Representing documents with these latent structures can reduce the dimensionality of feature space while retaining essential semantic information, yielding significant improvement in subsequent tasks, in information retrieval [2] for example.

Despite widespread adoption for many tasks, only a limited amount of work has investigated utilizing Wikipedia as a knowledge base for document clustering [1,5,6]. Our previous work focuses on how to generate the concept-based representation for text documents and use Wikipedia to provide supervision for active learning [6]; the present paper focuses on extending the relatedness between concepts to measuring the relatedness between documents, evaluating the impact of the semantically enriched document similarity measure on clustering, and gives a more detailed analysis of the

[2] http://www.fastica.org/

concept-based document representation. The algorithm described in this paper also differs from our previous one in how it selects the best concepts to represent each document.

Our approach differs markedly from that of Hu et al. [5]. Our process for selecting and disambiguating terms to identify relevant Wikipedia concepts draws directly on previous work [9] and has been separately evaluated against manually-defined ground truth. In contrast, theirs was developed specifically for the task and has not been investigated independently. Another significant difference is the way in which the document similarity measures are calculated. They develop their own methods of measuring similarity through Wikipedia's category links and redirects, and append this to the traditional metric obtained from the BOW model. We instead start with an independently proven method of measuring relatedness between concepts [10] that takes all of Wikipedia's hyperlinks into account, and generalize this to compare documents.

6 Conclusions

This paper has presented a new approach to document clustering that extends a semantic relatedness measure defined between concepts in Wikipedia to measure document similarity. Results on two datasets prove the effectiveness of our BOC model and the enriched document similarity measure. We also investigated clustering based on a transformed feature space that encodes semantic information derived directly from the dataset, by applying LSI and ICA to the BOC model and using the latent semantic structures instead of original concepts as features for clustering, as a comparison to using the semantically enriched document similarity. Results suggest that these techniques do not improve clustering using the BOC model when performed globally.

We also observed from our earlier work [6] that BOC model can often be improved by adding further words from the document that are not represented in the BOC model, especially when the topics involved are similar. Yet in this paper we consider only concepts, albeit with an approved similarity measure. This suggests a hierarchical approach: first cluster coarsely using the BOC model, and refine clusters using hybrid models like the *Replaced* model in [6]—another interesting avenue for future work.

References

1. Banerjee, S., Ramanathan, K., Gupta, A.: Clustering Short Texts using Wikipedia. In: Proceedings of the SIGIR, pp. 787–788. ACM, New York (2007)
2. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T., Harshman, R.: Indexing by Latent Semantic Analysis. Journal of the American Society for Information Science 41(6), 391–407 (1990)
3. Gabrilovich, E., Markovitch, S.: Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. In: Proceedings of AAAI, pp. 1301–1306. AAAI, Menlo Park (2006)
4. Hotho, A., Staab, S., Stumme, G.: WordNet improves Text Document Clustering. In: Proceedings of SIGIR Semantic Web Workshop, pp. 541–544. ACM, New York (2003)
5. Hu, J., Fang, L., Cao, Y., Zeng, H.J., Li, H., Yang, Q., Chen, Z.: Enhancing Text Clustering by Leveraging Wikipedia Semantics. In: Proceedings of SIGIR, pp. 179–186. ACM, New York (2008)

6. Huang, A., Milne, D., Frank, E., Witten, I.H.: Clustering Documents with Active Learning using Wikipedia. In: Proceedings of ICDM, pp. 839–844. IEEE, Los Alamitos (2008)
7. Hyvärinen, A., Karhunen, J., Oja, E.: Independent Component Analysis. Wiley Interscience, Hoboken (2001)
8. Kolenda, T., Hansen, L.K.: Independent Components in Text. In: Girolami, M. (ed.) Advances in Independent Component Analysis, ch. 13, pp. 235–256. Springer, Heidelberg (2000)
9. Milne, D., Witten, I.H.: Learning to Link with Wikipedia. In: Proceedings of CIKM, pp. 509–518. ACM, New York (2008)
10. Milne, D., Witten, I.H.: An Effective, Low-Cost Measure of Semantic Relatedness obtained from Wikipedia Links. In: Proceedings of AAAI Workshop on Wikipedia and Artificial Intelligence (WIKIAI), pp. 25–30. AAAI, Menlo Park (2008)
11. Minier, Z., Bodo, Z., Csato, L.: Wikipedia-Based Kernels for Text Categorization. In: Proceedings of SYNASC, pp. 157–164. IEEE, Los Alamitos (2007)
12. Recupero, D.R.: A New Unsupervised Method for Document Clustering by Using WordNet Lexical and Conceptual Relations. Information Retrieval 10, 563–579 (2007)
13. van Rijsbergen, C.J.: Information Retrieval. Butterworths, London (1979)
14. Torkkola, K.: Discriminative Features for Document Classification. In: Proceedings of ICPR, pp. 10472–10475. IEEE, Los Alamitos (2002)
15. Wang, P., Hu, J., Zeng, H.J., Chen, L., Chen, Z.: Improving Text Classification by Using Encyclopedia Knowledge. In: Proceedings of ICDM, pp. 332–341. IEEE, Los Alamitos (2007)

An Instantiation of Hierarchical Distance-Based Conceptual Clustering for Propositional Learning*

A. Funes[1,2], C. Ferri[1], J. Hernández-Orallo[1], and M.J. Ramírez-Quintana[1]

[1] DSIC, Universidad Politécnica de Valencia,
Camino de Vera s/n, 46022 Valencia, España
[2] Universidad Nacional de San Luis, Ejército de los Andes 950,
5700 San Luis, Argentina
{afunes, cferri, jorallo, mramirez}@dsic.upv.es

Abstract. In this work we analyse the relationship between distance and generalisation operators for real numbers, nominal data and tuples in the context of hierarchical distance-based conceptual clustering (HDCC). HDCC is a general approach to conceptual clustering that extends the traditional algorithm for hierarchical clustering by producing conceptual generalisations of the discovered clusters. This makes it possible to combine the flexibility of changing distances for several clustering problems and the advantage of having concepts which are crucial for tasks as summarisation and descriptive data mining in general. In this work we propose a set of generalisation operators and distances for the data types mentioned before and we analyse the properties by them satisfied on the basis of three different levels of agreement between the clustering hierarchy obtained from the linkage distance and the hierarchy obtained by using generalisation operators.

Keywords: conceptual clustering, hierarchical clustering, generalisation, distances, propositional learning.

1 Introduction

One issue related to some data mining techniques is the lack of comprehensibility. Although several learning techniques have been tested as useful in the way that they offer good predictions, they do not give a description, pattern or generalisation which justifies the decision made for a given individual. For instance, it is useful to know that a given molecule belongs to a cluster according to a certain distance measure, but it is even more interesting to know what the chemical properties shared by all the molecules in that cluster are.

Lack of comprehensibility is a common issue to clustering and classification techniques based on distances. The source of this problem is the dichotomy between distances and generalisations. It is well known that distances and generalisations give

* This work has been partially supported by the EU (FEDER) and the Spanish MEC/MICINN under grant TIN2007-68093-C02 and the Spanish project "Agreement Technologies" (Consolider Ingenio CSD2007-00022). A. Funes was supported by a grant from the Alfa Lernet project and the UNSL.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 637–646, 2009.
© Springer-Verlag Berlin Heidelberg 2009

rise to two different approaches in data mining and machine learning. On the one hand we have distance-based techniques, where we only need to count on a distance function for the data we are working with. However, distance-based techniques (such as [11, 12, 13]) do not provide patterns or explanations justifying the decisions made. On the other hand we have symbolic techniques [7, 8, 9, 10] that, unlike distance-based methods, are founded on the idea that a generalisation or pattern discovered from old data can be used to describe new data covered by this pattern.

An important issue when combining both techniques is to know whether the patterns discovered for each cluster by a distance-based technique are consistent with the underlying distance used to construct the clusters. Inconsistencies can arise when the notion of distance and generalisation are considered independently. That is, given a set of examples and a generalisation of them, it is expected that those examples that are close in a metric space according to its distance are covered by the generalisation, while those examples that are far away are expected to be outside the generalisation coverage. This problem has been extensively treated in [6]. In the present work we focus on the relationship between distances and generalisations in the context of HDCC [1], a general approach for agglomerative hierarchical clustering [2, 3]. HDCC, that stands for Hierarchical Distance-based Conceptual Clustering, constructs a cluster hierarchy by using a distance at the same time that in produces a hierarchy of patterns resulting in an extended dendrogram referred as conceptual dendrogram. The main aspect considered in [1] and that has been ignored by other conceptual clustering methods that use distances is knowing a priori whether the hierarchy of clusters induced by the underlying distance is consistent with the discovered patterns, i.e. how much the cluster elements covered by a given pattern reproduce the distribution of the elements in the metric space. Accordingly, in [1] three different levels of consistency between a distance and a generalisation operator have been defined.

The present work is an instantiation for the propositional learning case of the general framework presented in [1]. Here, we give the results of a formal analysis carried out for a set of distances and generalisation operators useful for propositional clustering, where we prove that intervals and absolute difference distance for real numbers, and the union set and discrete distance for nominal data work well together in HDCC. More importantly, we have also shown that it is also the case when using them as generalisation operators and distances for tuples of real numbers and nominal data. This rounds up the approach for propositional learning. But, additionally, this composability result for tuples is obtained independently from the base data types. The property of composability allows our framework to be directly extended to tuples of any complex data type provided that the generalisation operators associated to the component data types satisfy the property wanted for tuples. For instance, we can assert properties of tuples of graphs, strings and numbers provided we know the properties for the underlying data types. Besides these theoretical results we also present some experiments.

The paper is organised as follows. Due to space limitations, all necessary preliminary concepts about the HDCC approach can be found in [1] and the proposition proofs can be found in [14]. In Section 2 we propose pairs of generalisation operators and distances for numerical and nominal data, which are used in turn to define generalisation operators and distances for tuples. In Section 3 we present some experiments by applying the operators and distances proposed in

Section 2, and we also compare the results obtained in HDCC wrt. traditional hierarchical clustering. Finally, Section 4 closes the paper with the conclusions and future work.

2 Instantiation for Propositional Learning

In this section, we present an instantiation of HDCC for propositional clustering where flat data are expressed in terms of attributes and instances. We propose generalisation operators for numerical and categorical data and also for tuples, which are the data types typically used in propositional learning. In all cases the different levels of consistency defined in [1] between the proposed operators and distances have been verified through a satisfability analysis of the strong and weak boundedness and acceptability properties given in [1].

2.1 Nominal Data

A nominal data type, also referred as enumeration or categorical data type denotes a finite set of possible values that an attribute can take, e.g. gender, days of the week, colours, etc. A Boolean data type is a special case where there are only two possibilities. The metric space for nominal data type is composed of a set X, which is just a finite set of symbolic values, and a distance d.

There are many distances defined for nominal values. Some of the most commonly used distances are the discrete distance –that returns 0 when both values match and 1 otherwise– and the VDM (Value Difference Metric) distance [4], among others. In some cases, a distance defined by the user can be useful. For instance, in the metric space (X, d) where X = {XXL, XL, L, M, S, XS, XXS}, the distance d defined as d(XXL, XL) = 1, d(XXL, L) = 2, d(XXL, M) = 3, d(XXL, S) = 4, d(XXL, XS) = 5, d(XXL, XXS) = 6, d(XL, L) = 1, d(XL, M) = 2, d(XL, S) = 3, d(XL, XS) = 4, d(XL, XXS) = 5, d(L, M) = 1, d(L, S) = 2, d(L, XS) = 3, d(L, XXS) = 4, d(M, S) = 1, d(M, XS) = 2, d(M, XXS) = 3, d(S, XS) = 1, d(S, XXS) = 2, d(XS, XXS) = 1 organizes the points into a line where XXL and XXS are the extreme points.

Typical patterns for nominal data are expressed as conditions over the values of the attributes, e.g. *attributeName* = XL or *attributeName* ≠ XL. However, since X is finite, the coverages[1] of the possible patterns are also finite and they can be expressed extensionally as subsets of X. Thus, the pattern language $\mathcal{L}$ for nominal data can reduce to 2^X.

We propose for the generalisation of a pair of nominal values the set that contains both values.

Proposition 1. *Let (X, d) be a metric space, X a set of nominal data, and* 2^X *the pattern language. The function* $\Delta: X \times X \rightarrow 2^X$ *defined by* $\Delta(e_1, e_2) = \{e_1, e_2\}$ *is a binary generalisation operator*[2] *for nominal data.*

Given that patterns are sets of nominal values, we propose the set union as the generalisation of two patterns.

[1] See [1] for definition.of *coverage*.
[2] See [1] for definition.of *binary generalisation operator*.

Proposition 2. *Let (X, d) be a metric space, X a set of nominal data, and 2^X the pattern language. The function Δ^*: $2^X \times 2^X \rightarrow 2^X$ defined by $\Delta^*(s_1, s_2) = s_1 \cup s_2$ is a pattern binary generalisation operator[3] wrt. 2^X.*

Proposition 3 gives the properties satisfied by the proposed operators Δ and Δ^*.

Proposition 3. *Let Δ and Δ^* be the generalisation operators given in and Proposition 2, d a distance between nominal data and d_L a linkage distance. Δ and Δ* are (i) strongly bounded[4] by d and d_L, respectively; (ii) weakly bounded[5] by d and d_L, respectively; (iii) acceptable[6].*

The example in Fig (left) shows the use of HDCC for the evidence E = {XXS, S, M, XXL}. We have used the discrete distance, and the generalisation operators given in and Proposition 2 to compute the patterns. Note that applying the user-defined distance given above, the dendrogram changes to that shown in Fig (right).We can also affirm by Proposition 1 in [1] and Proposition 3 that both conceptual dendrograms are equivalent to the corresponding traditional dendrograms.

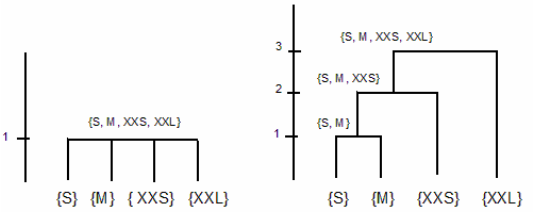


Fig. 1. Two applications of HDCC to nominal data under the single linkage distance, using the discrete distance (left) and a user-defined distance (right)

2.2 Numerical Data

Numerical data are widely used to express amounts and measures and many attributes of real word objects. A well known metric space for numeric data is ($\Re$, d) where d is the distance defined as the absolute difference of two real numbers, i.e. $d(e_1, e_2) = |e_1 - e_2|$. A usual generalisation for a set of numbers is the minimal interval whose extreme values are the least and the greatest values in the set. Thus the pattern language $\mathcal{L}$ we consider here is the set of all the finite closed intervals in $\Re$. We propose for the generalisation in $\mathcal{L}$ of two elements in $\Re$ the minimal interval that includes both elements.

Proposition 4. *Let $\mathcal{L}$ be the set of all the finite closed intervals in $\Re$. For all e_1, e_2 in $\Re$ such that $e_1 \leq e_2$, the function Δ: $\Re \times \Re \rightarrow \mathcal{L}$ defined by $\Delta(e_1, e_2) = [e_1, e_2]$ is a binary generalisation operator for real numbers.*

Next we propose for the generalisation of two intervals the minimal interval that covers both.

---

[3] See [1] for definition.of *pattern binary generalisation operator*.
[4] See [1] for definition of *strongly bounded.*
[5] See [1] for definition of *weakly bounded.*
[6] See [1] for definition of *acceptable.*

Proposition 5. *Let $\mathcal{L}$ be the set of all the finite closed intervals in $\Re$. The function Δ^*: $\mathcal{L} \times \mathcal{L} \rightarrow \mathcal{L}$ defined by $\Delta^*([e_{i1}, e_{f1}], [e_{i2}, e_{f2}]) = [e_i, e_f]$, where e_i is the least value in $\{e_{i1}, e_{i2}\}$ and e_f is the greater value in $\{e_{f1}, e_{f2}\}$, is a pattern binary generalisation operator wrt. $\mathcal{L}$.*

Proposition 6. *Let Δ and Δ^* be the generalisation operators of Proposition 4 and Proposition 5, d the absolute difference between numbers and d_L a linkage distance. Δ and Δ^* are (i) strongly bounded by d and d_L, respectively; (ii) weakly bounded by d and d_L, respectively; (iii) acceptable.*

Fig 2 shows a simple application of HDCC under single linkage using the proposed operators and distance for real numbers. By Proposition 1 in [1] and by Proposition 6 the conceptual dendrogram is equivalent to the traditional one.

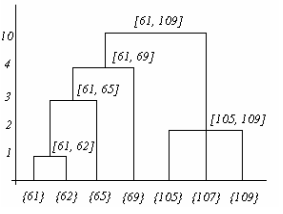


Fig. 2. Conceptual dendrogram using single linkage distance for a set of real numbers

2.3 Tuples

A tuple is a widely-used structure for knowledge representation in propositional learning since examples are represented as tuples of nominal and numerical data.To define a generalisation operator for tuples, unlike the previous data types, we base it on the properties of the basic types from which the tuple type is constructed. We assume they are embedded in metric spaces, therefore we can use the distances defined over each space to define distances between tuples. Analogously, to define the pattern language for tuples, we also use the pattern languages defined for each space.

Let (X_i, d_i) be a collection of metric spaces and $\mathcal{L}_i$ a collection of pattern languages $(i=1,\ldots,n)$ corresponding to each of the n dimensions of a tuple. We denote X the space $X_1 \times \ldots \times X_n$. Therefore, if $x \in X$ then x is a n-tuple $(x_1,\ldots, x_n)$, where $x_i \in X_i$.

Let $d_i(\cdot,\cdot)$ be a distance function defined over X_i $(i=1,\ldots,n)$. The expressions shown in Table 1 are distance functions in X. In that follows, we denote as d_T any of them.

We define the pattern language for tuples $\mathcal{L}$ by using the basic pattern languages $\mathcal{L}_i$ $(i=1,\ldots,n)$ as $\mathcal{L}=(\mathcal{L}_1,\ldots, \mathcal{L}_n)$. Thus, the generalisation Δ of two tuples x and y (formalized by Proposition 7 below) can be defined as the tuple whose components are the generalisations of the respective components in x and y, while the coverage of a pattern in $\mathcal{L}$ is given by Definition 1.

Definition 1. *Given $p = (p_1,\ldots, p_n) \in \mathcal{L}$, the coverage Set(p) of the pattern p over $\mathcal{L}$ is defined as $\{(x_1,\ldots, x_n) \in X \mid x_i \in Set(p_i), i = 1,\ldots, n\}$.*

Table 1. Some distance functions for tuples

$d(x,y)=\sum_{i=1}^{n} d_i(x_i,y_i)$ *Manhattan distance*	$d(x,y)=\sqrt{\sum_{i=1}^{n} d_i(x_i,y_i)^2}$ *Euclidean distance*	$d(x,y)=\max_{1\le i\le n} d_i(x_i,y_i)$ *Box or Chebyshev distance*
$d(x,y)=\sum_{i=1}^{n} \alpha_i.d_i(x_i,y_i)$ *Weighted Manhattan*	$d(x,y)=\sqrt{\sum_{i=1}^{n} \alpha_i.d_i(x_i,y_i)^2}$ *Weighted Euclidean distance*	$d(x,y)=\max_{1\le i\le n} \alpha_i.d_i(x_i,y_i)$ *Weighted Box distance*

For example, given the pattern p = ([34, 54], {XXL, XL, XS, XXS}, [0, 130]), the examples e_1 = (54, XXL, 100) and e_2 = (36, XS, 60) are covered by the pattern. However, the tuple (40, M, 70) is not covered by p since M $\notin$ *Set*({XXL, XL, XS, XXS}).

Proposition 7. *Let $X = X_1 \times ... \times X_n$ be the space of tuples, $\mathcal{L}_i$ (i=1,…,n) a pattern language on the basic type X_i; Δ_i: $X_i \times X_i \to \mathcal{L}_i$ a binary generalisation operator in X_i and $\mathcal{L} = (\mathcal{L}^1,\ldots, \mathcal{L}^n)$ the pattern language of tuples. The function Δ: $X \times X \to \mathcal{L}$ defined by $\Delta((x_1,\ldots,x_n), (y_1,\ldots,y_n)) = (\Delta_1(x_1,y_1),\ldots,\Delta_n(x_n,y_n))$ is a binary generalisation operator for X.*

Given that patterns in $\mathcal{L}$ are tuples whose elements are patterns in $\mathcal{L}_i$, the generalisation of two tuples of patterns p and q can be defined as the tuple whose components are the generalisations of the respective components in p and q. This is formalised in Proposition 8.

Proposition 8. *Let $\mathcal{L} = (\mathcal{L}^1,\ldots, \mathcal{L}^n)$ be a pattern language for tuples, with $\mathcal{L}_i$ (i=1,…,n) a pattern language on a basic type X_i; Δ^*_i: $\mathcal{L}_i \times \mathcal{L}_i \to \mathcal{L}_i$ a pattern binary generalisation operator in $\mathcal{L}_i$. The function Δ^*: $\mathcal{L} \times \mathcal{L} \to \mathcal{L}$ defined by $\Delta^*((p_1, \ldots, p_n), (q_1, \ldots, q_n)) = (\Delta^*_1(p_1, q_1),\ldots,\Delta^*_n(p_n, q_n))$ is a pattern binary generalisation operator wrt. $\mathcal{L}$.*

In HDCC, generalisations of unitary sets are computed as the generalisation of the element with itself. Therefore, the pattern associated to a cluster with only one tuple $\{(x_1, \ldots, x_n)\}$ is given by $\Delta((x_1,\ldots, x_n), (x_1,\ldots, x_n))$, i.e. $(\Delta_1(x_1, x_1),\ldots, \Delta_n(x_n, x_n))$.

Proposition 9. *(Composability of Δ) The binary generalisation operator Δ for tuples given by Proposition 7 when applied to tuples in the space $X = X_1 \times \ldots \times X_n$, where (X_i,d_i) (i = 1,…,n) is a metric space equipped with a binary generalisation operator Δ_i is:*

(*i*) *Strongly bounded by d_T if Δ_i is strongly bounded by d_i, $\forall$ i: i = 1,…,n.*
(*ii*) *Weakly bounded by d_T if Δ_i is strongly bounded by d_i, $\forall$ i: i = 1,…,n.*
(*iii*) *Acceptable if Δ_i is acceptable, $\forall$ i: i = 1,…,n.*

The dendrograms shown by Fig. (d) and (e) can be seen as instantiations of propositional clustering in $X = \Re \times \Re$ for the evidence given in Fig. (a). We have used the language of closed intervals in $\Re$ as pattern language for each dimension in X, and the absolute difference as the distance between real numbers. Note that a tuple pattern, in this case, describes an axis-parallel rectangle. Fig. (d) shows the conceptual dendrogram resulting from the application of HDCC using the single linkage distance d^s_L while Fig. (e) using the complete linkage distance d^c_L. We can

see that the conceptual dendrogram is not equivalent to the traditional one under single linkage given that although the binary generalisation operator Δ for tuples given in Proposition 7 is strongly bounded by d_T by Proposition 9, Δ^* is not strongly bounded by d^s_L since the generalisation of two rectangles *p1* and *p2* associated to clusters C_1 and C_2 is a rectangle p that cover points that can fall outside the balls with centre in the linkage points of C_1 and C_2 and radius $d^s_L(C_1, C_2, d_T)$, as it happens for instance with $\{i\}$ which is covered by *p4* (see Fig. 3 (b)).

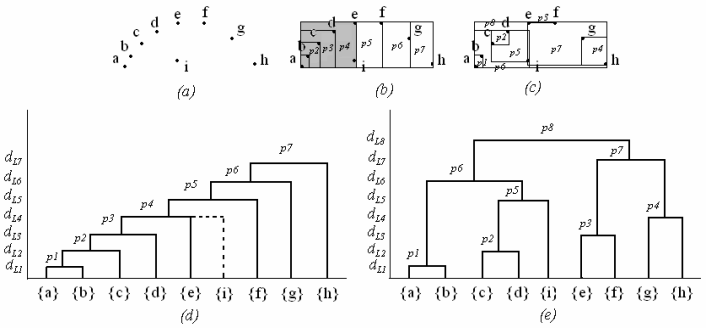


Fig. 3. (*a*) A set of points in $\Re\times\Re$. (*b*) Discovered patterns under d^s_L. (*c*) Discovered patterns under d^c_L. (*d*) Application of HDCC for tuples using d^s_L and (*e*) using d^c_L.

Note that the same could happen for tuples in $X_1\times\ldots\times X_n$ when at least two domains X_i are instantiated to $\Re$. Let us consider the following example. C_1={(0, 0, x_3,…, x_n), (1, 1, x_3,…, x_n), (2, 2, x_3,…, x_n), (4, 4, x_3,…, x_n)} and C_2 = {(5.1, 5.1, x_3,…, x_n)} with patterns *p1* = ([0, 4], [0, 4], p_3,…, p_n) and *p2* = ([5.1, 5.1], [5.1, 5.1], p_3, …, p_n), respectively. We have that $\Delta^*(p1, p2) = p$ = ([0, 5.1], [0, 5.1], p_3,…, p_n) and $d^s_L(C_1, C_2, d_T)$ = 1.55 where d_T is the Euclidean distance. However, there exists x = (4.5, 0.5, x_3,…,x_n) that is covered by p but $d^s_L(\{x\}, C_1, d)$ = 2.91 > 1.55, and $d^s_L(\{x\},C_2,d)$ = 4.63 > 1.55.

In fact, the composability property of Δ^* can only be proved wrt. the complete linkage distance d^c_L, as the next proposition establishes.

Proposition 10. *(Composability of Δ*) The pattern binary generalisation operator Δ^* for tuples in the space $X = X_1\times\ldots\times X_n$ given by Proposition 8 when applied to patterns in the space $\mathcal{L} = \mathcal{L}_1\times\ldots\times\mathcal{L}_n$, where $\mathcal{L}_i$ (i = 1,…,n) is a pattern language for elements in X_i and (X_i, d_i) is a metric space equipped with a pattern binary generalisation operator Δ^*_i, is:*

(*i*) *Strongly bounded by d^c_L if Δ^*_i is strongly bounded by d^c_L, $\forall$ i: i = 1,…,n.*
(*ii*) *Weakly bounded by d^c_L if Δ^*_i is strongly bounded by d^c_L, $\forall$ i: i = 1,…,n.*
(*iii*) *Acceptable if Δ^*_i is acceptable, $\forall$ i: i = 1,…,n.*

We can see in Fig 3 (c) that the application of HDCC to $X = \Re\times\Re$ under complete linkage produces a conceptual dendrogram that is equivalent to the traditional dendrogram as Proposition 1 in [1] establishes given that Proposition 10 (i) and Proposition 9 (i) hold.

3 Experimental Results

In the previous section we proposed a set of generalisation operators and distances for tuples that applied to HDCC under complete linkage distance produces equivalent conceptual dendrograms with the additional advantage of providing a description of each cluster in the hierarchy. We have also seen through an example that the same operators and distances when used under single linkage distance can produce dendrograms that are not equivalent. The experiments described in this section are aimed to (i) empirically illustrate the first result with a real dataset and (ii) show that the new conceptual clustering, coming from the on-line re-arrangement of the dendrogram, although not equivalent to the traditional dendrogram does not undermine cluster quality when applied under single linkage.

A first experiment was conducted on the Iris Dataset [5]. The dataset consists of three classes, 50 instances each and four numeric attributes. Each class refers to a type of iris plant namely Iris Setosa, Iris Versicolor and Iris Virginica. The numeric attributes refers to the sepal and petal lengths and widths in cms.

To assess the quality of the clustering we employed two different measures: (i) One internal measure, called *S,* which reflects the mean scattering over k clusters with n_i (i = 1, …, k) instances each. This measure is given by eq. (1) where d denotes the Euclidean distance. The lower S is the better the clustering is. (ii) One external measure, the purity P given by eq. (2), where k is the number of clusters, n is the total number of instances and n_i^j the number of instances in cluster i of class j. Purity can be interpreted as classification accuracy under the assumption that all the objects of a cluster are classified to be members of the dominant class for that cluster. Although the class was considered for obtaining purities, it was removed from the dataset to build the clusters.

$$S = \frac{1}{k}\sum_{i=1}^{k}\sqrt{\sum_{j=1}^{n_i}\sum_{l=j+1}^{n_i} d(\vec{x_j}, \vec{x_l})^2} \qquad (1) \qquad\qquad P = \frac{1}{n}\sum_{i=1}^{k}\max_j\left(n_i^j\right) \qquad (2)$$

Table 2 shows the patterns discovered by HDCC considering complete and single linkage. Each pattern is a 4-tuple where the component i is also a pattern that provides a description of attribute i.

Table 2. Patterns discovered by HDCC for three clusters

		Pattern
C1	**Single**	([4.3,5.8],[2.3,4.4],[1.0,1.9],[0.1,0.6])
	Complete	([4.3,5.8],[2.3,4.4],[1.0,1.9],[0.1,0.6])
C2	**Single**	([4.9,7.7],[2.0,3.6],[3.0,6.9],[1.0,2.5])
	Complete	([4.9,6.1],[2.0,3.0],[3.0,4.5],[1.0,1.7])
C3	**Single**	([7.7,7.9],[3.8,3.8],[6.4,6.7],[2.0,2.2])
	Complete	([5.6,7.9],[2.2,3.8],[4.3,6.9],[1.2,2.5])

In cluster C1 the dominant class was Iris Setosa, in C2 was Iris Versicolor and in C3 was Iris Virginica.

In fact, each of these patterns can be seen as a rule. For instance the discovered pattern for C1 under complete linkage and single linkage is ([4.3, 5.8], [2.3, 4.4], [1.0, 1.9], [0.1, 0.6]) that can be interpreted as the rule

```
(sepallength ≥ 4.3 AND sepallength ≤ 5.8 AND sepalwidth ≥ 2.3
AND sepalwidth ≤ 4.4 AND petallength ≥ 1.0 AND petallength ≤ 1.9
AND petalwidth ≥ 0.1 AND petalwidth ≤ 0.6)
```

where `sepallength`, `sepalwidth`, `petallength` and `petalwidth` are the 1st to 4th attributes in the dataset, respectively.

Table 3 shows the values of S and P for HDCC and the traditional hierarchical clustering algorithm under complete distance d^c_L and single linkage distance d^s_L for k = 3 that corresponds to the number of classes in the Iris dataset. As we can see the quality of the conceptual clustering does not differ from that of traditional hierarchical clustering even under single linkage and it provides useful descriptions that allow interpreting the meaning of each group of instances. This result, i.e. cluster quality preserved by HDCC, was confirmed by four experiments carried out on 100 artificial datasets each. Datasets were formed by 600 points drawn from 3 Gaussian distributions in $\Re^2$. In each of the four experiments, means and standard deviations were set to the values reported in Table 4. In these experiments the average values of S over the 100 experiments were obtained for HDCC and the traditional algorithm under single and complete linkage. These values are also reported in Table 4.

Table 3. Values of S and P for the traditional and conceptual dendrograms under d^c_L and d^s_L

Linkage distance	$S_{\text{Traditional}}$	$S_{\text{Conceptual}}$	$P_{\text{Traditional}}$	$P_{\text{Conceptual}}$
Single (d^s_L)	46.56	46.56	0.68	0.68
Complete (d^c_L)	37.44	37.44	0.84	0.84

Table 4. Values of S averaged over 100 experiments each for HDCC (Conc.) and the traditional hierarchical algorithm (Trad.) for 3 Gaussian distributions with (i) $\sigma = 1$ and $\mu \in [0, 10] \times [0, 10]$; (ii) $\sigma = 1$ and $\mu \in [0, 200] \times [0, 200]$; (iii) $\sigma = 5$ and $\mu \in [0, 100] \times [0, 100]$; (iv) $\sigma = 5$ and $\mu \in [0, 200] \times [0, 200]$

	Trad. (i)	**Conc. (i)**	**Trad. (ii)**	**Conc. (ii)**	**Trad. (iii)**	**Conc. (iii)**	**Trad. (iv)**	**Conc. (iv)**
d^s_L	524,820	514,417	282,605	282,605	1830,421	1851,406	1607,842	1595,194
d^c_L	285,622	285,622	282,605	282,605	1401,350	1401,350	1410,499	1410,499

4 Conclusions

Hierarchical distance-based conceptual clustering provides an integration of hierarchical distance-based clustering and conceptual clustering. It can be easily seen that for complex datatypes (sequences, graphs, etc.) the original dendrograms are usually different to the dendrograms obtained by applying the generalisation operators. In order to cope with these (negative) results, the notion of conceptual dendrogram and three consistency properties that should be analysed for every pair of distance and generalisation operator have been proposed. Some pairs of distances and generalisation operators are compatible at some degree resulting in equivalent, order-preserving or acceptable conceptual dendrograms while some other pairs are not, so showing that some distances and generalisation operators should not be used together.

In this work, however, we have shown a much more positive picture. In a propositional world, and using the most common distances and generalisation operators

for nominal data, numerical data and tuples, we have found out that the strongest properties (in fact all of them) hold. From these results, we can affirm that the integration of hierarchical distance-based clustering and conceptual clustering for propositional data (i.e., tables, which are still the bulk of most data mining applications) is feasible, congruent and relatively straightforward.

Additionally, the composability result obtained with the tuple datatype and several distances, allow the handling of more elaborate information in the form of tables, where some attributes can have structure, provided that the distance and generalisation operators used for every attribute have some degree of consistency.

In this regard, our immediate future work is focussed on finding operative pairs of distances and generalisation operators for common datatypes in data mining applications, such as sequences, graphs and multimedia objects.

References

1. Funes, A., Ferri, C., Hernández-Orallo, J., Ramirez-Quintana, M.J.: Hierarchical Distance-based Conceptual Clustering. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 349–364. Springer, Heidelberg (2008)
2. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Comput. Survey 31(3), 264–323 (1999)
3. Berkhin, P.: A Survey of Clustering Data Mining Techniques. In: Grouping Multidimensional Data, pp. 25–71. Springer, Heidelberg (2006)
4. Stanfill, A., Waltz, D.: Toward memory-based reasoning. Comm. of the ACM 29, 1213–1228 (1986)
5. Black, C.L., Merz, C.J.: UCI Repository of Machine Learning Databases (1998)
6. Estruch, V.: Bridging the gap between distance and generalisation: Symbolic learning in metric spaces. PhD thesis, DSIC-UPV (2008), http://www.dsic.upv.es/~vestruch/thesis.pdf
7. Fisher, D.: Knowledge acquisition via incremental conceptual clustering. In: Machine Learning, pp. 139–172 (1987)
8. Michalski, R.S.: Knowledge Acquisition Through Conceptual Clustering: A Theoretical Framework and an Algorithm for Partitioning Data into Conjunctive Concepts. Policy Analysis and Information Systems 4(3), 219–244 (1980)
9. Michalski, R.S., Stepp, R.E.: Learning from Observation: Conceptual Clustering. In: Michalski, et al. (eds.) Machine Learning: An Artificial Intelligence Approach, pp. 331–363. TIOGA Publishing Co. (1983)
10. Talavera, L., Béjar, J.: Generality-Based Conceptual Clustering with Probabilistic Concepts. IEEE Transactions on Pattern Analysis & Machine Intelligence 23(2) (2001)
11. Fisher, R.: The use of multiple measurements in taxonomic problems. Ann. Eurgenics, Part II 7, 179–188 (1936)
12. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proc. of the 5th Berkeley Sym. on Math. Statistics & Probability, pp. 281–297. Univ. of California Press (1967)
13. Cover, T.M., Hart, P.E.: Nearest neighbour pattern classification. IEEE Trans. Info. Theory IT-13, 21–27 (1967)
14. Funes, A.: Agrupamiento Conceptual Jerárquico Basado en Distancias, Definición e Instanciación para el Caso Proposicional. Master thesis. DSIC-UPV (2008), http://www.dsic.upv.es/~afunes/masterThesis.pdf

Computing Substitution Matrices for Genomic Comparative Analysis

Minh Duc Cao[1], Trevor I. Dix[1,2], and Lloyd Allison[1]

[1] Clayton School of Information Technology,
Monash University, Clayton 3800, Australia
[2] Faculty of Information & Communication Technologies,
Swinburne University of Technology, Hawthorn 3122, Australia
{minhduc,trevor,lloyd}@infotech.monash.edu.au

Abstract. Substitution matrices describe the rates of mutating one character in a biological sequence to another character, and are important for many knowledge discovery tasks such as phylogenetic analysis and sequence alignment. Computing substitution matrices for very long genomic sequences of divergent or even unrelated species requires sensitive algorithms that can take into account differences in composition of the sequences. We present a novel algorithm that addresses this by computing a nucleotide substitution matrix specifically for the two genomes being aligned. The method is founded on information theory and in the expectation maximisation framework. The algorithm iteratively uses compression to align the sequences and estimates the matrix from the alignment, and then applies the matrix to find a better alignment until convergence. Our method reconstructs, with high accuracy, the substitution matrix for synthesised data generated from a known matrix with introduced noise. The model is then successfully applied to real data for various malaria parasite genomes, which have differing phylogenetic distances and composition that lessens the effectiveness of standard statistical analysis techniques.

1 Introduction

Most important tools for mining in biological data such as sequence alignment and phylogenetics generally rely on a substitution matrix which ideally reflects the probability of mutating a character in a sequence to another in other sequences. Most sequence alignment algorithms attempt to find the optimal match of sequences where matching scores are derived from a substitution matrix [1,2]. It is well known that using a reliable substitution matrix significantly improves the sensitivity of sequence alignment and database search tools [3,4]. Substitution matrices also provide clues to dating of various evolutionary events and many molecular evolution mechanisms, and thus are often used in phylogenetic analysis [5,6].

Classically, a substitution matrix is empirically selected based on some assumptions about the sequences being analysed. For protein analysis, the PAM

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 647–655, 2009.
© Springer-Verlag Berlin Heidelberg 2009

substitution matrices [7] are calculated by observing the differences in related sequences with a certain ratio of substitution residues. The PAM-n matrix estimates what rate of substitution would be expected if $n\%$ of the amino acids had changed. On the other hand, the BLOSUMs [3] are derived from segments in a block with a sequence identity above a certain threshold.

While much research has been done on substitution matrices for protein, little attention has been paid to DNA substitution matrices despite the need of reliable tools for aligning genome size sequences of the next generation sequencing technology. Since not all DNA substitutions change the encoded amino acids, looking at the amino acid level only would lose some information. As more than one codons can code for the same amino acid, and different strains can show different preferences for codons that encode a given amino acid [8], a generic PAM or BLOSUM like substitution matrix for nucleotides such as RIBOSUM [9] can hardly work well on specific DNA sequences.

Work on DNA substitution matrices [10,11,12] often bases on a substitution model. Such examples of substitution models are the CJ69 model [13] which assumes all changes among four nucleotides occurring with equal probability and the K80 model [14] which allows transitions and transversions to occur with different rates. These models are rarely precise in practice. Traditional substitution matrix derivation methods also depend on sequence alignment, which in turn is plausible only when a reliable substitution matrix is used.

In this paper, we introduce a novel method to generate DNA substitution matrices for genomic comparative study. The method is based on information theory foundation [15] and is in expectation maximisation framework. Our method finds the substitution matrix directly from the data being analysed without having to make any assumptions. It considers the substitution matrix as parameters to align sequences and applies an expectation maximisation approach to estimate the parameters that optimise the alignment score. To the best of our knowledge, this method is the first to be able to compute specific substitution matrices for genome size sequences without any assumptions or a prior alignment of the data. The presented technique could be generalised to other types of data as well.

2 Methods

Information theory directly relates entropy to the transmission of a sequence under a statistical model of compression. Suppose a sequence X is to be efficiently transmitted over a reliable channel. The sender first compresses X using a compression model and transmits the encoded message to the receiver, who decodes the compressed stream using the same model to recover the original message. The *information content* $\mathcal{I}_X$ of X is the amount of information actually transmitted, i.e. the length of the encoded message.

Suppose a sequence Y related to X is available to both parties, the sender needs to transmit only the information of X that is not contained in Y. Since the receiver also knows Y, X can be recovered correctly. The amount of information actually transmitted in this case is called *conditional information content* of X

given Y, denoted as $\mathcal{I}_{X|Y}$. The more related the two sequences are, the more information the two sequences share, the shorter message is transmitted. The *mutual information* of X and Y is defined as the difference between the information content and the conditional information content: $\mathcal{I}_{X;Y} = \mathcal{I}_X - \mathcal{I}_{X|Y}$.

Compression of sequences requires a compression model. To measure the conditional information content of one sequence given another, the compression model needs to use a substitution matrix as parameters. In light of the *Minimum Message Length* principle [16,17], we propose an expectation-maximisation (EM) algorithm to find the substitution matrix that can produce the most compact encoding of the two sequences. In the E-step, one sequence is compressed on the background knowledge of the other to measure the conditional information content. The compression uses a substitution matrix, initialised to some default values, as the parameters to be estimated. We use the *expert model* [18] as the compression model because of its efficient performance and its ability to produce an local alignment of the two sequences [19]. In the M-step, the substitution matrix is then re-estimated based on the mutations observed from the local alignment. The EM process continues until the conditional information content obtained converges to an optimal value.

2.1 The Expert Model

The expert model algorithm [18] compresses a sequence X, symbol by symbol by forming the probability distribution of each symbol and then using a primary compression scheme to encode it. The probability distribution at a position is based on all symbols seen previously. Correspondingly, the decoder, having seen all previous decoded symbols, is able to compute the identical probability distribution and thus can recover the symbol. The information content of symbol x_i is computed as the negative log of the probability of the symbol [15]:

$$\mathcal{I}(i) = -logPr(x_i) \tag{1}$$

The algorithm maintains a set of *experts* to estimate the probability of a symbol. An expert is any entity that can provide a probability distribution of the symbol. An example is the *Markov expert* of order k which uses a Markov model learnt from the statistics of X to give the probability of a symbol given k preceding symbols. If a related sequence Y is available, the expert model employs *align experts* each of which considers the next symbol x_i in X to be part of a homologous region and align with a symbol y_j in Y. The align experts assume a substitution matrix P the entry $P(x, y)$ of which is the probability of substituting symbol y in Y by symbol x in X. The probability of symbol x_i predicted by an align expert is $Pr(x_i|y_j) = P(x_i, y_j)$. The expert model uses a hash table to propose align expert candidates. A hash table of hash size h suggests every matching h-mer as an align expert which is then evaluated and is discarded if it does not perform significantly better than the Markov expert.

The core part of the expert model is the combination of expert predictions. Suppose a panel of experts E is available to the encoder. Expert θ_k gives the

prediction $Pr(x_{m+1}|\theta_k, x_{1..m})$ of symbol x_{m+1} based on its observations of preceding m symbols. Expert predictions are combined based on Bayesian averaging:

$$\begin{aligned} Pr(x_{m+1}|x_{1..m}) &= \sum_{k\in E} Pr(x_{m+1}|\theta_k, x_{1..m}) w_{\theta_k,m} \\ &= \sum_{k\in E} Pr(x_{m+1}|\theta_k, x_{1..m}) Pr(\theta_k|x_{1..m}) \end{aligned} \tag{2}$$

The weight $w_{\theta_k,m}$ of expert θ_k for encoding x_{m+1} is assigned to $Pr(\theta_k|x_{1..m})$ and can be estimated by Bayes's theorem:

$$w_{\theta_k,m} = Pr(\theta_k|x_{1..m}) = \frac{\prod_{i=1}^{m} Pr(x_i|\theta_k, x_{1..i-1}) Pr(\theta_k)}{\prod_{i=1}^{m} Pr(x_i|x_{1..i-1})} \tag{3}$$

where $Pr(\theta_k)$ is the prior probability of expert θ_k before encoding any symbol. As Eq. 2 can be normalised to have $\sum Pr(x_{m+1}|x_{1..m}) = 1$, we can ignore the common denominator in Eq. 3 and take the negative log of the numerators:

$$-log_2(w_{\theta_k,m}) \propto -\sum_{i=1}^{m} log_2 Pr(x_i|\theta_k, x_{1..i-1}) - log_2 Pr(\theta_k) \tag{4}$$

Since $-log_2 Pr(x_i|\theta_k, x_{1..i-1})$ is the cost of encoding symbol x_i by expert θ_k, the right hand side of Eq. 4 represents the length of encoding subsequence $x_{1..m}$ by expert θ_k. As experts are evaluated on a recent history of w symbols, the message length of encoding $x_{m-w+1..m}$ is used to determine the weights of experts. Rewriting Eq. 4 for the weight of expert θ_k at position $m+1$ gives:

$$w_{\theta_k,m} \propto 2^{MsgLen(x_{m-w+1..m}|\theta_k) - log_2 Pr(\theta_k)} \tag{5}$$

Using only the Markov expert can produce the information content of sequence X. The conditional content of X given Y is obtained by combining the Markov expert with align experts. Align experts are first combined according to Eq. 5 to become the *blended align expert*, whose prediction is then combined with the Markov expert's prediction. The experts' weights specified in Eq. 5 involves the prior probability $Pr(\theta_k)$ of each expert. As all align experts are proposed by the same hash table, they have the same prior probability and hence the common factor $2^{-log_2 Pr(\theta_k)}$ can be ignored. However, for combination of the blended align expert and the Markov expert, their prior probabilities have to be specified. The prior probability of the blended align expert can be estimated from previous iterations of the EM process.

An align expert might be proposed simply by chance. The algorithm considers an align expert plausible if it performs significantly better than the Markov expert. It must encode the last w symbols better than the Markov expert by a threshold T bits, which is a parameter of the algorithm. When the align expert predicts beyond its homologous region, its performance worsens and it is discarded subsequently. Each align expert suggests an alignment of the region starting at the position it is proposed and ending at the position it is discarded. This region is called the *maximum-scoring segment pair* (MSP). The set of MSPs forms an local alignment of the two sequences.

2.2 Alignment Score and Mutual Information Content

Consider an align expert that uses a substitution matrix P and aligns x_i in X to y_j in Y. The alignment score is specified by the logarithm of the odds ratio of model H which assumes homology, and model R assuming random[20]:

$$S(x_i, y_j) = log_2 \frac{Pr(x_i, y_j|H)}{Pr(x_i, y_j|R)} = log_2 \frac{Pr(x_i, y_j|H)}{Pr(x_i)Pr(y_j)} \tag{6}$$

By Bayes's theorem, the numerator of the right hand side can be expressed as:

$$Pr(x_i, y_j|H) = Pr(x_i|y_j, H)Pr(y_j) = P(x_i, y_j)Pr(y_j) \tag{7}$$

Therefore,

$$S(x_i, y_j) = log_2 \frac{P(x_i, y_j)Pr(y_j)}{Pr(x_i)Pr(y_j)} = log_2 P(x_i, y_j) - log_2 Pr(x_i) \tag{8}$$

The alignment score of a MSP is the sum of alignment scores of all symbols in the region. If the MSP is from two regions starting at x_m and y_n respectively and is k symbols long, its alignment score is

$$S(x_m, y_n, k) = \sum_{i=0}^{k-1} -log_2 Pr(x_{m+i}) - \sum_{i=0}^{k-1} -log_2 Pr(x_{m+i}, y_{n+i}) \tag{9}$$

The two terms are the lengths of the compressed messages of the region $x_{m,k}$ by the Markov expert and by the align expert, respectively. In other words, the alignment score of a MSP is the mutual information content of the two regions.

2.3 Computing the Substitution Matrix

Once the local alignment of the two sequences is constructed, the substitution matrix is computed from the substitutions observed from the alignment. Entry $P(x, y)$ of the substitution matrix gets the value

$$P(x, y) = \frac{C_{x|y}}{C_y} \tag{10}$$

where $C_{x|y}$ is the number of symbol x in X that are aligned to symbol y in Y, and C_y is the number of symbol y in all MSPs.

A statistical hypothesis testing is performed to select the "good" MSPs to compute the substitution matrix. From Karlin-Altschul statistics [21], the E-value of occurrences of MSPs with a score S or greater is $E = KMN2^{-S}$ where M and N are the lengths of the two sequences and K is the Karlin-Altschul parameter. The occurrences of MSPs can be modelled by a Poisson process with characteristic parameter E. At the significance level $\alpha = 0.05$, the substitution matrix is estimated from mutations in MSPs having E-value $\leqslant \alpha$ or a score:

$$S \geqslant -log_2 \frac{\alpha}{KMN} \tag{11}$$

3 Experiment Results

We implemented the algorithm in Java and ran experiments on a PC with Intel Core 2 Duo 2.33Ghz CPU and 8GB of RAM, using Sun Java runtime environment 1.5. In our experiments, we used a hash table with hash key of 20 to propose align experts. The threshold T was set to 0.5 bits. The initial substitution matrix is set to have entries of 0.7 on the diagonal and 0.1 off the diagonal.

It is hard to verify substitution matrices derived from real data. We therefore performed experiments on a set of synthesised data so that the substitution matrix computed can be compared with the matrix used to generate data. The experiment is described in Subsection 3.1. We then ran experiments on a set of real data, as described in Subsection 3.2.

3.1 Experiment on Synthesised Data

Synthesised data was used to ensure the correct derivation of substitution matrices. The benefit of using artificial data is that the data can be generated with added noise from a known substitution matrix, and hence the computed matrix can be verified. We generated two "model genomes" each of which is one million bases long. About 10% of the first genome is "coding regions" which are copied to the second genome with substitution rates specified by a matrix P_{target}. The "non-coding regions" of the two genomes are independent on each other.

Table 1. The target and computed substitution matrices in the synthesised data experiment

$$P_{target} = \begin{vmatrix} .600 & .050 & .300 & .050 \\ .030 & .650 & .070 & .250 \\ .300 & .040 & .600 & .060 \\ .050 & .300 & .050 & .600 \end{vmatrix} \quad P_{computed} = \begin{vmatrix} .596 & .051 & .300 & .052 \\ .029 & .652 & .009 & .250 \\ .299 & .041 & .599 & .061 \\ .052 & .299 & .050 & .598 \end{vmatrix}$$

The substitution matrix is reconstructed from the data by aligning the second genome against the first one. After the fifth iteration the changes to the matrix between two consecutive iterations were negligible. In other words, the matrix converges after 5 iterations and in less than 10 minutes. Table 1 presents the target matrix P_{target} and the computed matrix $P_{computed}$ whose rows and columns are in $ACTG$ order. Given the noise introduced during the generation of the two sequences, the similarity of the computed matrix and the target matrix shows the effectiveness of our algorithm.

3.2 Experiment on Plasmodium Genomes

We analysed the genomes of four *Plasmodium* species, namely *P. falciparum, P. knowlesi, P. vivax* and *P. yoelii* which cause malaria in various hosts. The genomes are obtained from PlasmoDB release 5.4 (`http://www.plasmodb.`

Table 2. Plasmodium genomes characteristics

Species	Host	Genome Size (Mb)	%(AT) in Genome AT	%(AT) in CDS
P.falciparum	Human	23.2	80.63%	76.22%
P.vivax	Human	26.9	57.71%	53.70%
P.knowlesi	Monkey	23.4	60.79%	69.77%
P.yoelii	Rodent	20.1	77.36%	75.22%

org/common/downloads/release-5.4/). The nucleotide compositions in these species' genomes are very different. The AT content in the genome of *P. falciparum* is as high as 80% and in coding regions is 76.22% while the AT content in the *P. vivax* genome and *P. vivax* coding regions is 57.71% and 53.70% respectively. The characteristics of these genomes are presented in table 2.

Table 3. The substitution matrices of different malaria genomes

$$P_{Pf-Pk} = \begin{vmatrix} .701 & .074 & .144 & .081 \\ .107 & .707 & .054 & .131 \\ .184 & .066 & .642 & .108 \\ .089 & .156 & .075 & .680 \end{vmatrix} \quad P_{Pk-Pf} = \begin{vmatrix} .779 & .040 & .081 & .100 \\ .137 & .372 & .057 & .436 \\ .419 & .060 & .381 & .140 \\ .103 & .083 & .040 & .774 \end{vmatrix}$$

$$P_{Pf-Pv} = \begin{vmatrix} .613 & .086 & .227 & .074 \\ .085 & .705 & .077 & .133 \\ .146 & .084 & .687 & .083 \\ .073 & .233 & .086 & .608 \end{vmatrix} \quad P_{Pv-Pf} = \begin{vmatrix} .797 & .039 & .069 & .095 \\ .136 & .386 & .049 & .429 \\ .428 & .053 & .378 & .141 \\ .095 & .072 & .037 & .796 \end{vmatrix}$$

$$P_{Pf-Py} = \begin{vmatrix} .762 & .041 & .084 & .113 \\ .112 & .613 & .059 & .216 \\ .226 & .059 & .603 & .112 \\ .115 & .082 & .040 & .763 \end{vmatrix} \quad P_{Py-Pf} = \begin{vmatrix} .765 & .041 & .082 & .112 \\ .114 & .567 & .059 & .260 \\ .236 & .057 & .593 & .113 \\ .112 & .080 & .043 & .765 \end{vmatrix}$$

$$P_{Pk-Pv} = \begin{vmatrix} .741 & .063 & .145 & .051 \\ .060 & .754 & .076 & .110 \\ .101 & .072 & .757 & .060 \\ .052 & .142 & .065 & .741 \end{vmatrix} \quad P_{Pv-Pk} = \begin{vmatrix} .808 & .050 & .083 & .059 \\ .091 & .677 & .061 & .171 \\ .200 & .067 & .641 & .092 \\ .063 & .084 & .050 & .803 \end{vmatrix}$$

$$P_{Pk-Py} = \begin{vmatrix} .796 & .036 & .068 & .100 \\ .140 & .451 & .051 & .358 \\ .357 & .051 & .450 & .142 \\ .101 & .068 & .036 & .795 \end{vmatrix} \quad P_{Py-Pk} = \begin{vmatrix} .687 & .075 & .146 & .092 \\ .107 & .577 & .066 & .250 \\ .121 & .048 & .726 & .105 \\ .073 & .124 & .074 & .729 \end{vmatrix}$$

$$P_{Py-Pv} = \begin{vmatrix} .630 & .086 & .212 & .072 \\ .081 & .696 & .077 & .146 \\ .134 & .069 & .715 & .082 \\ .071 & .208 & .085 & .636 \end{vmatrix} \quad P_{Pv-Py} = \begin{vmatrix} .822 & .034 & .056 & .088 \\ .146 & .444 & .046 & .364 \\ .363 & .047 & .442 & .148 \\ .088 & .057 & .033 & .822 \end{vmatrix}$$

We applied our method to find the substitution matrix for each pair of these genomes. To compute the substitution matrix P_{Y-X} of genome Y to genome X, we compressed the genome X on the background knowledge of genome Y. Generally, about 4 or 5 iterations were required for convergence. The substitution matrices of these genomes are presented in Table 3.

4 Conclusions

We have presented a method for dynamically deriving a substitution matrix for analysis of any two long DNA sequences. The method is based on the sound theoretical foundation from information theory. We have shown that the method successfully regains the substitution matrix from synthesised data derived from a known matrix with introduced noise. The method has also been applied on real data with differing phylogenetic distances and nucleotide composition which would mislead classical statistical methods. Unlike traditional methods, our algorithm does not rely on the pre-alignment of sequences or on a substitution model. It incorporates the alignment of sequences and the substitution matrix computed in a expectation maximisation process. Furthermore, it can handle very long sequences in practical running time. The method therefore, would facilitate knowledge discovery in large and statistical biased databases.

References

1. Altschul, S.F., Madden, T., Schaffer, A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucl. Acids Res. 25(17), 3389–3402 (1997)
2. Kurtz, S., Phillippy, A., Delcher, A.L., Smoot, M., Shumway, M., Antonescu, C., Salzberg, S.: Versatile and open software for comparing large genomes. Genome. Biol. 5(2) (2004)
3. Henikoff, S., Henikoff, J.G.: Amino acid substitution matrices from protein blocks. Proc. Natl. Acad. Sci. 89(22), 10915–10919 (1992)
4. Altschul, S.F., Gish, W., Miller, W., Myers, E., Lipman, D.: Basic local alignment search tool. J. Mol. Biol. 215, 403–410 (1990)
5. Lio, P., Goldman, N.: Models of Molecular Evolution and Phylogeny. Genome. Res. 8(12), 1233–1244 (1998)
6. Felsenstein, J.: Evolutionary trees from DNA sequences: a maximum likelihood approach. J. Mol. Biol. 76(6), 368–376 (1981)
7. Dayhoff, M.O., Schwartz, R.M., Orcutt, B.C.: A model for evolutionary change in proteins. In: National Biochemical Research Foundation, Washington DC (1978)
8. Comeron, J.M., Aguade, M.: An evaluation of measures of synonymous codon usage bias. J. Mol. Biol. 47(3), 268–274 (1998)
9. Klein, R., Eddy, S.: Rsearch: Finding homologs of single structured RNA sequences. BMC Bioinformatics 4(1) (2003)
10. Goldman, N.: Statistical tests of models of DNA substitution. J. Mol. Evol. 36(2), 182–198 (1993)
11. Yang, Z.: Estimating the pattern of nucleotide substitution. J. Mol. Evol. 39(1), 105–111 (1994)
12. Yap, V.B., Speed, T.P.: Modeling dna base substitution in large genomic regions from two organisms. J. Mol. Evol. 58(1), 12–18 (2004)
13. Jukes, T.H., Cantor, C.: Evolution of protein molecules. Mammalian Protein Metabolism, 21–132 (1969)
14. Kimura, M.: A simple method for estimating evolutionary rate of base substitutions through comparative studies of nucleotide sequences. J. Mol. Evol. 16, 111–120 (1980)

15. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal 27, 379–423 (1948)
16. Wallace, C.S., Boulton, D.M.: An information measure for classification. Computer Journal 11(2), 185–194 (1968)
17. Wallace, C.S., Freeman, P.R.: Estimation and inference by compact coding. Journal of the Royal Statistical Society series 49(3), 240–265 (1987)
18. Cao, M.D., Dix, T.I., Allison, L., Mears, C.: A simple statistical algorithm for biological sequence compression. In: Data Compression Conference, pp. 43–52 (2007)
19. Cao, M.D., Dix, T.I., Allison, L.: A genome alignment algorithm based on compression. Technical Report 2009/233, FIT, Monash University (2009)
20. Altschul, S.F.: Amino acid substitution matrices from an information theoretic perspective. J. Mol. Biol. 219(3), 555–565 (1991)
21. Karlin, S., Altschul, S.F.: Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. Proc. Nat. Acad. Sci. 87(6), 2264–2268 (1990)

Mining Both Positive and Negative Impact-Oriented Sequential Rules from Transactional Data*

Yanchang Zhao[1], Huaifeng Zhang[1], Longbing Cao[1], Chengqi Zhang[1], and Hans Bohlscheid[1,2]

[1] Data Sciences and Knowledge Discovery Lab
Faculty of Engineering & IT, University of Technology, Sydney, Australia
{yczhao,hfzhang,lbcao,chengqi}@it.uts.edu.au
[2] Projects Section, Business Integrity Programs Branch, Centrelink, Australia
hans.bohlscheid@centrelink.gov.au

Abstract. Traditional sequential pattern mining deals with positive correlation between sequential patterns only, without considering negative relationship between them. In this paper, we present a notion of *impact-oriented negative sequential rules*, in which the left side is a positive sequential pattern or its negation, and the right side is a predefined outcome or its negation. *Impact-oriented negative sequential rules* are formally defined to show the impact of sequential patterns on the outcome, and an efficient algorithm is designed to discover both positive and negative impact-oriented sequential rules. Experimental results on both synthetic data and real-life data show the efficiency and effectiveness of the proposed technique.

Keywords: negative sequential rules, sequential pattern mining.

1 Introduction

Association rule mining [1] and sequential pattern mining [2] were proposed over a decade ago, and have been well developed and studied by many researchers. Traditional association rules and sequential patterns study only the co-occurrence of itemsets/events, that is, the positive relationship between itemsets/events. However, it is sometimes interesting to find negative correlation, such as two items are seldom bought together in a same basket, or one item is seldom bought after another item. Recently, a couple of techniques have been designed to find *negative association rules* [3,10,12]. However, *negative sequential patterns* are still seldom studied.

Previously we have introduced *event-oriented negative sequential rules* in the form of $P \rightarrow \neg e$, $\neg P \rightarrow e$ or $\neg P \rightarrow \neg e$, where P is a positive sequential pattern

* This work was supported by the Australian Research Council (ARC) Linkage Project LP0775041 and Discovery Projects DP0667060 & DP0773412, and by the Early Career Researcher Grant from University of Technology, Sydney, Australia.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 656–663, 2009.
© Springer-Verlag Berlin Heidelberg 2009

and e denotes a single event [14]. However, in many real-world applications, users are not interested in negative sequential rules associated with all possible events, but only those rules associated with a special target outcome, e.g., fraud or no fraud, debt or no debt, buy or not buy, etc. That is, the target event is the occurrence or non-occurrence of a specific outcome, instead of an arbitrary event. For example, for web click-stream analysis in online retail, an analyst may want to find the relationship between webpage visiting sequences and whether a user buys something. For a credit card company, it is interesting to discover the positive and negative relationship between transaction sequences and an unrecovered debt. In homeland security, the correlation between a series of activities and a terrorism attack is an important target of analysis.

To tackle the above problem, we develop in this paper an idea of *impact-oriented negative sequential rules*, where the left side is a traditional positive sequential pattern or its negation and the right side is a target outcome or its negation. A new efficient algorithm is designed for mining such rules and two novel metrics are defined to measure the impact on outcome.

2 Related Work

The technique of negative association rules has been well studied [3,10,12]. Negative association rules are defined in the form of $A \rightarrow \neg B$, $\neg A \rightarrow B$ and $\neg A \rightarrow \neg B$ [12]. Savasere et al. designed negative association rules as $A \not\rightarrow B$ [10]. Antonie and Zaïane defined *generalized negative association rule* as a rule containing a negation of an item, such as $A \wedge \neg B \wedge \neg C \wedge D \rightarrow E \wedge \neg F$, and defined *confined negative association rules* as $A \rightarrow \neg B$, $\neg A \rightarrow B$ and $\neg A \rightarrow \neg B$ [3].

The idea of sequential patterns was proposed in 1995 to find frequent sequential patterns in sequence data [2]. Some well-known algorithms for sequential pattern mining are AprioriALL [2], FreeSpan [6], PrefixSpan [9], SPADE [13] and SPAM (Sequential PAttern Mining) [4].

For sequential patterns, the non-occurrence of an element may also be interesting. For example, in social welfare, the lack of follow-up examination after the address change of a customer may result in overpayment to the customer. Such kind of sequences with the non-occurrence of elements are negative sequential patterns. However, most research on sequential patterns focus on positive patterns, and negative sequential patterns are underdeveloped. Some reported researches on negative sequential patterns are as follows. Sun et al. proposed negative event-oriented patterns [11] in the form of $\neg P \xrightarrow{T} e$, where e is a target event, P is a negative event-oriented pattern, and the occurrence of P is unexpectedly rare in T-sized intervals before target events. Bannai et al. proposed a method for finding optimal pairs of string patterns to discriminate between two sets of strings [5]. The pairs are in the forms of $p' \wedge q'$ and $p' \vee q'$, where p' is either p or $\neg p$, q' is either q or $\neg q$, and p and q are two substrings. Ouyang and Huang proposed negative sequences as $(A, \neg B)$, $(\neg A, B)$ and $(\neg A, \neg B)$ [8]. Lin et al. designed an algorithm NSPM (Negative Sequential Patterns Mining) for mining negative sequential patterns [7].

3 Mining Impact-Oriented Sequential Rules

3.1 Negative Sequential Rules

The negative relationships in transactional data are defined as follows.

Definition 1 (Negative Sequential Rules (NSR)). *A negative sequential rule is in the form of* $A \rightarrow \neg B$, $\neg A \rightarrow B$ *or* $\neg A \rightarrow \neg B$, *where* A *and* B *are positive sequential patterns composed of items in time order.*

Definition 2 (Event-oriented Negative Sequential Rules (ENSR)). *An event-oriented negative sequential rule is a special NSR, where the right side* B *is a single event, that is, the length of* B *is one.*

Definition 3 (Impact-oriented Negative Sequential Rules (INSR)). *An impact-oriented negative sequential rule is a special ENSR, where the right side is a predefined target outcome* T, *such as a specific class or a predetermined event.*

Definition 4 (Negative Sequential Patterns (NSP)). *A negative sequential pattern is a sequence of the occurrence or non-occurrence of items in time order, with at least one negation in it.*

Definition 5 (Generalized Negative Sequential Rules (GNSR)). *A generalized negative sequential rule is in the form of* $A \rightarrow B$, *where one or both of* A *and* B *are negative sequential patterns.*

Based on the above definitions, we can get: $I_{\text{GNSR}} \supset I_{\text{NSR}} \supset I_{\text{ENSR}} \supset I_{\text{INSR}}$, where I_{GNSR}, I_{NSR}, I_{ENSR} and I_{INSR} denotes respectively the sets of the above four kinds of rules. Although an INSR looks similar to an ENSR, the former is specially focused on a specific subset of ENSRs, so it demands more efficient techniques tailored for its special needs.

Traditional sequential rules are positive sequential rules, which are in the form of $A \rightarrow B$, where both A and B are positive sequential patterns. It means that pattern A is followed by pattern B. We refer to such positive rules as Type I sequential rules. By changing A or/and B to its/their negations, we can get the following three types of negative sequential rules:

- Type II: $A \rightarrow \neg B$, which means that pattern A is not followed by pattern B;
- Type III: $\neg A \rightarrow B$, which means that if pattern A does not appear, then pattern B will occur; and
- Type IV: $\neg A \rightarrow \neg B$, which means that if pattern A doesn't appear, then pattern B will not occur.

For types III and IV whose left sides are the negations of sequences, the meaning of the rules is: if A doesn't occur in a sequence, then B will (type III) or will not (type IV) occur in the sequence. That is to say, there is no time order between the left side and the right side. Note that A and B themselves are sequential

Table 1. Supports, confidences and lifts of four types of sequential rules

Type	Rules	Support	Confidence	Lift
I	$A \rightarrow B$	$P(AB)$	$\frac{P(AB)}{P(A)}$	$\frac{P(AB)}{P(A)P(B)}$
II	$A \rightarrow \neg B$	$P(A)-P(AB)$	$\frac{P(A)-P(AB)}{P(A)}$	$\frac{P(A)-P(AB)}{P(A)(1-P(B))}$
III	$\neg A \rightarrow B$	$P(B)-P(A\&B)$	$\frac{P(B)-P(A\&B)}{1-P(A)}$	$\frac{P(B)-P(A\&B)}{P(B)(1-P(A))}$
IV	$\neg A \rightarrow \neg B$	$1-P(A)-P(B)+P(A\&B)$	$\frac{1-P(A)-P(B)+P(A\&B)}{1-P(A)}$	$\frac{1-P(A)-P(B)+P(A\&B)}{(1-P(A))(1-P(B))}$

patterns, which makes them different from negative association rules. However, if time constraint is considered in sequential rules, the last two types of rules may have new meanings, which is out of the scope of this paper. The supports, confidences and lifts of the above four types of sequential rules are shown in Table 1. In the table, $P(A\&B)$ denotes the probability of the concurrence of A and B in a sequence, no matter which one occurs first, or whether they are interwoven.

3.2 Algorithm for Mining Impact-Oriented Sequential Rules

To discover impact-oriented negative sequential rules, we use SPAM (Sequential PAttern Mining) [4] as a start point, because it was demonstrated by Ayres et al. to be more efficient than SPADE and Prefixpan [4], another two well-known algorithms for sequential pattern mining. SPAM is very efficient in that it uses bitmap to count the frequency of sequences. It searches the sequence lattice in a depth-first way, and candidates of longer sequences S_g are generated by append frequent items $\{i\}$ to existing frequent sequences S_a. The candidate generation of SPAM is composed of two steps: S-step and I-step. The S-step appends i to S_a, which builds a longer sequence $S_g = S_a \bowtie i$. The I-step adds i to the last itemset of S_a, which builds a new sequence of the same length as S_a. In this paper, we consider transaction with one item only and an element in the sequence is a single item, instead of an itemset. Therefore, only S-step from SPAM is used in our technique.

Figure 1 gives the pseudocode for finding impact-oriented negative sequential rules, which is based on the function "FindSequentialPatterns" from SPAM [4]. Lines 2-17 show the code for appending the target outcome to a sequential pattern and computing the chi-square and direction for the derived sequential rule. Lines 2-6 use bitmaps to compute the counts, support, confidence and lift for the sequential rule. Lines 7-17 compute the observed frequencies and expected frequencies, and then calculate chi-square and direction. Lines 19-23 generate positive sequential patterns. Lines 25-32 are the S-step of SPAM, which tries to extend the sequential pattern at current node by appending an additional item to it. Lines 34-43 generate three types of negative sequential patterns.

3.3 New Metrics for Impact-Oriented Sequential Rules

Two new metrics, *contribution* and *impact*, are designed as follows to select interesting impact-oriented sequential rules.

```
ALGORITHM: FindINSR - a recursive call that goes down the lattice to find INSR
INPUT: curNode: information about the current node
OUTPUT: impact-oriented negative sequential rules
 1: /* Assume that n is the number of customers. */
 2: cntA = curNode → count; cntT=targetEventCount;
 3: bitmapAT =SequentialAnd(bitmapA, bitmapT); cntAT = bitmapAT → Count();
 4: bitmapAorT =Or(bitmapA, bitmapT); cntAorT = bitmapAorT → Count();
 5: cntAandT = cntA + cntT − cntAorT;
 6: supp = cntAT/n; conf = cntAT/cntA; lift = cntAT∗n/(cntA∗cntT);
 7: /* observed frequencies of AT, A¬T, ¬AT and ¬A¬T */
 8: f1 = cntAT; f2 = cntA − cntAT;
 9: f3 = cntT − cntAT; f4 = n − cntA − cntT + cntAT;
10: /* expected frequencies of AT, A¬T, ¬AT and ¬A¬T*/
11: ef1 = cntA ∗ cntT/n; ef2 = cntA ∗ (1 − cntT/n);
12: ef3 = (1 − cntA/n) ∗ cntT; ef4 = (1 − cntA/n) ∗ (n − cntT);
13: chiSquare = Σ (fi−efi)^2/efi;
14: IF chiSquare < 3.84 /* 95% confidence to reject the independence assumption */
15:     direction = 0;
16: ELSE IF lift > 1 THEN direction = +1, ELSE direction = −1;
17: END IF
18: IF cntAT ≥ minsupp ∗ n
19:     /* generating positive sequential patterns */
20:     IF direction = +1
21:         compute supp, conf and lift for Type I rule based on Table 1;
22:         output "A → T" when supp ≥ minsupp, conf ≥ minconf and lift ≥ minlift;
23:     END IF
24:     /* s-step*/
25:     FOR each possible s-extension i from this level
26:         tempAndBitmap=Bit-Wise-And(bitmap of curNode, bitmap of i);
27:         cntAB=tempAndBitmap.count; /* corresponding to P(AB) */
28:         IF cntAB ≥ minsupp ∗ n
29:             add i to nextNode's s-extension list;
30:             FindINSR(nextNode); /* checking the node at next level */
31:         END IF
32:     END FOR
33: ELSE
34:     /* generating negative sequential patterns */
35:     IF direction = −1
36:         compute supp, conf and lift for Type II rule based on Table 1;
37:         output "A → ¬T" if supp ≥ minsupp, conf ≥ minconf and lift ≥ minlift;
38:         compute supp, conf and lift for Type III rule based on Table 1;
39:         output "¬A → T" if supp ≥ minsupp, conf ≥ minconf and lift ≥ minlift;
40:     ELSE IF direction = +1
41:         compute supp, conf and lift for Type IV rule based on Table 1;
42:         output "¬A → ¬T" if supp ≥ minsupp, conf ≥ minconf and lift ≥ minlift;
43:     END IF
44: END IF
```

Fig. 1. Pseudocode for discovering impact-oriented negative sequential rules

Definition 6 (Contribution). *For a sequential rule $P \rightarrow T$, where P is a sequential pattern, assume i to be the last item in P. The contribution of i to the occurrence of outcome T in rule $P \rightarrow T$ is*

$$contribution(i, P) = \frac{lift(P \rightarrow T)}{lift(P \setminus i \rightarrow T)} \tag{1}$$

where $P \setminus i$ denotes the sequential pattern derived by removing i from P.

Definition 7 (Impact). *For the above rule and i, the impact of i on the outcome in the rule is*

$$impact(i, P) = \begin{cases} contribution(i, P) - 1 : if\ contribution \geq 1, \\ \frac{1}{contribution(i,P)} - 1 \quad : otherwise. \end{cases} \tag{2}$$

Contribution shows how much the last item i in the rule contributes to the occurrence of the outcome T, and impact measures how much it can change the outcome. Both of them fall in $[0, +\infty)$.

4 Experimental Results

4.1 Performance and Scalability

Our designed algorithm (referred to as INSR) was implemented with C++ based on SPAM [4], and its performance and scalability was tested on synthetic datasets generated with IBM data generator [2]. All the tests were conducted on a PC with Intel Core 2 CPU of 1.86GHz, 2GB memory and Windows XP Pro. SP2. The number of items per transaction was set to one when generating data.

Our algorithm was first tested on a dataset with 50,000 customers, 40 items per sequence and the length of maximal patterns as 13. The minimum supports range from 0.2 to 0.7, and the results are shown in Figure 2a. From the figure, both INSR and Spam [4] run faster with larger minimum support, because the search space becomes smaller. Moreover, INSR runs faster than Spam, and the reason is that, when a pattern A is frequent and $A \bowtie T$ is infrequent, INSR doesn't search A's children nodes, but Spam continues checking all its descendants until it becomes infrequent.

The scalability with the number of sequences was tested on datasets with average sequence length as 30, length of maximal patterns as 11. The number of customers ranges from 10,000 to 100,000, and the support threshold is set to 0.3. Figure 2b shows the result of the above test. It's clear from the figure that INSR is linear with the number of sequences.

The running time with varying sequence lengths is shown Figure 2c, where the datasets used have 50,000 customers, with length of maximal patterns as 10, and the average sequence length ranging from 10 to 45. The support threshold is set to 0.3. The figure shows that the running time becomes longer with the increase of the average number of items per sequence and that INSR is almost linear with the length of sequences.

4.2 Selected Results in a Case Study

The proposed technique was applied to the real data from Centrelink, Australia. Centrelink is a Commonwealth Government agency distributing social welfare payments to entitled customers. For various reasons, customers on benefit payments or allowances sometimes get overpaid and these overpayments lead to debts owed to Centrelink. We used impact-oriented negative sequential rules to find the relationship between transactional activity sequences and debt occurrences, and also find the impact of additional activities on debt occurrence.

A sample of historical transactional data from July 2007 to February 2008 were used for the analysis. After data preprocessing, 15,931 sequences were constructed. Minimum support was set to 0.05, that is, 797 out of 15,931 sequences. There are 2,173,691 patterns generated and the longest pattern has 16 activities. Some selected sequential rules are given in Table 2, where "DEB" stands for debt and the other codes are activities. "Direction" shows whether the pattern is positively (+1) or negatively (-1) associated with debt occurrence.

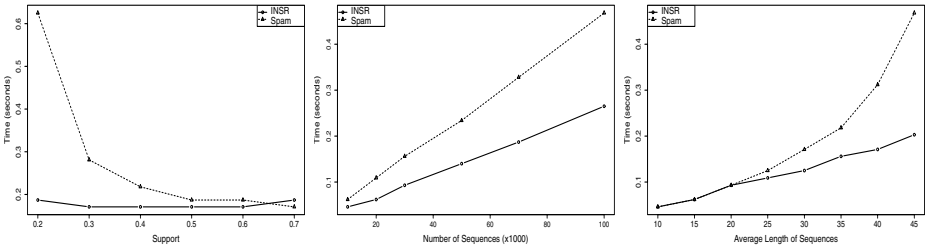


Fig. 2. Scalability with (a) support; (b) the number of sequences; and (c) the length of sequences (from left to right)

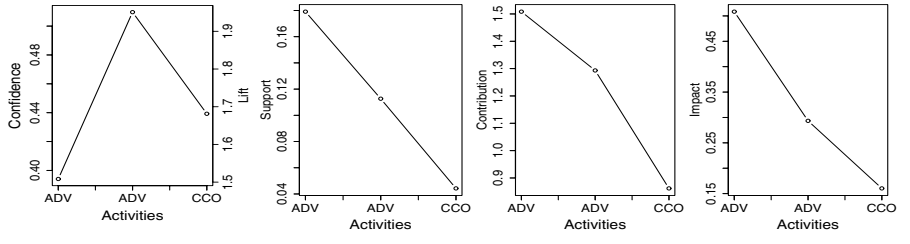


Fig. 3. A growing sequential pattern "ADV ADV CCO"

Table 2. Selected positive and negative sequential rules

Type	Rule	Supp	Conf	Lift	Direction
I	REA ADV ADV→DEB	0.103	0.53	2.02	+1
I	RPR ANO→DEB	0.111	0.33	1.25	+1
I	STM PYI→DEB	0.106	0.30	1.16	+1
II	MND→ ¬DEB	0.116	0.85	1.15	-1
II	REA PYR RPR RPT→ ¬DEB	0.176	0.84	1.14	-1
II	REA CRT DLY→ ¬DEB	0.091	0.83	1.12	-1
III	¬{PYR RPR REA STM}→DEB	0.169	0.33	1.26	-1
III	¬{PYR CCO}→DEB	0.165	0.32	1.24	-1
III	¬{PLN RPT}→DEB	0.212	0.28	1.08	-1
IV	¬{REA EAN}→ ¬DEB	0.650	0.79	1.07	+1
IV	¬{DOC FRV}→ ¬DEB	0.677	0.78	1.06	+1

Figure 3 shows an example of discovered growing sequential pattern,

$$\begin{cases} ADV \rightarrow DEB \\ ADV, ADV \rightarrow DEB \\ ADV, ADV, CCO \rightarrow DEB \end{cases}. \tag{3}$$

Each point in every chart gives the value for the sequential pattern from the first activity to the corresponding activity. All four charts in Figure 3 show the growth from "ADV" to "ADV ADV" and "ADV ADV CCO". ADV increases the probability of debt occurrence, because its confidence in debt occurrence is 0.395, 1.5 times the likelihood of debt occurrence in the whole population (see the first chart). There are 18% of all sequences supporting that ADV is followed by debt (see the second chart). As shown in the third chart, the two ADVs contributes to debt occurrence, but CCO contributes negatively, as its contribution is less than one. The impacts of two ADVs on outcome are different, with the first one having larger impact (see the fourth chart).

5 Conclusions

We have defined impact-oriented negative sequential rules and have designed an efficient algorithm for mining such sequential rules. We have also designed two metrics, contribution and impact, to measure the effect of an item on the outcome, which help to select interesting growing sequential patterns. A case study has been presented to show the effectiveness of the proposed technique.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proc. of the ACM SIGMOD Int. Conf. on Management of Data, Washington D.C., USA, May 1993, pp. 207–216 (1993)
2. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Yu, P.S., Chen, A.S.P. (eds.) Proc. of the 11th Int. Conf. on Data Engineering, Taipei, Taiwan, pp. 3–14 (1995)
3. Antonie, M.-L., Zaïane, O.R.: Mining positive and negative association rules: an approach for confined rules. In: Proc. of the 8th Eur. Conf. on Principles and Practice of Knowledge Discovery in Databases, New York, USA, pp. 27–38 (2004)
4. Ayres, J., Flannick, J., Gehrke, J., Yiu, T.: Sequential pattern mining using a bitmap representation. In: KDD 2002: Proc. of the 8th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, New York, NY, USA, pp. 429–435 (2002)
5. Bannai, H., Hyyro, H., Shinohara, A., Takeda, M., Nakai, K., Miyano, S.: Finding optimal pairs of patterns. In: Jonassen, I., Kim, J. (eds.) WABI 2004. LNCS (LNBI), vol. 3240, pp. 450–462. Springer, Heidelberg (2004)
6. Han, J., Pei, J., et al.: Freespan: frequent pattern-projected sequential pattern mining. In: KDD 2000: Proc. of the 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, New York, NY, USA, pp. 355–359 (2000)
7. Lin, N.P., Chen, H.-J., Hao, W.-H.: Mining negative sequential patterns. In: Proc. of the 6th WSEAS Int. Conf. on Applied Computer Science, Hangzhou, China, pp. 654–658 (2007)
8. Ouyang, W.-M., Huang, Q.-H.: Mining negative sequential patterns in transaction databases. In: Proc. of 2007 Int. Conf. on Machine Learning and Cybernetics, Hong Kong, China, pp. 830–834 (2007)
9. Pei, J., Han, J., et al.: Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In: ICDE 2001: Proc. of the 17th Int. Conf. on Data Engineering, Washington, DC, USA, p. 215 (2001)
10. Savasere, A., Omiecinski, E., Navathe, S.B.: Mining for strong negative associations in a large database of customer transactions. In: ICDE 1998: Proc. of the 14th Int. Conf. on Data Engineering, Washington, DC, USA, pp. 494–502 (1998)
11. Sun, X., Orlowska, M.E., Li, X.: Finding negative event-oriented patterns in long temporal sequences. In: Proc. of the 8th Pacific-Asia Conf. on Knowledge Discovery and Data Mining, Sydney, Australia, pp. 212–221 (May 2004)
12. Wu, X., Zhang, C., Zhang, S.: Efficient mining of both positive and negative association rules. ACM Transactions on Information Systems 22(3), 381–405 (2004)
13. Zaki, M.J.: Spade: An efficient algorithm for mining frequent sequences. Machine Learning 42(1-2), 31–60 (2001)
14. Zhao, Y., Zhang, H., Cao, L., Zhang, C., Bohlscheid, H.: Efficient mining of event-oriented negative sequential rules. In: Proc. of the 2008 IEEE/WIC/ACM Int. Conf. on Web Intelligence (WI 2008), Sydney, Australia, pp. 336–342 (December 2008)

Aggregated Subset Mining

Albrecht Zimmermann and Björn Bringmann

Department of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200A, 3001 Leuven, Belgium
{Albrecht.Zimmermann,Bjorn.Bringmann}@cs.kuleuven.be

Abstract. The usual data mining setting uses the full amount of data to derive patterns for different purposes. Taking cues from machine learning techniques, we explore ways to divide the data into subsets, mine patterns on them and use post-processing techniques for acquiring the result set. Using the patterns as features for a classification task to evaluate their quality, we compare the different subset compositions, and selection techniques. The two main results – that small independent sets are better suited than large amounts of data, and that uninformed selection techniques perform well – can to a certain degree be explained by quantitative characteristics of the derived pattern sets.

1 Introduction

Data mining essentially comes in two flavors, *descriptive* mining, finding descriptions of the data, and *predictive* mining, constructing features for effective classification. In predictive mining, class-correlating patterns, patterns showing strong correlation with a class value, are often a good choice. No matter the reliability of statistic measures, however, they can still fall prey to over-fitting, which in turn may harm classifiers. This becomes even more problematic if the amount of patterns is large, or pairs and combinations of patterns reinforce each other's bias.

In a recent work [3], the effect of decreasing redundancy between patterns on the accuracy of classifiers using those particular features was evaluated. While we could show that reducing redundancy – in some cases rather strongly – did in fact improve accuracy, we used the entire data set set for mining patterns which we then filtered. This setting, which is the *standard* data mining setting, is well-suited for descriptive mining. Predictive mining is closer related to machine learning, however, which knows different techniques using parts of the labeled data for verification purposes of found patterns/built classifiers.

We therefore take a page out of the playbook of ML, first mining several sets of correlating patterns, and then using different criteria to create final result sets from them. These are used as features for learning an SVM [5] classifier.

The paper is structured as follows: In the next section, we explain the basic mechanisms for mining patterns, creating subsets of the data for mining and selection purposes, and lay out several selection methods for deriving the final result set. In Section 3, we report on the experimental evaluation of the proposed methods before concluding in Section 4.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 664–672, 2009.
© Springer-Verlag Berlin Heidelberg 2009

2 Mining and Merging Correlating Patterns

We start from set of instances $\mathcal{D}_m$ each being labeled with one of the class labels $\{pos, neg\}$. In this set $\mathcal{D}_m$ we search for patterns drawn from a language $\mathcal{L}$. More specifically for a set of k patterns whose occurrence on the instances correlates best with the presence of the target class according to χ^2 [4]. Further we require the found patterns to be *free* according to [1].

The solutions to the mining task can then be conveniently modeled using

$$\mathcal{T}h_k(\mathcal{D}_m) = \{p \in \mathcal{L} \mid p \text{ among the k-best free patterns on } \mathcal{D}_m \text{ w.r.t. } \chi^2\}$$

As said before, this is the *standard* data mining setting which operates on the full dataset $\mathcal{D}_m$, which we will use as a base-line technique. In the following sections we propose different methods for selecting the final pattern set.

2.1 Using a Validation Set

The most basic approach consist of using a certain fraction q of the total data $\mathcal{D}_m$ as the actual mining set $\overline{\mathcal{D}}_m$, with size $q \cdot |\mathcal{D}_m|$. The rest would be used as a validation set $\hat{\mathcal{D}}_m = \mathcal{D}_m \setminus \overline{\mathcal{D}}_m$, of size $(1-q) \cdot |\mathcal{D}_m|$. After termination of the mining process on $\overline{\mathcal{D}}_m$, the k_m patterns $\mathcal{T}h_{k_m}(\overline{\mathcal{D}}_m)$ returned by the miner are evaluated on $\hat{\mathcal{D}}_m$ and re-ranked, according to their correlation score χ^2 achieved on this validation set. Out of those the k_s best scoring patterns are returned to the user. It can easily be derived that k_s should be chosen such that $k_s < k_m$ since for $k_m \leq k_s$ the validation scores (and re-ranking) have no effect on the selection of patterns. The final result set is then

$$val_{k_s}(\mathcal{T}h_{k_m}(\overline{\mathcal{D}}_m), \hat{\mathcal{D}}_m) = \{p \in \mathcal{T}h_{k_m}(\overline{\mathcal{D}}_m) \mid p \text{ in the } k_s\text{-best patterns on } \hat{\mathcal{D}}_m \text{ w.r.t. } \chi^2\}$$

Given the use of statistically significant patterns, one would expect a certain robustness against statistical quirks. The degree to which the full distribution can be modeled by a subset could however very well be governed by q. A not unusual choice for q in the machine learning literature is $\frac{2}{3}$.

2.2 Aggregating Subset Results

In the second approach, subsets $\mathcal{D}_m^i$ of $\mathcal{D}_m$ are created, and the top k_m patterns mined from each of them. For the union of their results $\Phi_{all} = \bigcup_i \mathcal{T}h_{k_m}(\mathcal{D}_m^i)$ we know that $|\Phi_{all}| \geq k_m$. All patterns $p \in \Phi_{all}$ are re-evaluated according to some aggregation metric, and a subset (e.g. the top-k_m patterns) returned to the user.

This approach is illustrated in Figure 1, with merge denoting the merging/re-evaluation step. What should be immediately obvious from this figure is that this kind of approach lends itself to distributed/parallel mining, although the merging step needs to be performed on one particular site.

There are two main decisions that influence the result of this approach, namely the choice of subsets and the aggregation metric used. The size of the final set to be returned is obviously also important but has less effect than the aforementioned two choices, we believe.

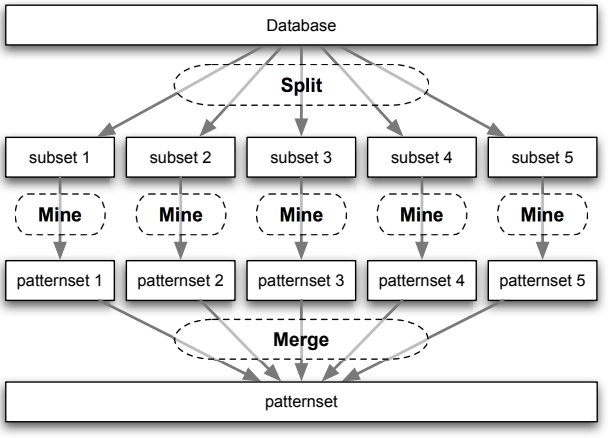


Fig. 1. The aggregated subset mining technique

Formation of subsets. We investigate two approaches to forming subsets of $\mathcal{D}_m$. Their main difference lies in whether there is overlap among subsets used or not. The straightest-forward approach consists of segmenting $\mathcal{D}_m$ into f disjunct folds $\mathcal{F}_i$. We define $\hat{\mathcal{D}}^i_m = F_i$ and $\overline{\mathcal{D}}^i_m = \bigcup_{j \neq i} F_j$. In both cases *all* instances in the data have an effect on the final result with the same weight.

Aggregation metrics. The goal of any aggregation metric used lies in ranking the patterns in Φ_{all} by using information from all subsets $\mathcal{D}^i_m$ mined on. To this end, we propose three metrics:

1. A first measure takes the form $\mu_{count}(p) = |\{i \ : \ p \in \mathcal{T}h_{k_m}(\mathcal{D}^i_m)\}|$
   Basically μ_{count} counts for each pattern p in how many of the $\mathcal{D}^i_m$ it was found among the top-k_m. This measure only checks whether a pattern was mined at all, however, not what its particular rank was in the respective result sets.
2. Hence, a second metric, μ_{rank} would consist of the following:

$$\mu_{rank}(p) = \frac{1}{f} \sum_i rank(p, \mathcal{D}^i_m)$$

   with

$$rank(p, \mathcal{D}) = \begin{cases} 1 + k_m - \inf\{k \mid p \in \mathcal{T}h_k(\mathcal{D})\} & \textit{if} \ \ p \in \mathcal{T}h_{k_m}(\mathcal{D}) \\ 0 \ \textit{otherwise} \end{cases}$$

   taking a pattern's "local" quality into account.
3. A final metric, μ_{χ^2} would take the form: $\mu_{\chi^2}(p) = (\chi^2$ of p on $\mathcal{D}_m)$ calculating the score for each pattern according to χ^2 on the entire set $\mathcal{D}_m$. This metric is related to the validation approach of 2.1, with the difference that here the data on which the pattern was mined is also used for validation.

Selection criteria. Due to the fact that $|\Phi_{all}| \geq k_m$, one can simply return the top-k_m after the re-ranking via one of the metrics $\mu(p)$. Thus, given a value k_m, a metric μ, and a set of subsets $\mathcal{M} = \{\mathcal{D}^i_m\}$, our goal is to select $\varphi_{k_m}(\mathcal{M}, \mu) \subseteq \Phi_{all}$ such that the $p_i \in \varphi_{k_m}(\mathcal{M}, \mu)$ are the k_m highest ranked pattern in Φ_{all} w.r.t. μ.

Additionally, this framework does allow for a second k-value (k_s), similar to the one of the validation set approach which is used to define the size of the final result set leading to $\Phi_{all} = \bigcup_i \mathcal{T}h_{k_m}(\mathcal{D}^i_m)$ with $\varphi_{k_s}(\mathcal{M}, \mu) \subseteq \Phi_{all}$.

3 Experimental Evaluation

For the experimental evaluation, we arbitrarily picked 8 data sets from the NCI-60 data set collection [6] and mine sequential patterns on them. Each has about 3500 instances (one outlier having only 2778) with a class distribution of $50-53\%$ (another outlier of 63.7%) for the positive class. We chose $k_m \in \{10, 25, 50, 75, 100\}$, giving a reasonable range of values across which to compare. We evaluated two aspects of the outlined techniques experimentally:

Q1 **Quantitative Analysis:** The effect of different *subset formation* strategies and subset sizes. Specifically, we consider similarity between the pattern sets finally selected and the ones mined in the *standard* setting.

Q2 **Qualitative Analysis:** The effect of different *aggregation* methods on the quality of the pattern sets selected.

To get a robust accuracy estimate, a 10-fold cross-validation was performed. All folds – both for accuracy and selection purposes – were stratified. As mentioned above, an SVM classifier was used for accuracy estimates. SVMs possess certain inherent feature selection capabilities, giving low-relevance features small weights. In addition, an SVM attempts to find a separating hyperplane with a maximal margin to both classes. Both of these characteristics guard against overfitting at the classifier level, allowing to evaluate the quality of the *feature set*. The SVM's C parameter was tuned via a 5-fold cross-validation on the training data, with potential values $2^i, i \in [-2, 14]$.

3.1 Validating Patterns on Additional Data

In the first setting, using a validation set for assessing found patterns' quality, we evaluated two stratified random splits of the mining data, with $q = \frac{2}{3}$ and $q = \frac{4}{5}$, respectively. As to the top-k parameters, $k_m, k_s \in \{10, 25, 50, 75, 100\} : k_m > k_s$.

Quantitative results. A useful measure for assessing quantitative characteristics is that of overlap. Given two sets of patterns $\mathbb{S}, \mathbb{S}'$, we simply define $ovlp(\mathbb{S}, \mathbb{S}') = |\mathbb{S} \cap \mathbb{S}'|$. For evaluating the quantitative characteristics of selected pattern sets, we calculate the overlap with the standard mining operation (denoted by *standard* in Table 1): $ovlp(val_{k_s}(\mathcal{T}h_{k_m}(\overline{\mathcal{D}}_m), \hat{\mathcal{D}}_m), \mathcal{T}h_{k_s}(\mathcal{D}_m))$, and with the *non-validated* pattern set: $ovlp(val_{k_s}(\mathcal{T}h_{k_m}(\overline{\mathcal{D}}_m), \hat{\mathcal{D}}_m), \mathcal{T}h_{k_s}(\overline{\mathcal{D}}_m))$ (denoted by *non-validated*).

Table 1. Overlap between the validated sets and the non-validated/standard setting, respectively

non-validated $q = \frac{2}{3}$	standard $q = \frac{2}{3}$	k_s/k_m	non-validated $q = \frac{4}{5}$	standard $q = \frac{4}{5}$
3.1 ±1.1005	2.7 ±1.05935	10/25	3.6 ±1.26491	3.7 ±0.823273
0.7 ±0.948683	0.5 ±0.971825	10/50	1.2 ±1.31656	1.4 ±1.17379
0.2 ±0.421637	0.2 ±0.421637	10/75	0.6 ±0.966092	0.7 ±0.948683
0.1 ±0.316228	0 ±0	10/100	0.3 ±0.674949	0.3 ±0.674949
11.1 ±1.96921	10 ±2	25/50	11 ±2.16025	10.9 ±2.55821
6.3 ±1.82878	6.1 ±2.33095	25/75	6.3 ±2.11082	6.3 ±1.88856
4.3 ±0.948683	4 ±1.56347	25/100	4 ±1.33333	4.1 ±1.59513
32.5 ±1.71594	28.5 ±2.4608	50/75	32.6 ±1.7127	30.8 ±1.68655
24 ±1.69967	21.6 ±1.57762	50/100	22.9 ±2.37814	22.5 ±2.83823
56.8 ±1.8738	48.3 ±3.56059	75/100	56.3 ±1.41814	53.4 ±1.77639

The pattern set overlap values show that the greater k_m for a given k_s, the smaller the overlap between pattern sets becomes. This means that patterns are ranked rather differently on the validation set although similar underlying distributions should be expected. Overlap is of course higher for comparison against the non-validated set since the validated set is constructed from this. It is interesting however, that the difference between comparison to the standard and the non-validated setting are not that great. Furthermore, there is no big change between the results for $q = \frac{2}{3}$ and $q = \frac{4}{5}$.

Qualitative results. Regarding Q2, we use the selected features to encode $\mathcal{D}_m$ as binary vectors and evaluate the SVM's performance. The main focus of our comparison lies on determining which q is better suited to the mining of "good" features, and whether there are particularly well-suited $k_m - k_s$ combinations.

We report the results on a representative data set in Table 2. Unfortunately, the answer seems to be that neither q is a good choice. Using a validation set selects features that are less well suited for classification than mining on the full data. This indicates that randomly splitting the data can give rise to so radically different distributions (hinted at in the quantitative analysis) that top-k selections based on χ^2 becomes meaningless.

Table 2. Predictive accuracies of the validation settings and the standard setting (top row)

k_s	10	25	50	75
Standard setting (full $\mathcal{D}_m$)				
$k_m = k_s$	59.752 ± 1.974	60.949 ± 2.361	62.205 ± 3.168	64.859 ± 2.479
Validation setting $q = 0.66$				
k_m=25	54.507 ± 1.358	–	–	–
k_m=50	54.65 ± 2.590	55.134 ± 2.389	–	–
k_m=75	54.565 ± 1.605	55.704 ± 1.690	59.152 ± 3.848	–
k_m=100	52.997 ± 0.768	56.732 ± 2.492	56.017 ± 4.070	57.617 ± 5.830
Validation setting $q = 0.80$				
k_m=25	53.139 ± 1.172	–	–	–
k_m=50	51.972 ± 0.895	55.163 ± 2.890	–	–
k_m=75	52.256 ± 0.061	54.166 ± 2.515	57.928 ± 3.547	–
k_m=100	52.227 ± 0.140	53.452 ± 2.214	58.015 ± 2.450	61.293 ± 2.243

3.2 Aggregated Pattern Selection

For the second setting –using different subsets to mine the data and using aggregation metrics– we chose $f \in \{3, 5, 7\}$, thus allowing for different sizes of $\mathcal{D}_m$. In addition to the standard setting, we compare to a post-processing method which uniformly picks k_s patterns from Φ_{all} at random. Since this method does not use *explicit* information on patterns' quality, nor their relationship, we use it as a baseline to see whether the better informed methods enjoy an advantage.

Quantitative results. Regarding Q1 and given similar results for all data sets, we report quantitative characteristics of Φ_{all} in Tables 3 and 4 on one example. For both alternatives regarding construction of the $\mathcal{D}_m$ we list the minimum and maximum μ_{count} and μ_{rank} for patterns in Φ_{all}, $|\Phi_{all}|/k_m$, and $ovlp(\varphi_{k_m}(\mathcal{M}, \mu_\sigma), \mathcal{T}h_{k_m}(\bigcup \mathcal{M}))$. We would expect that:

- $ovlp_{\overline{\mathcal{D}}_m} \geq ovlp_{\hat{\mathcal{D}}_m}$ – Larger $\mathcal{D}_m$ give similar results as the standard setting
- $|\Phi_{all,\hat{\mathcal{D}}_m}|/k_m \gg |\Phi_{all,\overline{\mathcal{D}}_m}|/k_m$ – Smaller $\mathcal{D}_m$ give a larger variety of patterns
- $\min_{p \in \Phi_{all}} \mu_{count}(p) > 1$ – No pattern appears in only one result set
- $\max_{p \in \Phi_{all}} \mu_{count}(p) \approx f$ – The best patterns generalize over most $\mathcal{D}_m$
- $\min_{p \in \Phi_{all}} \mu_{rank}(p) > 1/f$ – No pattern is always ranked worst
- $\max_{p \in \Phi_{all}} \mu_{rank}(p) \approx k_m$ – The best patterns generalize over most $\mathcal{D}_m$, appearing with a high ranking

The evaluation shows that most of our expectations hold, the only serious exceptions being our assumptions about the "worst" patterns – which often appear in only one $\mathcal{T}h_k(\mathcal{D}_m)$. This indicates that even when using correlation measures, different data sets quickly lead to differing mining results. It is interesting to see that *overlap*, $|\Phi_{all}|/k_m$, and $\max_{p \in \Phi_{all}} \mu_{rank}(p)$ are rather stable for the $\overline{\mathcal{D}}_m$ setting for a given k_m, but depend on the value of f for the $\hat{\mathcal{D}}_m$ setting.

Table 3. Quantitative characteristics for pattern sets mined on $\hat{\mathcal{D}}_m^i$

f	Overlap	$\lvert\Phi_{all}\rvert/k_m$	$\max_{p \in \Phi_{all}} \mu_{count}(p)$	$\max_{p \in \Phi_{all}} \mu_{rank}(p)$	$\min_{p \in \Phi_{all}} \mu_{rank}(p)$
			$k_m = 10$		
3	3.5 ±1.178	2.000 ± 0.200	3 ± 0.000	7.800 ± 1.033	0.333 ± 0.000
5	1.1 ±0.567	3.260 ± 0.302	4.2 ± 0.632	6.260 ± 0.766	0.200 ± 0.000
7	0.3 ±0.483	4.630 ± 0.434	4.7 ± 0.483	4.814 ± 0.919	0.143 ± 0.000
			$k_m = 25$		
3	8.7 ±2.213	2.136 ± 0.163	3 ± 0.000	22.800 ± 1.033	0.333 ± 0.000
5	4.7 ±1.494	3.428 ± 0.204	4.8 ± 0.422	19.420 ± 1.459	0.200 ± 0.000
7	3.7 ±1.159	4.668 ± 0.305	5.8 ± 0.632	15.614 ± 1.355	0.143 ± 0.000
			$k_m = 50$		
3	19.6 ±2.756	2.114 ± 0.131	3 ± 0.000	47.800 ± 1.033	0.333 ± 0.000
5	11.3 ±1.702	3.290 ± 0.208	5 ± 0.000	43.820 ± 2.165	0.200 ± 0.000
7	8.9 ±1.370	4.454 ± 0.367	6.6 ± 0.699	36.700 ± 2.203	0.143 ± 0.000
			$k_m = 75$		
3	32 ±3.126	2.056 ± 0.123	3 ± 0.000	72.800 ± 1.033	0.367 ± 0.105
5	19.7 ±2.496	3.167 ± 0.171	5 ± 0.000	68.760 ± 2.299	0.200 ± 0.000
7	15 ±1.763	4.389 ± 0.283	6.7 ± 0.675	60.171 ± 4.035	0.143 ± 0.000
			$k_m = 100$		
3	43.3 ±5.375	2.018 ± 0.128	3 ± 0.000	97.800 ± 1.033	0.333 ± 0.000
5	26.7 ±2.945	3.114 ± 0.171	5 ± 0.000	93.760 ± 2.299	0.200 ± 0.000
7	20.5 ±1.715	4.321 ± 0.234	6.8 ± 0.422	84.071 ± 5.767	0.143 ± 0.000

Qualitative results. Given the findings above, the more interesting question is which of the proposed techniques select patterns which are useful features for classification. Again, we used an SVM and 10-fold cross-validation to estimate the quality of pattern sets. Inasmuch as differences in accuracy were almost never significant, we omit the actual accuracy estimates here. Instead we report how the different methods (each time a combination of $\mathcal{D}_m$ composition and selection method) compare given a fixed $k_m \in \{10, 25, 50, 75, 100\}$ and $f \in \{3, 5, 7\}$ (Table 5). Note that the table shows the total number of wins for each approach.

Each number denotes how often a particular technique has performed better than any other on any data set. We evaluated 9 techniques against each other on 8 data sets. Thus any given approach can have maximally 64 wins. Bold values denote the best-performing technique, given a k_m and value of f, while a circle ($\circ$) shows for which k_m a technique performed best, given f.

The first, somewhat surprising, insight is that using large, overlapping $\mathcal{D}_m$, which should recreate phenomena over different mining situations, does not lead to good pattern selection. $\overline{\mathcal{D}}_m$ settings never perform best for a given k_m and usually perform better if only relatively few patterns are selected, suggesting that resampling does too little to counteract bias. Given that resampling forms the basis for, e.g., BAGGING [2] techniques, we did not expect this outcome.

It is also noticeable that the standard approach produces suboptimal pattern sets. Only once is this baseline approach best, for $f = 3$, meaning relatively large folds where *informed* selection techniques such as *count* and *rank* do not enjoy a large advantage. Even there it is closely followed by the random selection - essentially the least informed one. This means that an unwritten paradigm of data mining (using large amounts of data to the fullest leads to meaningful patterns) turns out to be questionable in this case.

The random technique is the big winner of the entire comparison, given its simplicity. While reducing redundancy entirely by chance, it performs well in 4

Table 4. Quantitative characteristics for pattern sets mined on $\overline{\mathcal{D}}_m^i$

f	Overlap	$\lvert\Phi_{all}\rvert/k_m$	$\max_{p\in\Phi_{all}} \mu_{count}(p)$	$\max_{p\in\Phi_{all}} \mu_{rank}(p)$	$\min_{p\in\Phi_{all}} \mu_{rank}(p)$
			$k_m = 10$		
3	6.6 ±1.074	1.440 ± 0.150	3 ± 0.000	9.033 ± 0.508	0.367 ± 0.105
5	6.5 ±0.849	1.430 ± 0.125	5 ± 0.000	9.420 ± 0.416	0.260 ± 0.135
7	6.6 ±1.429	1.400 ± 0.176	7 ± 0.000	9.500 ± 0.318	0.157 ± 0.045
			$k_m = 25$		
3	13.6 ±1.429	1.644 ± 0.125	3 ± 0.000	24.033 ± 0.508	0.333 ± 0.000
5	13.6 ±1.264	1.660 ± 0.080	5 ± 0.000	24.420 ± 0.416	0.220 ± 0.063
7	12.7 ±1.766	1.684 ± 0.115	7 ± 0.000	24.500 ± 0.318	0.157 ± 0.045
			$k_m = 50$		
3	28.1 ±2.424	1.648 ± 0.088	3 ± 0.000	49.033 ± 0.508	0.333 ± 0.000
5	29.1 ±1.286	1.632 ± 0.067	5 ± 0.000	49.420 ± 0.416	0.200 ± 0.000
7	29.8 ±1.549	1.578 ± 0.066	7 ± 0.000	49.500 ± 0.318	0.171 ± 0.090
			$k_m = 75$		
3	45.6 ±1.837	1.599 ± 0.082	3 ± 0.000	74.033 ± 0.508	0.433 ± 0.161
5	47 ±2.494	1.545 ± 0.068	5 ± 0.000	74.420 ± 0.416	0.280 ± 0.103
7	47.3 ±2.830	1.512 ± 0.073	7 ± 0.000	74.500 ± 0.318	0.157 ± 0.045
			$k_m = 100$		
3	61.8 ±3.521	1.535 ± 0.067	3 ± 0.000	99.033 ± 0.508	0.333 ± 0.000
5	64.3 ±3.128	1.480 ± 0.064	5 ± 0.000	99.420 ± 0.416	0.320 ± 0.140
7	65.6 ±3.893	1.456 ± 0.082	7 ± 0.000	99.500 ± 0.318	0.157 ± 0.045

Table 5. Total accuracy wins for aggregation techniques and the baseline approach for combinations of different k_m-values and f settings

f	3					5					7				
k_s	10	25	50	75	100	10	25	50	75	100	10	25	50	75	100
$\hat{\mathcal{D}}_m$, chi	4	0	11 ∘	3	4	5	1	14 ∘	1	4	4	1	13 ∘	3	1
$\hat{\mathcal{D}}_m$, random	41	41	44	43	53 ∘	53	**56**	50	**59** ∘	47	44	50	**54**	**58** ∘	51
$\hat{\mathcal{D}}_m$, rank	**51** ∘	43	39	**49**	34	**54**	51	**52**	52	**61** ∘	44	**54** ∘	52	52	**52**
$\hat{\mathcal{D}}_m$, top	44	**52** ∘	**49**	40	35	52 ∘	44	47	45	37	**49** ∘	44	38	39	36
$\overline{\mathcal{D}}_m$, chi	8	8	7	6	13 ∘	9	7	17 ∘	7	11	7	7	10 ∘	7	10
$\overline{\mathcal{D}}_m$, random	28	50 ∘	44	43	34	34	44 ∘	34	38	30	25	48 ∘	44	38	41
$\overline{\mathcal{D}}_m$, rank	38	26	42 ∘	34	32	29	24	26	35	36 ∘	40 ∘	32	33	30	36
$\overline{\mathcal{D}}_m$ top	43 ∘	42	26	40	27	23	33 ∘	28	26	24	35 ∘	29	22	32	23
baseline	31	26	26	30	**56** ∘	29	28	20	25	38 ∘	40 ∘	23	22	29	38

of 15 settings. It is outperformed by *rank* (7 wins), but *count* is weaker (3 wins). Moreover, adding up *all* wins by technique, *random* outperforms *rank* and *count*, slightly for $\hat{\mathcal{D}}_m$ settings, more pronounced for $\overline{\mathcal{D}}_m$. So the information which patterns generalize well over different subsets does not give a strong advantage in our case study. However, the variety of patterns caused by several subsets is helpful. Re-evaluating patterns' χ^2 score does again not work satisfactory.

4 Conclusions

In this work, we investigated ways of using data for pattern mining to produce good features for classification of complex data. Two main insights arise from the experimental evaluation: 1) usual assumptions on how to best use data in data mining turn out to be questionable. Neither the standard data mining setting (using large data sets to smooth over-fitting effects), nor a single mining and validation set, nor re-sampling techniques producing overlapping mining sets to uncover true underlying phenomena proved to be the most effective use. The best usage we observed consisted of splitting data into small, *independent* subsets instead, mining patterns on these and evaluating those patterns' generalization capability on *different* subsets. 2) the actual selection method matters far less than could be expected. Given a large enough variety of patterns, picking patterns at random proved to be rather effective, as proved the average rank selector, which picks patterns that were highly ranked at least once, even if not in *all* subsets. Using a validation set (either independent or involving the data patterns were mined from) for reassessing the χ^2-score did not work satisfactory.

An unexpected boon of these results is that pattern mining can apparently be easily parallelised without having to fear the loss of valuable information in terms of patterns. Quite contrary, we have seen that merging pattern sets extracted from small independent data sets improves the merit of the found patterns.

There are still several open questions to pursue w.r.t. the evaluated techniques. As we have observed, the interplay between k_m and k_v for the validation set technique has an effect on the composition of resulting pattern sets, and different k_m seem to favor certain aggregation techniques. It would therefore

be valuable to perform stability studies, e.g. investigating whether final pattern sets stabilize for a certain value of k_m. Additionally, there are potential further selection criteria which time and space constraints did not allow us to investigate.

References

1. Boulicaut, J., Jeudy, B.: Mining free itemsets under constraints. In: Adiba, M.E., Collet, C., Desai, B.C. (eds.) IDEAS, pp. 322–329 (2001)
2. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
3. Bringmann, B., Zimmermann, A.: The chosen few: On identifying valuable patterns. In: ICDM, pp. 63–72. IEEE Computer Society, Los Alamitos (2007)
4. Bringmann, B., Zimmermann, A., De Raedt, L., Nijssen, S.: Don't be afraid of simpler patterns. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS, vol. 4213, pp. 55–66. Springer, Heidelberg (2006)
5. Joachims, T.: Making large-scale support vector machine learning practical. In: Advances in kernel methods: support vector learning, pp. 169–184. MIT Press, Cambridge (1999)
6. Swamidass, S.J., Chen, J.H., Bruand, J., Phung, P., Ralaivola, L., Baldi, P.: Kernels for small molecules and the prediction of mutagenicity, toxicity and anti-cancer activity. In: ISMB (Supplement of Bioinformatics), pp. 359–368 (2005)

Hot Item Detection in Uncertain Data

Thomas Bernecker, Hans-Peter Kriegel, Matthias Renz, and Andreas Zuefle

Institute for Informatics, Ludwig-Maximilians-Universität München, Germany
{bernecker,kriegel,renz,zuefle}@dbs.ifi.lmu.de

Abstract. An object o of a database $\mathcal{D}$ is called a *hot item*, if there is a sufficiently large population of other objects in $\mathcal{D}$ that are similar to o. In other words, *hot items* are objects within a dense region of other objects and provide a basis for many density-based data mining techniques. Intuitively, objects that share their attribute values with a lot of other objects could be potentially interesting as they show a typical occurrence of objects in the database. Also, there are a lot of application domains, e.g. sensor databases, traffic management or recognition systems, where objects have vague and uncertain attributes. We propose an approach for the detection of potentially interesting objects (*hot items*) of an uncertain database in a probabilistic way. An efficient algorithm is presented which detects *hot items*, where to each object o a confidence value is assigned that reflects the likelihood that o is a *hot item*. In an experimental evaluation we show that our method can compute the results very efficiently compared to its competitors.

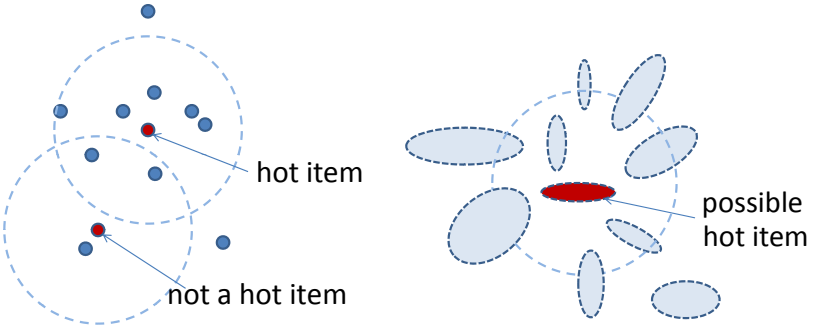
1 Introduction

The detection of objects which build dense regions with other objects within a feature space is a foundation of several density-based data mining techniques, in particular density-based clustering [8], outlier detection and other density-based mining applications [11,13]. We call an object o, for which exists a sufficiently large population of other objects in $\mathcal{D}$ that are similar to o, a *hot item*. Intuitively, an item that shares its attributes with a lot of other items could be potentially interesting as its shows a typical occurrence of items in the database. Application areas where the detection of *hot items* is potentially important exemplarily include scientific applications, e.g. astrophysics, biomedical, sociological and economic applications.

The applications mentioned above require special methods supporting the efficient search in modern databases that may contain not-standard data. Modern databases have to cope with uncertain or imprecise data. Example applications are location determination and proximity detection of moving objects, similarity search and pattern matching in sensor databases or personal identification and recognition systems based on video images or scanned image data. Several approaches that cope with uncertain objects have been proposed [6,9,14,15]. The proposed methods mainly address efficient solutions for similarity search on uncertain data including probabilistic distance range, k-nearest neighbor and ranking. To the best of our knowledge there does not exist any approach addressing retrieval of *hot items* in uncertain domains.

A *hot item* o has the property that the number of other items (objects) which are in the proximity of o, i.e. are similar to o, exceed a given minimal population value. In this

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 673–680, 2009.
© Springer-Verlag Berlin Heidelberg 2009



(a) Hot items in certain data. (b) Hot items in uncertain data.

Fig. 1. Examples of hot items

paper, we give a more general definition of hot items by relaxing the distance/similarity predicate between the objects.

Definition 1 (Hot Item). *Given a database $\mathcal{D}$ with objects and a minimum population threshold min_items. Furthermore, we assume a score function $d_{score} : \mathcal{D} \times \mathcal{D} \rightarrow \mathbb{R}_0^+$ which is defined on pairs of objects in $\mathcal{D}$ and a predicate $\Phi_\varepsilon : \mathbb{R}_0^+ \rightarrow \{true, false\}$, where $\Phi_\varepsilon \in \{<\varepsilon, \leq\varepsilon, =\varepsilon, \geq\varepsilon, >\varepsilon\}$ and $\varepsilon \in \mathbb{R}_0^+$ is a given scalar. An object $o \in \mathcal{D}$ is called* hot item*, iff there exist at least min_items objects $o' \in \mathcal{D}\backslash\{o\}$ which fulfill the predicate Φ_ε, formally*

$$|\{o' \in \mathcal{D}\backslash\{o\} : \Phi_\varepsilon(d_{score}(o, o')) = true\}| \geq min_items \Leftrightarrow o \text{ is a } \textbf{\textit{hot item}}.$$

In the case of uncertain objects, an exact score cannot be determined, particularly if the score relates to the object attributes which are assumed to be uncertain. Consequently, uncertain objects lead to uncertain scores which in turn lead to uncertain predicate results. Thus, the result of the predicate Φ_ε is no longer binary and instead yields a probability value. This probabilistic predicate result can be estimated. Based on this estimation we are able to compute for each object o of an uncertain database a probability value which reflects the likelihood that o is a hot item or not. A formal definition of probabilistic hot item detection is given later in Section 3. The solution for the efficient computation of hot item probabilities can be found in Section 4.

2 Related Work

In the context of this paper, hot items can be abstracted to objects that fulfill a given predicate together with a reasonably large set of other items. If we assume the *equality* predicate, i.e. $\Phi_\varepsilon(d_{score}) := d_{score} = 0$, then a hot item satisfies the *frequent item* property. The detection of frequent items or frequent itemsets as a preprocessing step for rule mining is one of the most important problems in data mining. Chui et al. study in [7]

the problem of mining frequent itemsets from uncertain data. They assume transactions whose items are associated with existential probabilities and introduce the U-Apriori algorithm, which is a modified version of the Apriori algorithm. They present a framework which probabilistically computes frequent items in an efficient way.

The aspect to identify objects that are similar to a given amount of other objects is the basis of several density-based algorithms for discovering clusters and outliers. There exist approaches for density-based clustering of uncertain data, e.g. [10] which are quite related to our approach. However the proposed model used to determine the probabilistic density does not respect the mutual exclusiveness of alternative attribute values. The missing conditional probability in their approach leads to approximative results only which disqualifies this approach from the accurate detection of hot items.

A lot of work has been published for managing uncertain data [4,5,14], probabilistic similarity queries [9] and quite recently for probabilistic top-k queries [12,15]. The detection of hot items can be efficiently supported by a similarity join query used in a preprocessing step, in particular the distance range self-join. Approaches for an efficient join on uncertain data are proposed in [9]. The main advantage of this approach is that sampled positions in space can efficiently be indexed using traditional spatial access methods thus allowing to reduce the computational complexity of complex query types. Our approach exploits the similarity join approach proposed in [9]. However, the cost of the probabilistic detection of hot items are originally highly CPU-bound which is demonstrated in our experimental evaluation (cf. Section 5). The advantage of an I/O cost efficient approach for the preprocessing step only becomes noticeable when applying the methods proposed in this paper such that the CPU cost less outbalance the overall query cost.

3 Problem Definition

In this section, we formally introduce the problem of probabilistic identification of hot items in uncertain databases.

3.1 Probabilistic Score

The identification whether an object is a hot item or not requires to know the neighborhood of the object according to a given (similarity) distance score function. Assuming that the object attributes the score function relates to are uncertain, then the score result is uncertain, too. Therefore, we require a probabilistic score function which is defined as follows: Let $P_{\Phi_\varepsilon} : \mathcal{D} \times \mathcal{D} \rightarrow [0,1]$ be a probabilistic function defined on a pair of objects that returns the likelihood that a given score w.r.t. both objects fulfills a given predicate Φ_ε. For example, if we use the distance $d(o_i, o_j)$ between two uncertain vector objects o_i and o_j as score function and we use the predicate $\Phi_\varepsilon = d(o_i, o_j) \leq \varepsilon$, then $P_{\Phi_\varepsilon}(o_i, o_j)$ denotes the probability that o_j is within the ε-range of o_i and vice versa.

3.2 Probabilistic Hot Items

Based on the definitions given above, we can compute hot items in uncertain data in a probabilistic way. However, we have to solve the problem of dependencies of the

uncertain attributes. Though we assume that the attributes of uncertain objects are independent of each other, we have to respect that the values of an uncertain object attribute are mutually exclusive. For this reason, first we have to define probabilistic hot items based on a conditional probability.

Definition 2 (Conditional Probabilistic Hot Item). *Given a database $\mathcal{D}$ with uncertain objects and a minimum population threshold min_items. Furthermore, we assume a predicate $\Phi_\varepsilon : \mathbb{R}_0^+ \rightarrow \{true, false\}$ which is defined on a probabilistic score function, where $\Phi_\varepsilon \in \{<\varepsilon, \leq\varepsilon, =\varepsilon, \geq\varepsilon, >\varepsilon\}$ and $\varepsilon \in \mathbb{R}_0^+$ is a given scalar. Under the condition that an uncertain object $o \in \mathcal{D}$ is equal to a certain vector $x \in \mathbb{R}^d$, the probability that o is a* hot item *can be computed by*

$$P(o \text{ is a } \textbf{\textit{hot item}} | o = x) =$$

$$P(|\{o' \in \mathcal{D}\backslash\{o\} : \Phi_\varepsilon(d_{score}(o, o')) = true\}| \geq min_items) =$$

$$\sum_{\substack{S_{min_items} \subseteq \mathcal{D}\backslash\{o\} \\ |S_{min_items}| \geq min_items}} \Big(\prod_{o' \in S_{min_items}} P_{\Phi_\varepsilon}(o, o') \cdot \prod_{o' \in \mathcal{D}\backslash(S_{min_items} \cup \{o\})} (1 - P_{\Phi_\varepsilon}(o, o')) \Big).$$

The above definition gives rise to the following general definition of probabilistic hot items which depends on the used uncertainty model. The probability P(o is a **hot item**) of an object o being an (unconditionally) probabilistic hot item can be computed by aggregating the conditional hot item probabilities over all possible instances x of o multiplied with the probability that object o corresponds to x, i.e.

$$\sum_{x \in \mathbb{R}^d} P(o = x) \cdot P(|\{o' \in \mathcal{D}\backslash\{o\} : \Phi_\varepsilon(d_{score}((x, 1), o')) = \text{true}\}| \geq min_items).$$

4 Hot Item Detection Algorithm

Let $\mathcal{D}$ be a database with uncertain objects. Each object $o \in \mathcal{D}$ is probed w.r.t. the hot item property. This computation can be split into the preprocessing step which finds candidates that match the predicate Φ_ε and the query step which detects the *hot items*.

4.1 Preprocessing Step

First, for each object $o' \in (\mathcal{D}\backslash\{o\})$ we have to compute the probability that o' fulfills a given predicate $\Phi_\varepsilon \in \{<\varepsilon, \leq\varepsilon, =\varepsilon, \geq\varepsilon, >\varepsilon\}$ w.r.t. object o, i.e. we have to compute $P_{\Phi_\varepsilon}(o, o')$. Obviously, only those objects $o' \in \mathcal{D}' \subset \mathcal{D}$ for which the predicate Φ_ε is fulfilled with a probability greater than zero, i.e. $P_{\Phi_\varepsilon}(o, o') > 0$, have to be taken into account in order to compute the probability $P(o \text{ is a hot item})$. Note that, depending on the used predicate Φ_ε, usually only a small portion $\mathcal{D}' \subset \mathcal{D}$ of the database fulfills the predicate $\Phi_\varepsilon(o, o' \in \mathcal{D}')$ with a probability greater than zero. A quick search of those objects which have to be taken into account can be efficiently supported by means of an index structure, e.g. the R*-tree. In particular for the predicate $\Phi_\varepsilon = \text{“} \leq \varepsilon\text{”}$, the index supported ε-range join [2] can be used to speed-up the search as proposed in [3].

Here, approximative representations like the minimal bounding rectangle (mbr) of an uncertain object are very appropriate to be used as index key for a filter step following the multi-step query processing paradigm. A solution for the ε-range join on uncertain data is proposed in [9] which can be used as a preprocessing step for our proposed algorithm for the detection of hot items.

4.2 Query Step

In the following, we introduce our new approach which is able to efficiently compute the probability that an object $o \in \mathcal{D}$ is a hot item. As mentioned above, our algorithm has quadratic runtime or even needs linear time if min_items is assumed to be constant. The key idea of our approach is based on the following property. Given a set of j predicates $\mathcal{S} = \{p_1, p_2, \ldots, p_j\}$ for which the probability $P(p_i)$ that the predicate $p_i \in \mathcal{S}$ is "true" is known, respectively. Now, we want to compute the probability $P_{k,\mathcal{S}}$ that at least k predicates of $\mathcal{S}$ are "true".

Lemma 1. *If we assume that predicate p_j is "true", then $P_{k,\mathcal{S}}$ is equal to the probability that at least $k-1$ predicates of $\mathcal{S}\backslash\{p_j\}$ are "true". Otherwise, $P_{k,\mathcal{S}}$ is equal to the probability that at least k predicates of $\mathcal{S}\backslash\{p_j\}$ are "true".*

The above lemma leads to the following recursion that allows to compute $P_{k,\mathcal{S}}$ by means of the paradigm of dynamic programming:

$$P_{k,\mathcal{S}} = P_{k-1,\mathcal{S}\backslash\{p_j\}} \cdot p_j + P_{k,\mathcal{S}\backslash\{p_j\}} \cdot (1 - p_j),$$

where

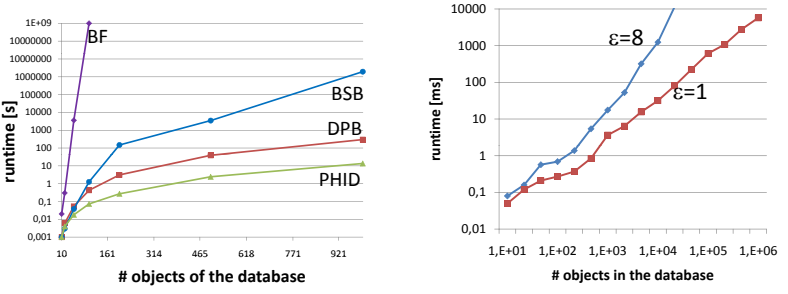
$$P_{0,\mathcal{S}} = 1.$$

The above dynamic programming scheme is an adaption of a technique previously used in the context of probabilistic top-k queries [15]. Here, we generalize this technique for arbitrary probabilistic predicates. We apply this method to compute the probability that an uncertain object $o \in \mathcal{D}$ is a hot item. Given an uncertain object $o \in \mathcal{D}$, the value for min_items and the set $\mathcal{D}' \subseteq \mathcal{D}$ of objects for which the probability that the predicate $P_{\Phi_\varepsilon}(o', o)$ ($o' \in \mathcal{D}'$) is "true" is greater than zero, i.e. $\forall o' \in \mathcal{D}' : P_{\Phi_\varepsilon}(o, o') > 0$. The probability P that object o is a hot item is equal to the probability $P_{min_items,\mathcal{D}'}(o)$ that for at least min_items objects $o' \in \mathcal{D}'$ the predicates $\Phi_\varepsilon(o, o')$ are "true". With Lemma 1 and the dynamic programming technique described above we can compute $P = P_{min_items,\mathcal{D}'}(o)$ efficiently by:

$$P_{min_items,\mathcal{D}'}(o) = \begin{cases} P_{min_items-1,\mathcal{D}'\backslash\{o'\}}(o) \cdot P_{\Phi_\varepsilon}(o, o') + \\ P_{min_items,\mathcal{D}'\backslash\{o'\}}(o) \cdot (1 - P_{\Phi_\varepsilon}(o, o')) & \text{, if } min_items > 0 \\ 1 & \text{, if } min_items = 0. \end{cases}$$

5 Experimental Evaluation

In this section, we present the results of an experimental evaluation of the proposed methods w.r.t. efficiency. First we specify the used datasets and experimental setup. In the artificial *ART* dataset, each object is represented by a set of positions sampled from



(a) Evaluation of competing techniques.

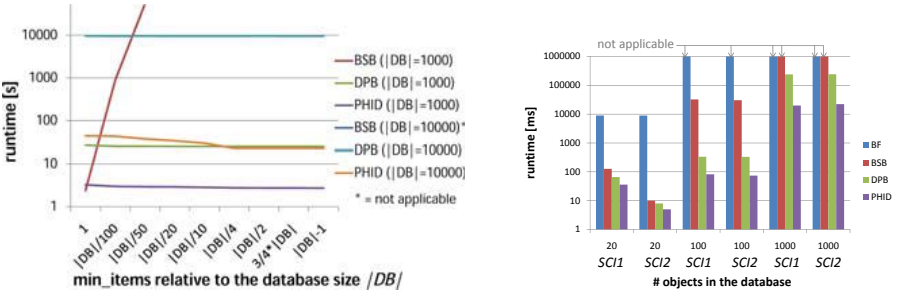
(b) Scalability experiments.

Fig. 2. Performance w.r.t database size

an individual five-dimensional hyper-rectangle R with a given size. The samples are uniformly distributed within the rectangles. The rectangles are arbitrarily distributed within the object space. Each of the 1500 objects of the two real-world datasets *SCI1* and *SCI2* consists of 10 samples, where each sample corresponds to a set of environmental sensor measurements of one single day that consist of several dimensions (attributes). The attribute set of *SCI1* describes temperature, humidity and CO concentration, whereas *SCI2* has a larger set of attributes (temperature, humidity, speed and direction of wind as well as concentrations of CO, SO_2, NO, NO_2 and O_3).

In this section, we compare two variants of our approach denoted by *DPB* and *PHID*. In contrast to *PHID*, *DPB* applies dynamic programming on the complete database, i.e. $\mathcal{D}' = \mathcal{D}$ and, thus, does not require the pre-processing step. The performance of *PHID* and *DPB* is compared to that of the brute-force solution (*BF*) by simply applying the formulas given in Section 3.2. Furthermore, we compare them to the bisection-based method (*BSB*) which is adapted to the method proposed in [1]. This method is able to significantly speed-up computation compared to the brute-force method, but is still exponential. Note that in our algorithm, we concentrate on the evaluation of the CPU-cost only. The reason is that the *PHID*-algorithm is clearly CPU-bound. The only I/O bottleneck is the initial computation of the likelihood that o is in the ϵ-range of $s_{o'}$, for each object $o \in DB$ and each sample $s_{o'}$, where $o' \in DB$ and $o \neq o'$. This requires a distance-range-self-join of the database which can be performed by a nested-block-loop join that requires $O(|DB|^2)$ page-faults in the worst case. In contrast, the CPU time for the *PHID*-algorithm is cubic: Each call of the dynamic programming algorithm requires $O(|DB|^2)$ time and has to be performed once for each sample in the database.

The first experiments relate to the scalability of the proposed approaches. The results depicted in Figure 2 demonstrate how the runtime of the competing techniques is influenced by the database size. Figure 2(a) shows that, though the bisection-based approach has exponential runtime, it outperforms the brute-force approach by several orders of magnitude. However, the dynamic-programming-based approaches scale significantly better than their competitors which in contrast to *DPB* and *PHID* have exponential runtime. Furthermore, the pre-processing step of *PHID* obviously pays off.



(a) Influence of min_items.

(b) Experiments on real-world data.

Fig. 3. Performance experiments

The performance can be further improved by an order of magnitude when applying the dynamic-programming technique only on objects o' where the probabilistic predicate $P_{\Phi_\varepsilon}(o, o')$ is not zero. The next experiment shows the scalability of *PHID* for different ε-range values. Here, the average time required to compute the hot item probability for an object was measured. The results shown in Figure 2(b) demonstrate that *PHID* scales well, even for very large databases. Figure 3(a) demonstrates the performance w.r.t. the min_items value for different database sizes. Contrary to *DPB* and *PHID*, the *BSB* method is very affected by the min_items value due to the expensive probability computation. The slight increase of the *DPB* and *PHID* performances can be explained by the reduced number of hot items with increasing min_items value.

Finally, we evaluate the performance based on real-world data (cf. Figure 3(b)). Unlike the exponential algorithms, *DPB* and *PHID* are able to perform a full hot item scan of the database in reasonable time, even for a relatively large database size.

6 Conclusions

In this paper, we propose an efficient approach for probabilistic queries for hot items, i.e. objects for which at least min_items other objects exist which are similar to o. In particular our approach computes for each object o in an uncertain database the probability that o is a hot item. We proposed methods that are able to break down the high computational complexity required to compute for an object o the probability, that o is a hot item. We theoretically and experimentally show that our approach can efficiently solve the problem (in worst-case $O(n^3)$) while the competing techniques have exponential runtime. Thereby, we achieve a speed-up of several orders of magnitude.

References

1. Bernecker, T., Kriegel, H.-P., Renz, M.: Proud: Probabilistic ranking in uncertain databases. In: Ludäscher, B., Mamoulis, N. (eds.) SSDBM 2008. LNCS, vol. 5069, pp. 558–565. Springer, Heidelberg (2008)

2. Brinkhoff, T., Kriegel, H.-P., Seeger, B.: Efficient processing of spatial joins using r-trees. SIGMOD Rec 22(2), 237–246 (1993)
3. Böhm, C., Braunmüller, B., Breunig, M.M., Kriegel, H.-P.: Fast Clustering Based on High-Dimensional Similarity Joins. In: Int. Conf. on Information Knowledge Management (CIKM) (2000)
4. Böhm, C., Pryakhin, A., Schubert, M.: The Gauss-Tree: Efficient Object Identification of Probabilistic Feature Vectors. In: Proc. 22nd Int. Conf. on Data Engineering (ICDE 2006), Atlanta, GA, US, p. 9 (2006)
5. Cheng, R., Kalashnikov, D., Prabhakar, S.: Evaluating Probabilistic Queries over Imprecise Data. In: Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD 2003), San Diego, CA), pp. 551–562 (2003)
6. Cheng, R., Xia, Y., Prabhakar, S., Shah, R., Vitter, J.: Efficient Indexing Methods for Probabilistic Threshold Queries over Uncertain Data. In: Proc. 30th Int. Conf. on Very Large Databases (VLDB 2004), Toronto, Canada, pp. 876–887 (2004)
7. Chui, C.-K., Kao, B., Hung, E.: Mining frequent itemsets from uncertain data. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS, vol. 4426, pp. 47–58. Springer, Heidelberg (2007)
8. Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD 1996), Portland, OR (1996)
9. Kriegel, H.-P., Kunath, P., Pfeifle, M., Renz, M.: Probabilistic Similarity Join on Uncertain Data. In: Li Lee, M., Tan, K.-L., Wuwongse, V. (eds.) DASFAA 2006. LNCS, vol. 3882, pp. 295–309. Springer, Heidelberg (2006)
10. Kriegel, H.-P., Pfeifle, M.: Density-based clustering of uncertain data. In: Proc. 11th Int. Conf. on Knowledge Discovery and Data Mining (KDD 2005), Chicago, IL, pp. 672–677 (2005)
11. Kriegel, H.-P., Renz, M., Schubert, M., Züfle, A.: Statistical density prediction in traffic networks. In: SDM, pp. 692–703 (2008)
12. Soliman, M., Ilyas, I., Chen-Chuan Chang, K.: Top-k Query Processing in Uncertain Databases. In: Proc. 23rd Int. Conf. on Data Engineering (ICDE 2007), Istanbul, Turkey, pp. 896–905 (2007)
13. Tang, J., Chen, Z., Fu, A.W.-C., Cheung, D.W.: Enhancing effectiveness of outlier detections for low density patterns. In: Chen, M.-S., Yu, P.S., Liu, B. (eds.) PAKDD 2002. LNCS, vol. 2336, p. 535. Springer, Heidelberg (2002)
14. Tao, Y., Cheng, R., Xiao, X., Ngai, W., Kao, B., Prabhakar, S.: Indexing Multi-Dimensional Uncertain Data with Arbitrary Probability Density Functions. In: Proc. 31th Int. Conf. on Very Large Data Bases (VLDB 2005), Trondheim, Norway, pp. 922–933 (2005)
15. Yi, K., Li, F., Kollios, G., Srivastava, D.: Efficient Processing of Top-k Queries in Uncertain Databases. In: Proc. 24th Int. Conf. on Data Engineering (ICDE 2008), Cancún, México (2008)

Spanning Tree Based Attribute Clustering

Yifeng Zeng, Jorge Cordero Hernandez, and Shuyuan Lin

Department of Computer Science, Aalborg University, DK-9220 Aalborg, Denmark
Department of Computer Science, Fuzhou University, FuJian, P.R.China
{yfzeng,corde}@cs.aau.dk,S030602108@fzu.edu.cn

Abstract. Attribute clustering has been previously employed to detect statistical dependence between subsets of variables. We propose a novel attribute clustering algorithm motivated by research of complex networks, called the Star Discovery algorithm. The algorithm partitions and indirectly discards inconsistent edges from a maximum spanning tree by starting appropriate initial modes, therefore generating stable clusters. It discovers sound clusters through simple graph operations and achieves significant computational savings. We compare the Star Discovery algorithm against earlier attribute clustering algorithms and evaluate the performance in several domains.

Keywords: Maximum Spanning Tree, Clustering.

1 Introduction

Probably one of the widest use of clustering in the past years has been the task of selecting genes (variable selection) in bioinformatics. The use of attribute clustering can be extended to any domain in the search for statistical correlation of variables. Several conventional clustering algorithms have been applied to regroup and reveal subsets of correlated attributes such as: the k-means algorithm [1], fuzzy clustering [2] and hierarchical clustering [3].

Recently, the k-modes algorithm [4] has been proved as one of the most efficient approaches for performing attribute clustering. However, it is subject to local optima due to a random selection of initial modes. In a parallel line, clustering based on tree partition receives more and more attention since it is firmly rooted in classical graph partition methods (detailed methods will be presented soon in the next section). More precisely, the clustering methods firstly build a maximum spanning tree (MAST) and then get the clusters using appropriate partition methods. For convenience, we call the methods as MAST-based clustering algorithms in this paper. Since the standard tree partition method is not directly oriented toward attribute clustering it may not produce competitive results. However, it avoids heavy computation in contrast with k-modes algorithm. Accordingly, the MAST-based clustering algorithms contribute to the growing line of research on attribute clustering.

For the effect of this investigation we focus on the MAST-based clustering method. Specifically, we introduce the Star Discovery (SD) algorithm that is

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 681–688, 2009.
© Springer-Verlag Berlin Heidelberg 2009

inspired by the research of complex networks [5]. We adopt the assumption that all variables can be seen as points in an Euclidean space (close points have a high correlation) because we have complete information regarding pairwise proximities. The SD algorithm sections the tree by detecting nodes which have a strong connectivity; then, it pulls neighboring nodes into clusters based on a simple heuristic. We compare our approach against both earlier tree-based clustering algorithms and the k-modes algorithm in comprehensive experiments.

2 Background

Given n domain attributes, $X = \{x_1, \cdots, x_n\}$, clustering methods aim to group a set of attributes[1] into clusters based on a similarity measure. In general, attributes in a cluster are more correlated to each other than to those ones belonging to different clusters. For this study, the data is statistically measured in terms of the interdependency redundancy measure $R(x_i, x_j) = \frac{I(x_i,x_j)}{H(x_i,x_j)}$; whereas $I(x_i, x_j) = \sum_{x_i,x_j \in X} p(x_i, x_j) \log \frac{P(x_i,x_j)}{p(x_i)p(x_j)}$ is the mutual information and $H(x_i, x_j) = \sum_{x_i,x_j \in X} p(x_i, x_j) \log p(x_i, x_j)$ is the joint entropy for the discrete random variables x_i and x_j [4]. The $R(\cdot, \cdot)$ measure discriminates a variable (containing many states) which has a weak statistical correlation with respect to another variable.

Without loss of generality, given a set of domain variables X, the objective of attribute clustering is to find a disjoint set of clusters $C = \{C_i | (i = 1, \cdots, k) \wedge (\forall_{i \neq j} C_i \cap C_j = \emptyset)\}$ that maximizes Eq. 1; where w_{o_i,x_j} denote the attached weight (measured by $R(o_i, x_j)$) from the center o_i to other variables x_j in the cluster C_i.

$$W^C = \sum_{C_i} \sum_{x_j \in (C_i - \{o_i\})} w_{o_i,x_j} \tag{1}$$

Two paradigms of clustering were taken in order to find optimal clusters of discrete random variables. The first technique is the k-modes algorithm that optimize Eq. 1 directly [4]. The k-modes can be seen as a graph partitioning algorithm. Thus, a set of discrete random variables are exhibited as nodes in a complete graph ($K = (V, E)$, where V denotes a set of nodes representing variables X, and E includes all edges that are associated with all pair-wise $R(\cdot, \cdot)$ estimates). Another clustering method is the MAST-based clustering algorithm which partitions and clusters a tree instead of the complete graph. The overhead of constructing a maximum spanning tree is in the order of $O(n \log n)$ using the Kruskal's algorithm. Typical approaches include: the standard Euclidean maximum spanning tree(SEMST) [6], the maximum cost spanning tree(CEMST) [7], and the Zahn's maximum spanning tree(ZEMST) [8].

[1] Discrete random variables (attributes) are seen as nodes in a graph ($V = X$, where V denotes a set of nodes). We will use any of these terms indifferently throughout this paper.

3 The Star Discovery Algorithm

We introduce the Star Discovery (SD) algorithm that iteratively partitions a MAST and form clusters until all nodes $x_i \in X$ are assigned to clusters. The SD algorithm clusters the domain in an unsupervised fashion (no initial number k of clusters is provided). Guiding the search for centers by only examining the topology or single weights is probably not a good idea since the whole domain is not taken into account. Similar to the ZEMST algorithm, we base the clustering on a simplistic search involving both topology and weights in neighborhoods. We look for subgraphs from the MAST that could reveal information about the "nature" of the domain. One abstraction of our technique is to look for spanning stars as subgraphs contained in the MAST. A spanning star [9] is a sub-tree over the MAST, $S = (V_S, E_S)$, and is composed of q nodes. It has a center $o \in V_S$ with a degree $q-1$ and all other nodes have a degree of one. The spanning star is our fundamental graph theoretical resource for expressing clusters that reside in a two dimensional Euclidean space.

Detecting the set of k-stars whose global weight is maximal(following Eq. 1) from a complete graph K requires expensive computation. Similar to the previous MAST partitioning algorithms, the SD algorithm aims to detect a set of spanning stars, $SS = \{S_1, \cdots, S_k\}$, such that the objective function in Eq. 2 is maximized.

$$W = \sum_{S_l \in SS} \Big(\sum_{x_i \in Adj_l} (w_{x_i,o_l}) + \sum_{x_j \in Adj_l, x_h \in Leaf_l} (w_{x_j,x_h}) \Big) \tag{2}$$

where o_l is the star(cluster) center, Adj_l is a set of adjacent nodes to the center node o_l, and $Leaf_l$ a set of leaf nodes that connect to either o_l or Adj_l.

Notice that we extend the notion of a star to include some leaf nodes (nodes whose degree is 1 in the graph). In the experimentation we found that leaf nodes have a higher correlation to the center of its adjacent node than to any other center in any other star. The SD algorithm optimizes the later function by ranking every variable according to its ability to serve as modes. The search heuristic will only select a star as a mode if its mode has not been used before in any other clusters. At the end we will acquire the set of clusters whose structure (modes, adjacent and leaf nodes) is maximal according to Eq. 2 and the heuristic presented in Fig. 1 [2].

The SD algorithm receives a MAST G and the set of weights W^G. At the very beginning the algorithm initializes an auxiliary set of variables V^{aux} and the counter l (line 1). After that, we build $n = |V|$ different stars, $S_r \in SS$, by specifying each variable x_r as the center o_r (line 3). For each star S_r, we include the adjacent nodes Adj_r to the center and leaf nodes $Leaf_r$ ($Deg(\cdot)$ denotes the node degree in the tree) (lines 4 and 6). Simultaneously, the edges are added (lines 5 and 7). Hence, the star S_r is a tuple having two sets: a set of nodes V_{S_r} and a set of edges E_{S_r} (line 9). In addition, we calculate the weight W^{S_r} in each star by adding all the weights attached to the star edges (line 10). Following,

[2] Note that $X \Leftarrow x$ indicates the addition of an element x to a given set X.

Star Discovery (SD) Algorithm

Input: $G = (V, E)$, W^G
Output: $C = \{C_1, C_2, \ldots, C_l\}$

1: $V^{aux} = V, V^{cont} = \emptyset, l = 1$
2: **FOR** $r = 1$ to n
3: $o_r = x_r$
4: $Adj_r \Leftarrow x_i$ **iff** $(x_i, o_r) \in E$
5: $E_{S_r} \Leftarrow (o_r, x_i)$
6: $Leaf_r \Leftarrow x_h$ **iff** $(x_i, x_h) \in E \wedge Deg(x_h) = 1$
7: $E_{S_r} \Leftarrow (x_i, x_h)$
8: $V_{S_r} = (o_r \cup Adj_r \cup Leaf_r)$
9: $S_r = (V_{S_r}, E_{S_r})$
10: $W^{S_r} = \sum_{(x_i, x_j) \in E_{S_r}} w_{x_i, x_j}$
11: $SS \Leftarrow S_r$
12: $W^{SS} \Leftarrow W^{S_r}$
13: Sort SS decreasingly according to W^{SS}
14: **WHILE** $V^{aux} \neq \emptyset$
15: $C_l = V_{S_l} - V^{cont}$
16: $V^{aux} = (V^{aux} - V_{S_l})$
17: $V^{cont} \Leftarrow V_{S_l}$
18: $C \Leftarrow C_l$
19: $l = l + 1$

Fig. 1. The Star Discovery Algorithm

the auxiliary star S_r is kept in SS (line 11) as well as its corresponding weight W^{S_r} in W^{SS} (line 12).

Once the set of stars, SS, have been built from the MAST we proceed to sort them decreasingly in terms of the star weights (line 13). The sorting forms a ranking of potential modes and those ones with a higher weight W^{S_r} will be selected to form clusters (this way we form only one possible arrangement of clusters). We elect the star as the cluster C_l that has the largest star weight among the remained stars (line 15). We use V^{cont} to exclude variables already contained in previous clusters (line 17). This avoids possible overlapping nodes between any pair of clusters. A set of clusters C are completed until no nodes are left.

Assuming that there are n variables and the highest cardinalities of adjacent nodes A_r and leaf nodes L_r are t and u respectively; then, the complexity in the first phase is $O(ntu)$ (lines 2-12) operations to search for all the adjacent nodes and leaves. The sorting operation takes at most $O(n \log n)$ if we use a merge-sort algorithm (line 13). The construction of clusters takes at most $O(l(t + u))$ operations (lines 14-19). Therefore the algorithm has a polynomial complexity $O((ntu) + (n \log n) + (l(t + u)))$. This polynomial complexity is better than the

one in k-modes since the number of variables t and u is fairly low. Moreover, the SD algorithm is executed for a single time and not for a number of iterations as in the k-modes algorithm.

The SD algorithm always provides solutions that are deterministic. On the other hand, SD might not offer results that are better in quality than the ones given from the k-modes algorithm. However, k-modes could obtain better solutions in some cases, but it has the risk of falling into local optima (the solution depends of the initial modes).

4 Experimental Results

We discuss the reliability of the k-modes algorithm and then compare the performance of the SD algorithm against the aforementioned algorithms. A sound estimate to evaluate the goodness of a set of clusters uses Eq. 1. In other words, we are concerned to calculate the local degree of dependency between the centers or 'modes', o_i, of each cluster, C_i, against its other elements. Then, a global weight adds up every local weight in the clusters to obtain a total weight, W^C.

For each experiment, we artificially generated datasets from some well known Bayesian networks such as: the Alarm (37 nodes), Barley (48 nodes), HeparII (70 nodes), Hailfinder (56 nodes) and Pathfinder (109 nodes) [3]. In this paper, we will show the performance of the SD algorithm against earlier algorithms; a detailed discussion of some specific application of attribute clustering is subject to future work.

Reliability of the k-modes algorithm: Indeed, the k-modes algorithm can detect the optimal clustering given our objective. However, there is a drawback by using this approach. Since the formulation of the k-modes algorithm is greedy, there is the risk of falling into local optima. In order to test the susceptibility of the k-modes algorithm to fall into local optima, we fed initial modes ($k = 2$) in each domain with all the possible $\binom{n}{2}$ combinations of variables, then we ran the experiment until it converges. For this experiment, we generated a dataset for each domain with a sample size $\Omega = 10000$. Table 1 presents the results.

Table 1. Number of local optima into which the k-modes algorithm falls

Domains	Alarm	HeparII	Hailfinder	Pathfinder
Local Optima	17	130	91	117

We found that k-modes does fall into local optima. For example, in the Alarm domain, it was interesting to see that k-modes converges into the optimal value of 6.13 with modes VentAlv and HR. However, it falls into 17 local optima having modes (VentAlv, LVEDVolume), (VentAlv, Shunt), etc. In the optimal result, the size of the clusters is about $\frac{n}{2}$ (18 variables). In many local optima, one

[3] http://genie.sis.pitt.edu/networks.html

cluster becomes relatively small (10 variables). Clearly, a small cluster is isolated because of the sub-optimal initial mode. Whenever LVEDVolume or Shunt are selected as a mode, then no improvement is made. These modes dominate their neighborhoods. The previous analysis is a straightforward example of techniques based solely on an iterative greedy search. As shown in Table 1, the k-modes algorithm falls in more local optima values in larger domains. These findings are a strong motivation for developing an algorithm that could detect the right initial modes.

Clustering quality and sensitivity: We ran all of the algorithms SEMST, CESMT, ZEMST, k-modes and SD (using $k = 8$); then, we compared the quality of the clustering results in terms of its global weight W^C. For the effects of this experiment and to avoid local optima we fed the k-modes algorithm with the resulting modes of the SD algorithm (notice that we also fed k-modes with the final modes which were obtained by the other methods, but it fell into local optima). On the other hand, it is interesting to investigate the response of the clustering algorithms using different sample sizes (k was set to 8). As the sample size Ω decreases, the lectures of the $R(\cdot, \cdot)$ measure become less accurate. Depending on the domain in study, there is a denominated level of sufficient statistics that determines the true nature of the MAST and reveals the true structure of correlated variables. Table 2 depicts the clustering results.

The SD algorithm performs better than the other tree-based clustering algorithms. Indeed, sometimes the SD algorithm is as effective as the k-modes algorithm. The later is true because if we consider the whole MAST in the cluster identification then we easily detect the strong components in the space. A highly connected variable in a MAST is very likely to be the best center in a given region. We can also conclude that more elaborated algorithms perform a better clustering. Clearly, the search spaces of the ZEMST and SD algorithms are relatively larger than the ones in the SEMST and CEMST approaches. Nevertheless, the search space of the SD algorithm is bigger than the one of ZEMST.

The SEMST, CEMST and ZEMST algorithms perform a *local* search on the MAST for clustering. For example, in the SEMST algorithm we completely disregard the inner relevance of an arc given the MAST topology. Thus, in practice, SEMST normally selects arcs connecting to leaf nodes as inconsistent (which in turn produces unbalanced bad clusters). In the CEMST algorithm, we take into account both weights and (up to some extent) the structure of the MAST. In this case, the inconsistent arcs have a maximal cost (which biases the search towards those arcs that are likely linked to highly connected nodes). The previous search technique is not enough since the search of inconsistent arcs is limited to a path of length 1. On the other hand, the ZEMST extends the search space by comparing the impact of removing an arc given some neighboring arcs and variables. Ultimately, the SD algorithm outperforms all the other tree-based algorithms because it calculates the clusters by considering both the weight and topology in the search. From the star formulation we realize that we could avoid local optima by discriminating those nodes that have a low connectivity and weight.

Table 2. Performance (W^C) of algorithms (Alg.) in five domains over different sample sizes ($\Omega = 10000, 8000, 6000, 4000$) and k=8. The k-modes algorithm is optimal when fed with the right initial modes.

Alg.	**10000**	**8000**	**6000**	**4000**
SEMST	4.13	18.41	21.61	22.99
CESMT	5.4	18.78	22.07	23.52
ZEMST	6.11	19.10	22.85	24.66
SD	7.85	**21.30**	**23.95**	**25.38**
k-modes	**8.35**	**21.30**	**23.95**	**25.38**

(*a*) Alarm Data

Alg.	**10000**	**8000**	**6000**	**4000**
SEMST	2.33	14.67	19.03	22.23
CESMT	2.55	14.85	19.24	22.48
ZEMST	3.85	14.91	20.70	24.20
SD	4.88	**15.39**	**21.02**	**25.41**
k-modes	**5.61**	**15.39**	**21.02**	**25.41**

(*b*) Barley Data

Alg.	**10000**	**8000**	**6000**	**4000**
SEMST	50.97	50.32	51.49	52.32
CESMT	51.21	50.55	51.71	52.89
ZEMST	51.27	51.43	52.55	53.54
SD	**55.57**	**56.98**	**58.34**	**59.56**
k-modes	**55.57**	**56.98**	**58.34**	**59.56**

(*c*) HeparII Data

Alg.	**10000**	**8000**	**6000**	**4000**
SEMST	30.26	31.33	32.42	33.65
CESMT	31.02	32.00	33.01	34.16
ZEMST	32.41	33.28	33.81	34.97
SD	**32.48**	**33.58**	**34.69**	**35.96**
k-modes	**32.48**	**33.58**	**34.69**	**35.96**

(*d*) Hailfinder Data

Alg.	**10000**	**8000**	**6000**	**4000**
SEMST	85.98	87.53	88.75	89.82
CESMT	88.63	88.22	89.40	90.19
ZEMST	88.315	88.75	89.64	90.61
SD	86.61	89.31	89.71	91.03
k-modes	**90.33**	**89.41**	**91.32**	**92.72**

(*e*) Pathfinder Data

Table 3. Elapsed times (in seconds) for algorithms in all domains

Alg.	Alarm	Barley	HeparII	Hailfinder	Pathfinder
SEMST	**0.031**	**0.04**	**0.044**	**0.049**	**0.047**
CESMT	0.04	0.042	0.056	0.05	0.062
ZEMST	0.078	0.057	0.065	0.07	0.094
SD	0.047	**0.04**	0.046	0.061	0.062
k-modes	0.109	0.063	0.077	0.078	0.125

Conclusively, we can learn that the MAST is in fact a useful dependence graph whenever a sound clustering method is applied to section it. The same trend holds if we supply different sample sizes or change the number k of clusters.

We can see that all algorithms have the same behavior for different sample sizes. Clearly, the SD algorithm outperforms any of other MAST-based clustering algorithms and obtains the same results as k-modes. Thus, the extensive search procedure of the SD algorithm secures competitive clustering.

Elapsed times: Finally, we investigated the running time of SD and other algorithms ($\Omega = 10000$). We used a system Centrino Duo with 2Ghz and 2

Gigabytes of memory. From Table 3 we can confirm that the algorithms calculate clusters obeying their complexity.

Logically, the SEMST algorithm is the fastest approach since it discards edges with the simplest rules. Ultimately, the elapsed times grow as the search space increases. The SD algorithm has a very competitive elapsed time (similar to the SEMST algorithm). We can see that in most cases, the SD clustering outperforms the k-modes algorithm in terms of elapsed times by a 50 percent ratio.

5 Conclusion

In this paper, we illustrated a comprehensive study between several clustering algorithms. We found that the SD algorithm is able to obtain clusters having a better quality than using other MAST-based clustering algorithms. Hence, the SD algorithm can compete with the k-modes algorithm in some cases; the advantage of the SD algorithm over k-modes is that we obtain a deterministic solution. The SD algorithm can also be used to select the initial modes to be fed to the k-modes algorithm for further clustering. We aid the search of clusters by revealing the nature of the domain through a MAST structure. The SD algorithm can be either used to perform the sectioning of a whole domain by itself, or to construct a hybrid algorithm (merged with the k-modes algorithm) which can find optimal clusterings (as shown in our experiments).

References

1. Smet, F.D., Mathys, J., Marchal, K., Thijs, G., DeMoor, B., Moreau, Y.: Adaptive quality-based clustering of gene expression profiles. Artificial Intelligence 18(5), 735–746 (2002)
2. Madeira, S., Oliveira, A.: Biclustering algorithms forn biological data analysis: A survey. IEEE/ACM Trans. on Computational Biology and Bioinformatics 1(1), 24–45 (2004)
3. Eisen, M.B., Spellman, P.T., Brown, P., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. In: Proceedings of National Academy of Sciences of the United States of America, pp. 14863–14868 (1998)
4. Au, W.H., Chan, K., Wong, A., Wang, Y.: Attribute clustering for grouping, selection, and classification of gene expression data. IEEE Trans. on Computational Biology and Bioinformatics 2(2), 83–101 (2005)
5. Cohen, D.B., Havlin, S.: Structural Properties of Scale Free Networks. Wiley-Vch., Berlin GmbH (2004)
6. Asano, M.K.T., Bhattacharya, B., Yao, F.: Clustering algorithms based on minimum and maximum spanning trees. In: Proceedings of the fourth annual symposium on Computational Geometry, pp. 252–257 (1998)
7. Ye, B., Chao, K.M.: Spanning Trees and Optimization Problems. Chapman and Hall, Boca Raton (2004)
8. Zahn, C.: Graph theoretical methods for detecting and describing gestalt clusters. IEEE Trans. in Computers 20, 68–86 (1971)
9. Gallian, J.: Dynamic survey of graph labeling. Electronic Journal of Combinatorics 14(6) (2007)

The Effect of Varying Parameters and Focusing on Bus Travel Time Prediction

João M. Moreira[1,3], Carlos Soares[2,3], Alípio M. Jorge[2,3], and Jorge Freire de Sousa[4]

[1] Faculdade de Engenharia, Universidade do Porto, DEI, Portugal
[2] Faculdade de Economia, Universidade do Porto, Portugal
[3] LIAAD-INESC Porto L.A., Portugal
[4] Faculdade de Engenharia, Universidade do Porto, DEIG, Portugal
jmoreira@fe.up.pt {csoares,amjorge}@fep.up.pt jfsousa@fe.up.pt

Abstract. Travel time prediction is an important tool for the planning tasks of mass transit and logistics companies. In this paper we investigate the use of regression methods for the problem of predicting the travel time of buses in a Portuguese public transportation company. More specifically, we empirically evaluate the impact of varying parameters on the performance of different regression algorithms, such as support vector machines (SVM), random forests (RF) and projection pursuit regression (PPR). We also evaluate the impact of the focusing tasks (example selection, domain value definition and feature selection) in the accuracy of those algorithms. Concerning the algorithms, we observe that 1) RF is quite robust to the choice of parameters and focusing methods; 2) the choice of parameters for SVM can be made independently of focusing methods while 3) for PPR they should be selected simultaneously. For the focusing methods, we observe that a stronger effect is obtained using example selection, particularly in combination with SVM.

1 Introduction

In the last 2 or 3 decades, mass transit companies made important investments in Advanced Public Transportation Systems, allowing them to collect massive amounts of data. The problem we address in this paper is the prediction of travel time 3 days ahead in order to improve planning of driver duties [4] and, consequently, to reduce costs with extra time. One of the challenges of this application is the dynamic nature of the data: different bus lines are affected by different factors (e.g., type of road) and the factors affecting the travel time for a single line may also change (e.g., creation of a bus lane). We empirically evaluate how the accuracy of several algorithms is affected by varying their parameters (Sect. 3) and by using different methods for the focusing tasks [5] (Sect. 4). The experimental setup is given in Sect. 2 and we close with conclusions.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 689–696, 2009.
© Springer-Verlag Berlin Heidelberg 2009

2 Experimental Setup

We simulate the real planning process by inducing models to predict travel times for day d using the data from days $d-32, \cdots, d-3$ for training. The experiments use data from a single bus line for the period 1/Jan/04 to 31/Mar/04. This corresponds to 90 days of data and, given that each training set has 30 days, we obtain $90-30-2=58$ different days for testing. The average number of trips in each training set is around 900. The set of variables used were {departure time, week day, day of the year and day type}, except where indicated otherwise. Three algorithms, support vector regression (SVM), random forests (RF) and project pursuit regression (PPR), were selected based on [3]. The generalization error is assessed using the *variation index* function, $varIndex = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{f}(\mathbf{x_i})-f(\mathbf{x_i}))^2}}{\frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x_i})}$, where f and $\hat{f}$ represent, respectively, the unknown true function and the predictor. All the experiments use the R-project [7].

3 Varying Parameters of Three Regression Algorithms

SVM. We use the $\nu-$SVM implementation from the R package e1071 [7], and we tested the linear, radial and sigmoid kernels. The parameters are presented in Table 1 and the results for each kernel are presented in Figs. 1 and 2 (left-hand side). Each parameter set is represented by c[C index]n[μ index]g[γ index]f[*coef0* index]. The parameter sets are ordered firstly by c then by n for the linear kernel, by c, n and g for the radial kernel and by f, g, n and c for the sigmoid kernel. The most striking result for SVM linear is the low sensitivity to different parameter settings, except for the largest values of C and the lowest values of ν. For SVM radial it is possible to observe that the higher the value of C is, the more sensitive the algorithm is to the remaining parameters. The sigmoid kernel has four parameters and consequently it is more difficult to analyze results. However, the performance of the algorithm decreases significantly with higher values *coef0*, except when the value of γ are low.

RF. Random Forest (RF) has two main parameters with impact on the accuracy of the predictions: the number of generated trees (the *ntree* parameter); and the number of variables randomly selected at each iteration (the *mtry* parameter). Given that result converge with increasing value of *ntree*, it is set to 1000, as

Table 1. Input parameters for SVM

algorithm	**cost** (C)	**nu** (ν)	**gamma** (γ)	**coef0**
SVM - linear	2^{2idx_1}	$\frac{\text{idx}_2}{10}$		
SVM - radial	$2^{2\text{idx}_3} \times 1000$	$\frac{\text{idx}_4}{5}$	$6\text{idx}_5/100000$	
SVM - sigmoid	$2^{2\text{idx}_3} \times 1000$	$\frac{\text{idx}_4}{5}$	$(2+12\text{idx}_6)/1000000$	-0.5idx_7

$\text{idx}_1 = -2, -1, \cdots, 6; \text{idx}_2 = 1, 2, \cdots, 10; \text{idx}_3 = 1, 2, \cdots, 5; \text{idx}_4 = 1, 2, \cdots, 4;$
$\text{idx}_5 = 1, 2, 3; \text{idx}_6 = 0, 1, \cdots, 4; \text{idx}_7 = -1, 0, \cdots, 4$

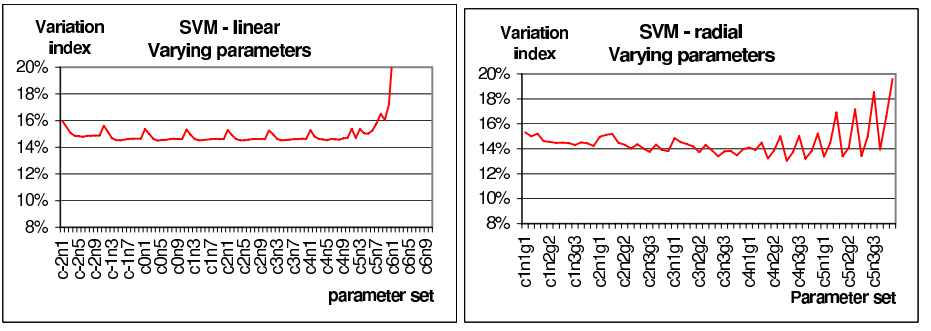


Fig. 1. The *variation index* for SVM linear and radial using different parameter sets

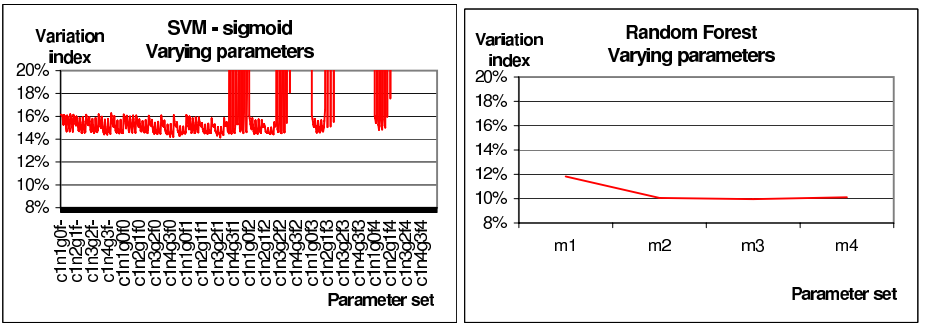


Fig. 2. The *variation index* for SVM sigmoid and RF using different parameter sets

suggested in [1], and given that we have four variables, $mtry \in \{1, 2, 3, 4\}$. The results (right-hand side of Fig. 2) show that RF is not very sensitive to *mtry*.

PPR. The PPR implementation in the stats package of R [7] was used. The settings tested for the general parameters were $nterms \in \{1, 2, 3, 4\}$ and $optlevel \in \{0, 1, 2, 3\}$. Additionally, the three ridge functions, called smoothers, available in the implementation were tested: super smoother, spline and gcvspline, each one with its own specific parameters (Table 2). Each parameter set is represented by n[*nterms* index]o[*optlevel* index]b[*bass* index]s[*span* index] for the super smoother and by n[*nterms* index]o[*optlevel* index]d[*df* index] for spline. The parameter sets are ordered by b, s, n and o for the super smoother and by d, n and o for the spline.

Results for PPR supsmu vary significantly (Fig. 3, left), depending on whether *bass* is equal to zero or not. Other than that, the effect of the other parameters is not very significant. When *bass* is equal to zero, the best results are obtained with lower values of *span*. For values different from zero, the results tend to degrade as the value of *bass* increases. For PPR spline (Fig. 3, right), the value of *df* strongly affects the impact of the remaining parameters. Indeed, for small

Table 2. Input parameters for PPR

algorithm	bass	span	df	gcvpen
PPR - supsmu	idx_1	0		
	0	$\text{idx}_2/10$		
PPR - spline			2^{idx_1}	
PPR - gcvspline				2^{2*idx_3}

$\text{idx}_1 = 0, 1, \cdots, 10; \text{idx}_2 = 1, 2, \cdots, 10; \text{idx}_3 = -2, -1, \cdots, 6$

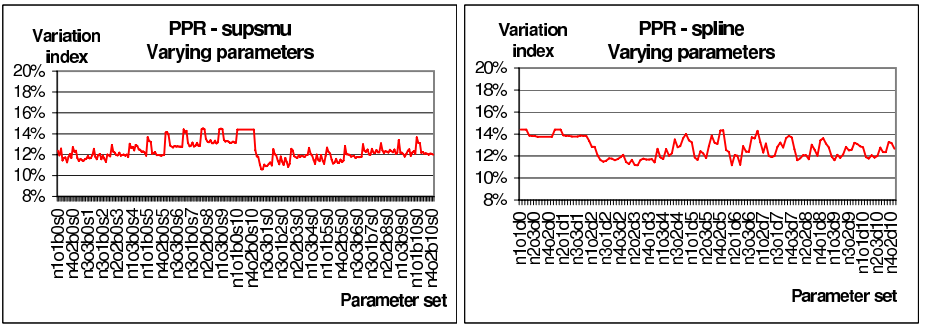


Fig. 3. The *variation index* for PPR supsmu and spline using different parameter sets

values of *df* the algorithm has low sensitivity to the remaining parameters while for larger values of *df*, the best results are obtained for smaller values of *nterm*. Results for gcvspline lead to similar conclusions to the ones presented here and, are thus omitted in interest of space.

4 Data Manipulation with Focusing Techniques

Example selection. The aim of example selection is to increase accuracy by selecting from the training set just a subset of the examples for the training task [5] Two approaches were tested. The first, referred to as *equivalent days* (ed), uses the examples from identical past days according to several groups, defined by visual inspection and using domain knowledge (e.g., Day Type = Normal and working days (from Monday to Friday); Day Type = Normal and Saturdays). If there is not a minimum of examples from equivalent days, according to the input parameter *min members*, all the examples from the training set are used. We have used *min members* = 10. The second, *leaf node* (ln), uses the examples from the same leaf of a CART [2] as the current test example: the CART model is induced on the full training set. In the following, 'All' presents the results using all the examples from the training set, i.e., without example selection.

The most important observation is that the two approaches behave differently for each one of the three methods. On the other hand, RF (results not plotted due to space limitations) are not very sensitive to example selection. This is not

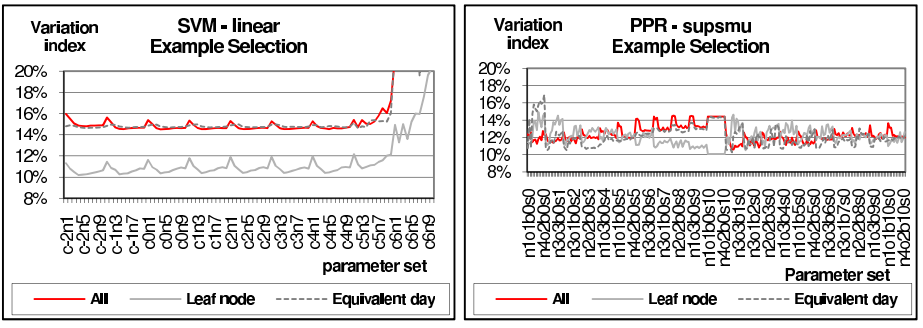


Fig. 4. The *variation index* for SVM linear and PPR - supsmu using example selection

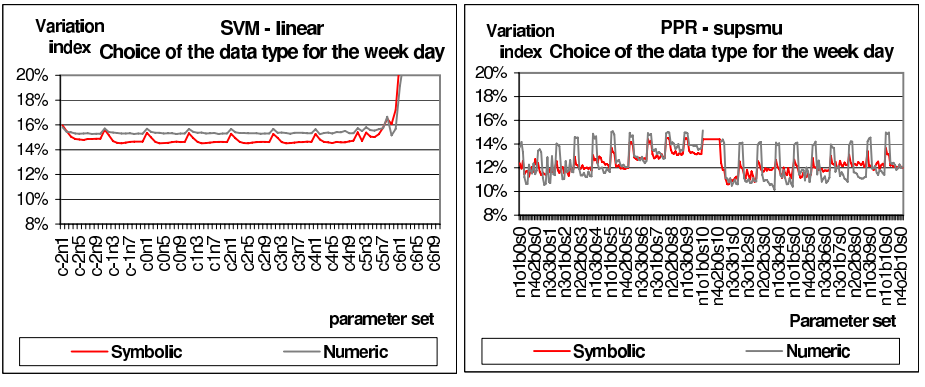


Fig. 5. The *variation index* for SVM linear and PPR supsmu using different data types for the variable week day

surprising because the CART algorithm already embeds example selection. For PPR (one variant is shown in Fig. 4, right), it is clear that both approaches for example selection can increase accuracy. For SVM (Fig. 4, left), the leaf node approach is the best one.

Domain values selection. Domain values selection can include: 1) the choice of the data type for each variable; 2) the discretization of continuous variables; or 3) the choice of appropriate values for a symbolic variable [5]. Here we have only tested changing the data type of the week day variable (WD), which was originally treated as symbolic.

The use of the numeric data type for the variable week day when using SVM is not promising (one variant is shown in Fig. 5, left). The change in data type of WD has small effect on the performance of SVM. PPR is more sensitive to the change in the type of WD, independently of the smoother (e.g., Fig. 5, right).

Table 3 presents the results of the best parameters settings after example selection and the choice of the data type for the week day variable. Focusing

Table 3. The *variation index* using example selection (ES) and different data types for the variable week day (CDT)

algorithm	**all**	**CDT**	**ES**	**ES & CDT**
SVM - linear	14.50%	15.14%	**10.18%**	
SVM - radial	13.03%	13.37%	**10.68%**	
SVM - sigmoid	14.15%	14.43%	**10.20%**	
RF	9.92%	9.88%	**9.80%**	
PPR - supsmu	10.60%	10.15%	10.09%	**9.73%**
PPR - spline	11.15%	10.24%	10.30%	**9.79%**
PPR - gcvspline	11.80%	10.21%	10.00%	**9.55%**

The best result using ES & CDT is, for the three PPR smoothers, obtained under the equivalent days approach.

with example selection achieves larger gains than the change in type of WD. Anyway, as observed earlier, the PPR benefits from the change in type of WD.

Feature selection. The four variables used so far were selected because of good results that were previously obtained [4]. However, to assess the effects of feature selection [5], more variables are necessary, so we included other variables describing the bus trips and also a few meteorological variables (Fig. 6, left). The first four variables capture the different seasonalities of the data, namely, daily (departure time), weekly (week day) and yearly (day of the year and week of the year) seasonality. The next three variables capture the possible impact factors, namely, the existence of holidays (day type), school holidays (school break) and number of Sundays before the pay day (Sundays unpd). The entrance and exit flow variables try to depict the occurrence of unusual flow at the entrances and exits of the town. The bus type can be short or long and the driver is a code identifying the driver of the bus. The service is an internal concept to the companies that represent the offer of trips. The three meteorological variables we use are the wind speed, temperature and precipitation. Feature selection was carried out using the RReliefF method [6]. For each variable, we compute the RReliefF weights for all training sets. Given that we have 58 different training sets (Sect. 2), we compute the statistics presented in Fig. 6 (left) for the set of 15 variables considered here. Based on these results, we have eliminated the variables week of the year, school break, Sundays until next pay, entrance flow and exit flow. A few observations can be made: (1) The low values for day type, entrance flow and exit flow can be explained by the small size of the training set (30 days) and the previous knowledge that these values are rarely different from the standard ones. (2) The low value for Sundays until next pay day can be due to the use of just one cycle on the seasonality of this event. Although we have eliminated it here, we note that this variable could be useful if using a larger training set. (3) Of the two variables used to capture the seasonality of the year: day of the year and week of the year, the first one is the most relevant. Due to the dependence between these two variables, the respective weights are expected to be lower than they should be [6].

	% of days where RW > 0.01
Departure time	96,62%
Week day	58,45%
Day of the year	14,98%
Week of the year	0,97%
Day type	1,45%
Schoolbreak	0,00%
Sundays unpd	0,97%
Bus type	64,25%
Driver	88,89%
Service	89,86%
Entrance flow	0,00%
Exit flow	0,00%
Wind speed	66,18%
Temperature	63,29%
Precipitation	16,91%

RW means RReliefF weight.

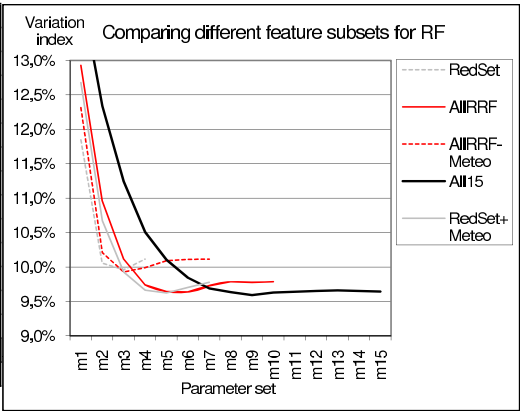


Fig. 6. RReliefF weights using 58 training windows (left-hand side) and the variation index for RF using different feature subsets (right-hand side)

Besides the set of all variables (All15) and the set of 10 variables selected with RReliefF (AllRRF), we have tested a variant without the meteorological variables (AllRRF-Meteo) and the original set of 4 variables, with and without the meteorological variables (RedSet+Meteo and RedSet). We only used the RF algorithm because its performance was quite stable in the previous experiments and, thus, it was not necessary to try different parameter settings and methods for the other focusing tasks. We have also tested the feature selection method embedded in RF, which limits the number of variables that are randomly selected in each split, given by the value of the *mtry* parameter. We note that, while the RReliefF method is global, because it selects a set of variables before running the algorithm, the method embedded in RF can be regarded as local, because the selection process is repeated several times during the learning process and may result in different sets of variables being selected. All possible values of the *mtry* parameter were tested for each set of variables (right-hand side of Fig. 6).

The best subset is All15, as expected, because of the feature selection mechanism which is characteristic of tree-based models (i.e., if a variable is not relevant, it will not be used in any of the nodes). However, the set obtained with RReliefF (AllRRF) obtains a similar accuracy. The results of the three sets containing meteorological values are generally better than the others. However, we note that we used the real values rather than 3-day ahead predictions. This means that, these are optimistic estimates of their accuracy. Finally, although the accuracy obtained with the original set of variables, RedSet, is generally the lowest, we observe that the range of values obtained by varying the *mtry* parameter is smaller in comparison to the other sets. This indicates that All15 contains informative variables which are not in the original set but also some which are misleading.

5 Conclusions

The prediction of travel time is an important tool to improve planning of driver duties, as well as other planning and customer information tasks. In this work, we evaluated empirically the impact of varying parameters on the performance of different regression algorithms (SVM, RF and PPR). We also evaluated the impact of the focusing tasks (example selection, domain value definition and feature selection) in the accuracy of those algorithms.

The results can be summarized as follows. (1) From the three regression methods we tested, RF is the most attractive as an off-the-shelf method. It has just one input parameter that is very easy to choose and the results are competitive. (2) The sensitivity of the results to the variation in input parameters displayed by SVM does not change meaningfully by varying the focusing methods. This means that the choice of parameter values and focusing methods can be made independently, thus reducing computational costs of modelling. Additionally, our results indicate that the leaf node approach for example selection is the most worthy of the focusing methods. (3) Conversely, the effect of varying parameters of PPR changes with the focusing method used. Therefore, it is advisable to set parameters and choose focusing methods simultaneously. (4) RF are able to obtain the best results from the extended set of variables. However, the results obtained with the smaller, original set are not much worse, and seem to be more robust, which indicates that the latter set may be more suitable with less robust algorithms, such as SVM and PPR.

Acknowledgments

This work was partially supported by FCT - Fundação para a Ciência e a Tecnologia, FEDER and Programa de Financiamento Plurianual de Unidades de I&D.

References

1. Breiman, L.: Manual - setting up, using, and understanding random forests v4.0. Technical report (2003)
2. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and regression trees. Chapman and Hall/CRC (1984)
3. Meyer, D., Leisch, F., Hornik, K.: The support vector machine under test. Neurocomputing 55(1-2), 169–186 (2003)
4. Moreira, J.M., Jorge, A.M., Sousa, J.F., Soares, C.: Trip time prediction in mass transit companies. a machine learning approach. In: Adavnced OR and AI methods in transportation, Poznan, pp. 276–283 (2005)
5. Reinartz, T.: A unifying view on instance selection. Data Mining and Knowledge Discovery 6(2), 191–210 (2002)
6. Robnik-Šikonja, M., Kononenko, I.: Theoretical and empirical analysis of relieff and rrelieff. Machine Learning 53(1-2), 23–69 (2003)
7. R Development Core Team.: R: A language and environment for statistical computing. Technical report, R Foundation for Statistical Computing (2006) ISBN 3-900051-07-0

Transfer Learning Action Models by Measuring the Similarity of Different Domains

Hankui Zhuo[1], Qiang Yang[2,⋆], and Lei Li[1]

[1] Software Research Institute, Sun Yat-sen University, Guangzhou, China
zhuohank@gmail.com, lnslilei@mail.sysu.edu.cn
[2] Hong Kong University of Science and Technology, Hong Kong
qyang@cse.ust.hk

Abstract. AI planning requires action models to be given in advance. However, it is both time consuming and tedious for a human to encode the action models by hand using a formal language such as PDDL, as a result, learning action models is important for AI planning. On the other hand, the data being used to learn action models are often limited in planning domains, which makes the learning task very difficult. In this paper, we present a new algorithm to learn action models from plan traces by transferring useful information from other domains whose action models are already known. We present a method of building a metric to measure the shared information and transfer this information according to this metric. The larger the metric is, the bigger the information is transferred. In the experiment result, we show that our proposed algorithm is effective.

1 Introduction

Planning systems require action models as input. A typical way to describe action models is to use action languages such as the planning domain description language (PDDL) [6]. A traditional way of building action models is to ask domain experts to analyze a planning domain and write a complete action model representation. However, it is very difficult and time-consuming to build action models in complex real world scenarios in such a way, even for experts. Thus, researchers have explored ways to reduce the human efforts of building action models by learning from observed examples or plan traces.

However, previous algorithms and experiments show that action model learning is a difficult task and the performances of the state-of-the-art algorithms are not very satisfying. A useful observation is that in many different planning domains, there exists some useful information that may be "borrowed" from one domain to another, provided that these different domains are similar in some aspects. In particular, we say that two domains A and B are similar if there is a mapping between some predicates of the two domains, in that the underlying principle of these actions, although their corresponding predicates are similar, resemble inherent similarities, then such a mapping can enable us to learn the action model in domain B by the mapping from the learned action model in domain A [9].

⋆ We thank the support of Hong Kong CERG Grant 621307.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 697–704, 2009.
© Springer-Verlag Berlin Heidelberg 2009

In this paper, we present a novel action model learning algorithm called t-LAMP (***t**ransfer **L**earning **A**ction **M**odels other domains*). We use the shared common information from *source* domains to help to learn action models from a *target* domain (we call the domains whose information is transferred *source* domains, while the domain from which the action models need to be learned a *target* domain). We propose a method of building a metric to measure the "similarity" between two domains, which is a difficult and not being answered question in planning domains. t-LAMP functions in the following three steps. Firstly, we encode the input plan traces into propositional formulas that are recorded as a DB. Secondly, we encode action models as a set of formulas. Finally, we learn weights of all formulas by transferring knowledge from source domains, and generate action models according to the weights of formulas.

The rest of the paper is organized as follows. We first give the definition of our problem and then describe the detailed steps of our algorithm. Then we will discuss some related works. In the experiment section, we will evaluate our algorithm in five planning domains of transfer learning action models and evaluate our transfer learning framework. Finally, we conclude the paper and discuss future work.

2 Related Work

Recently, some researchers have proposed various methods to learn action models from plan traces automatically. Jim, Jihie, Surya, Yolanda [3] and Benson [1] try to learn action models from plan traces with intermediate observations. What they try to learn are STRIPS-models [5,6]. One limitation of their algorithm is all the intermediate states need to be known. Yang, Wu and Jiang designed an algorithm called ARMS [2], which can learn action models from plan traces with only partial intermediate observations, even without observations.

Another related work is Markov Logic Networks (MLNs)[4]. MLN is a powerful framework that combines probability and first-order logic. A MLN is a set of weighted formulae to soften constraints in first-order logic. The main motivation behind MLN to "soften" constraints is that when a world violates a formula in a knowledge base, it is less probable, but not impossible.

In the transfer learning literature, Lilyana, Tuyen and Raymond[7] address the problem of how to leverage knowledge acquired in a source domain to improve the accuracy and speed of learning in a related target domain. [9] proposes to learn action models by transferring knowledge from another domain, which is the first try to transfer knowledge across domains.

3 Problem Definition

We represent a planning problem as $\mathcal{P} = (\Sigma, s_0, g)$, where $\Sigma = (S, A, \gamma)$ is the planning domain, s_0 is the initial state, and g is the goal state. In Σ, S is the set of states, A is the set of actions, γ is the deterministic transition function, which is $S \times A \rightarrow S$. A solution to a planning problem is called a plan, an action sequence $(a_0, a_1, \ldots, a_n)$ which makes a projection from s_0 to g. Each a_i is an *action schema* composed of a name and

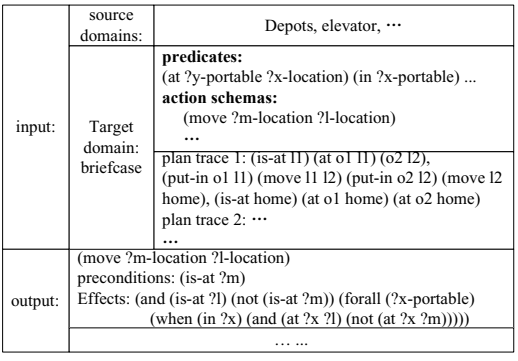


Fig. 1. An example of our problem definition (input and output)

zero or more parameters. A *plan trace* is defined as $T = (s_0, a_0, s_1, a_1, \ldots, s_n, a_n, g)$, where $s_1, ..., s_n$ are partial intermediate state observations that are allowed to be empty.

We state our learning problem as: given as input (1) a set of plan traces $\mathcal{T}$ in a *target domain* (that is, the domain from which we wish to learn the action models), (2) the description of predicates and action schemas in the target domain, and (3) the completely available action models in *source domains*, Our algorithm t-LAMP outputs preconditions and effects of each action model. An example of the input and output are shown in Fig.1.

4 The Transfer Learning Algorithm

Before giving our algorithm t-LAMP, we present an overview of the algorithm as shown in Fig.2. In the following subsections, we give the detail description of the main steps which are highlighted.

4.1 Encoding Each Plan Trace as a Proposition Database

As is defined in the problem definition, each plan trace can be briefly stated as an action sequence with observed states, including initial state and goal state. We need to encode states and actions which are also called state transitions. We represent facts that hold in states using propositional formulae, e.g. consider the *briefcase* domain in Fig.1. We have an object *o1* and a location *l1*. We represent the state where the object *o1* is in the briefcase and the briefcase is at location *l1* with the propositional formula: *in(o1)* $\wedge$ *is-at(l1)*, where *in(o1)* and *is-at(l1)* can be viewed as propositional variables. A model of the propositional formula is the one that assigns true value to the propositional variables *in(o1)* and *is-at(l1)*. Every object in a state should be represented by the propositional formula, e.g. if we have one more location *l2*, the above propositional formula should be modified as: *in(o1)*$\wedge$ *is-at(l1)* $\wedge$ $\neg$*is-at(l2)*. The behavior of deterministic actions is described by a transition function γ. For instance, the action *move(l1,l2)* in Fig.1 is described by $\gamma(s_1, move(l1, l2)) = s_2$. In s_1, the briefcase is at location *l1*, while in s_2, it is at *l2*. The states s_1 and s_2 can be represented by: *is-at(l1)* $\wedge$ $\neg$*is-at(l2)* and $\neg$*is-at(l1)*

the *t*-LAMP algorithm:
input: source domain descriptions $\{D_1, D_2, ..., D_n\}$, *plan traces* from the target domain,
action schemas of the target domain D_t.
output: *action model descriptions* of the target domain.

step 1. encode each plan trace as a formula in conjunctive form.
step 2. **for** each source domain D_i, **do**
step 3. encode all the action models of the domain D_i as a list of formulae $F(D_i)$.
step 4. find the best mapping MAP_i between D_i and D_t, the resulting formulae $F^*(D_t)$ and their weights.
step 5. **end**
step 6. generate *candidate formulae* to describe all the possible action models.
step 7. set the initial weights of all the *candidate formulae* as *zero*.
step 8. **for** each candidate formula f_j and its corresponding weight w_j **do**
step 9. **for** each MAP_i, **do**
step 10. if f_j is the same as f_k of the resulting $F^*(D_t)$ of MAP_i, then w_j=w_j+w_k.
step 11. **end**
step 12. **end**
step 13. learn weights of all the candidate formulae which are initially weighted by step 7-12.
step 14. select a subset of candidate formulae whose weights are larger than a threshold.
step 15. convert the selected candidate formulae to action models, and return.

Fig. 2. An overview of the t-LAMP algorithm

$\wedge$ *is-at(l2)*. We need different propositional variables that hold in different states to specify that a fact holds in one state but does not hold in another state. We introduce a new parameter in predicates, and represent the transition from the state s_1 to the state s_2 by *is-at(l1,* s_1*)* $\wedge$ $\neg$*is-at(l2,* s_1*)* $\wedge$ $\neg$*is-at(l1,* s_2*)* $\wedge$ *is-at(l2,* s_2*)*. On the other hand, the fact that the action *move(l1, l2)* causes the transition can be represented by a propositional variable *move(l1, l2,* s_1*)*. Thus, the function $\gamma(s_1, move(l1, l2))$ can be represented as $move(l1, l2, s_1) \wedge \text{is-at}(l1, s_1) \wedge \neg\text{is-at}(l2, s_1) \wedge \neg\text{is-at}(l1, s_2) \wedge \text{is-at}(l2, s_2)$. As a result, a plan trace can be encoded correspondingly.

Thus, plan traces can be encoded as a set of propositional formulae, each of which is a conjunction of propositional variables. As a result, each plan trace can be represented by a set of propositional variables, whose elements are conjunctive. This set is recorded in a database called DB, i.e. each plan trace is corresponded to its own DB.

4.2 Encoding Action Models as Formulae

We consider an action model is a strips model plus conditional effects, i.e. a precondition of an action model is a positive atom, and an effect is either a positive/negative atom or a conditional effect. According to the semantic of an action model, we equally encode an action model with a list of formulae, as addressed in the following.

T1: If an atom p is a positive effect of an action a, then p must hold after a is executed. The idea can be formulated by: $\forall i.a(i) \rightarrow \neg p(i) \wedge p(i{+}1)$, where i corresponds to s_i.
T2: Similar to T1, the negation of an atom p is an effect of some action a, which means p will *never hold* (be deleted) after a is executed, which can be formulated by: $\forall i.a(i) \rightarrow \neg p(i{+}1) \wedge p(i)$.
T3: If an atom p is a precondition of a, then p should hold before a is executed. That is to say, the following formula should hold: $\forall i.a(i) \rightarrow p(i)$.

the algorithm to learn weights w and the corresponding score *WPLL*:
input: a list of *DB*s, a list of formulae $F^*(D_t)$.
output: a list of weights w for the formulae $F^*(D_t)$, and *WPLL*.

step 1. initiate $w^0 = (0, \ldots, 0)$.
step 2. $i = 0$.
step 3. **repeat**
step 4. calculate $WPLL(w^i)$ using *DB*s and $F^*(D_t)$.
step 5. $w^{i+1} = w^i + \lambda * \partial WPLL(w^i)/\partial w^i$, where λ is a mall enough constant.
step 6. $i = i+1$;
step 7. **until** i is larger than a maximal iterative number.
step 8. output w^i and $WPLL(w^i)$.

Fig. 3. The algorithm to learn weights and the corresponding score `WPLL`

T4: A positive conditional effect, in PDDL form, like "*forall* $\bar{x}$ *(when* $f(\bar{x})$ $q(\bar{x})$*)*", is a conditional effect of some action a, which means for any $\bar{x}$, if $f(\bar{x})$ is satisfied, then $q(\bar{x})$ will hold after a is executed. Here, $f(\bar{x})$ is a formula in the conjunctive form of atoms. Thus a conditional effect can be encoded by: $\forall i.\bar{x}.a(\bar{x}, i) \wedge f(\bar{x}, i) \rightarrow q(\bar{x}, i{+}1)$.
T5: Similarly, a negative conditional effect of the form like "*forall* $\bar{x}$ *(when* $f(\bar{x})$ $\neg q(\bar{x})$*)*", can be encoded by: $\forall i.\bar{x}.a(\bar{x}, i) \wedge f(\bar{x}, i) \rightarrow \neg q(\bar{x}, i{+}1)$.
By T1-T5, we can encode an action model by requiring its corresponding formulas to be always true. Furthermore, for each source domain D_i, we can encode the action models in D_i with a list of formulae $F(D_i)$.

4.3 Building the Best Mapping

In step 4, we find the best mapping between the source domain and the target domain, to bridge these two domains. To map two domains, firstly, we need to map the predicates between the source domain D_i and the target domain D_t; secondly, map the action schemas between D_i and D_t. The mapping process of these two steps is the same, which is: *for each predicate* p_i *in* D_i *and a predicate* p_t *in* D_t*, we build a* ***unifier*** *by mapping their corresponding names and arguments (we require that the number of arguments are the same in* p_i *and* p_t*, otherwise, we find next* p_t *to be mapped with* p_i*); and then substitute all the predicates in* D_t *by this unifier; for each* p_i *and* p_t*, we repeat the process of* ***unifier-building*** *and* ***substitution*** *until the unifier-building process stops.*

By applying a mapping to the list of formulae $F(D_i)$, we can generate a new list of formulae $F^*(D_t)$, which encodes action models of D_t. We manage to calculate a score function on $F^*(D_t)$ to measure the similarity between D_i and D_t. We exploit the idea of [4,8] to calculate the score `WPLL` (which will be defined soon) when learning weights of formulae. The calculate process is given in Fig.3 In the highlighted step (step 4) of Fig.3, `WPLL`, the Weighted Pseudo-Log-Likelihood [4], is defined as $WPLL(w) = \sum_{l=1}^{n} \log P_w(X_l = x_l | MB_x(X_l))$ where, $P_w(X_l = x_l | MB_x(X_l)) = \frac{C_{(X_l=x_l)}}{C_{(X_l=0)}+C_{(X_l=1)}}$ and $C_{(X_l=x_l)} = \exp \sum_{f_i \in F_l} w_i f_i(X_l = x_l, MB_x(X_l))$. x is a possible world (a database DB). n is the number of all the possible *groundings* of atoms appearing in all the formulae $F^*(D_t)$, and X_l is the lth *groundings* of the all. $MB_x(X_l)$ is the state of the Markov blanket of X_l in x. The more detail description is presented by [4].

Using the algorithm, we will attain one score WPLL for each mapping. We keep the mapping (which is mentioned as the best mapping) with the highest score WPLL, the resulting $F^*(D_t)$ and their weights.

4.4 Generating Candidate Formulae and Action models

In steps 6 and 7, using the predicates and action schemas from D_t, we will generate all the possible action models by doing a combination between them. We initially associate each candidate formulae with a weight of zero to indicate that no contribution is provided initially.

From the definition of WPLL, we can see that the larger the WPLL is, the more probable the formulae $F^*(D_t)$ are satisfied by DBs, i.e. the more similar the source domain and the target domain (from which DBs are attained) are. Thus, we use WPLL to measure the similarity between source/target domains, and the weights of the resulting formulae $F^*(D_t)$ to transfer information of the "similarity". We exploit the idea that the "similarity" information is strengthened (weakened) when other domains strengthen (weaken) it, by simply adding up the weights "$w_j = w_j + w_k$" in step 10. With the weights attained by steps 7-12, in step 13 we learn weights of the candidate formulas by the algorithm of Fig.3.

From the learning process of WPLL, we can see that the optimization of WPLL indicates that when the number of true grounding of f_i is larger, the corresponding weight of f_i will be higher. In other words, the larger the weight of a candidate formula is, the more likely to be true that formula will be. When generating the final action models from these formulae in step 14, we need to determine a threshold, based on the validation set of plan traces and our evaluation criteria (definition of error rate), to choose a set of formulae converted to action models in step 15.

5 Experiments

5.1 Data Set and Evaluation Criteria

We collect plan traces from the following planning domains: *briefcase*[1], *elevator*[2], *depots*[3], *driverlog*[3], the plan traces numbers of which are 150, 150, 200 and 200 respectively. These plan traces are generated by generating plans from the given initial and goal states in these planning domains using the human encoded action models and a planning algorithm, FF planner[4]. Each of the domains will be used as the target domain in our experiment. The source domains are: *briefcase, elevator, depots, driverlog, zenotravel*[3].

We define error rates of our learning algorithm as the difference between our learned action models and the hand-written action models that are considered as the "ground truth". If a precondition appears in the preconditions of our learned action models

[1] http://www.informatik.uni-freiburg.de/ koehler/ipp.html
[2] http://www.cs.toronto.edu/aips2000/
[3] http://planning.cis.strath.ac.uk/competition/
[4] http://members.deri.at/ joergh/ff.html

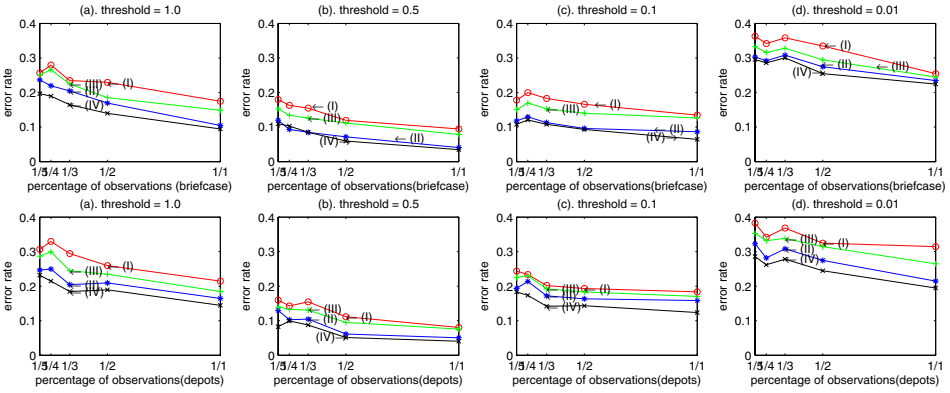


Fig. 4. Accuracy with different thresholds and percentage of observable intermediate states for learning action models of *briefcase* and *depots*

but not in the ones of hand-written action models, the error count of preconditions, denoted by $E(pre)$, increases by one. If a precondition appears in hand-written action models but not in our learned action models, $E(pre)$ increases by one. Likewise, error count of effects are denoted by $E(eff)$. Furthermore, we denote the total number of all the possible preconditions and effects of action models as $T(pre)$ and $T(eff)$, respectively. In our experiments, the error rate of an action model is defined as $R(a) = \frac{1}{2}(E(pre)/T(pre) + E(eff)/T(eff))$, where we assume the error rates of preconditions and effects are equally important, and the range of error rate $R(a)$ should be within [0,1]. Furthermore, the error rate of all the action models A is defined as $R(A) = \frac{1}{|A|}\sum_{a \in A} R(a)$, where $|A|$ is the number of A's elements.

5.2 Experimental Results

The evaluation results of t-LAMP in two domains are shown in Fig.4. The red curve (I) is the learning result without transferring any information from other domains; the blue curve (II) is the learning result with transferring information from the most similar domain based on WPLL; the green curve (III) is the result with transferring information from the least similar domain based on WPLL; the black curve (IV) is the result with transferring information from all the other source domains (when learning action models of *briefcase*, the source domains are *elevator, depots, driverlog, zenotravel*). From these two figures, we can see that, the result by transferring information from all the other source domains is the best. Furthermore, by comparing the results of (II) and (III), we can see that, when we choose the most similar domain for transferring, the result is generally better than choosing the least similar domain, i.e. the score function WPLL works well in measuring the similarity of two domains.

The first row of Fig.4 shows the result of learning the action models of *briefcase* with transferring the information from *depots, driverlog, zenotravel, elevator*, while the second row shows the result of learning the action models of *depots* with transferring the information from *briefcase, driverlog, zenotravel, elevator*. We have chosen different

thresholds with weights 1.0, 0.5, 0.1 and 0.01 to test the effect of the threshold on the performance of learning. The results show that generally the threshold can be neither too large nor too small, but the performance is not very sensitive to the choice of the value. An intuitive explanation is that, a threshold that is too large may lose useful candidate formulae, and a threshold that is too small may contain too many noisy candidate formulae that will affect the overall accuracy of the algorithm. This intuition has been verified by our experiment. In our experiment, it can be shown that when we set the threshold as 0.5, the mean average accuracy is the best.

Our experiment shows that in most cases, the more states that are observable, the lower the error rate will be, which is consistent with our intuition. However, there are some other cases, e.g. when threshold is set to 0.01, when there are only $1/4$ of states that are observable, the error rate is lower than the case when $1/3$ of states are observable.

From our experiment results, we can see that transferring useful knowledge from another domain will help improve our action model learning result. On the other hand, determining the similarity of two domains is important.

6 Conclusion

In this paper, we have presented a novel approach to learn action models through transfer learning and a set of observed plan traces. we propose a method to measure the similarity between domains and make use of the idea of *Markov Logic Networks* to learn action models by transferring information from other domains according to "similarity". Our empirical tests show that our method is both accurate and effective in learning the action models via information transfer. In the future, we wish to extend the learning algorithm to more elaborate action representation languages including resources and functions. We also wish to explore how to make use of other inductive learning algorithms to help us learn better.

References

1. Blythe, J., Kim, J., Ramachandran, S., Gil, Y.: An integrated environment for knowledge acquisition. IUI, 13–20 (2001)
2. Yang, Q., Wu, K., Jiang, Y.: Learning action models from plan examples using weighted MAX-SAT. Artif. Intell. 171(2-3), 107–143 (2007)
3. Benson, S.: Inductive Learning of Reactive Action Models. In: ICML, pp. 47–54 (1995)
4. Richardson, M., Domingos, P.: Markov Logic Networks. Machine Learning 62(1-2), 107–136 (2006)
5. Fikes, R., Nilsson, N.J.: STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. Artif. Intell. 2(3/4), 189–208 (1971)
6. Fox, M., Long, D.: PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains. J. Artif. Intell. Res. (JAIR) 20, 61–124 (2003)
7. Mihalkova, L., Huynh, T., Mooney, R.J.: Mapping and Revising Markov Logic Networks for Transfer Learning. In: AAAI (2007)
8. Kok, S., Singla, P., Richardson, M., Domingos, P.: The Alchemy system for statistical relational AI, University of Washington, Seattle (2005)
9. Zhuo, H., Yang, Q., Hu, D.H., Li, L.: Transferring Knowledge from Another Domain for Learning Action Models. In: PRICAI (2008)

On Optimal Rule Mining: A Framework and a Necessary and Sufficient Condition of Antimonotonicity

Yannick Le Bras[1,3], Philippe Lenca[1,3], and Stéphane Lallich[2]

[1] Institut Telecom; TELECOM Bretagne; UMR CNRS 3192 Lab-STICC
Technopôle Brest-Iroise - CS 83818 - 29238 Brest Cedex 3 - France
{yannick.lebras,philippe.lenca}@telecom-bretagne.eu
[2] Université de Lyon, Laboratoire ERIC, Lyon 2, France
stephane.lallich@univ-lyon2.fr
[3] Université européenne de Bretagne, France

Abstract. Many studies have shown the limits of support/confidence framework used in Apriori-like algorithms to mine association rules. There are a lot of efficient implementations based on the antimonotony property of the support but candidate set generation is still costly. In addition many rules are uninteresting or redundant and one can miss interesting rules like nuggets. One solution is to get rid of frequent itemset mining and to focus as soon as possible on interesting rules. For that purpose algorithmic properties were first studied, especially for the confidence. They allow all confidence rules to be found without a preliminary support pruning. More recently, in the case of class association rules, the concept of optimal rules gave a pruning strategy compatible with more measures. However, all these properties have been demonstrated for a limited number of interestingness measures. We present a new formal framework which allows us to make the link between analytic and algorithmic properties of the measures. We apply this framework to optimal rules, and we demonstrate a necessary and sufficient condition of existence for this pruning strategy, which can be applied to any measure.

Keywords: Association rules, pruning, algorithms, analytic, nuggets, redundance, optimal rules.

1 Introduction

Rule discovery has attracted a lot of attention from data mining researchers probably because rules are an understandable representation of knowledge. In unsupervised paradigm, association rule mining is certainly the most popular method. Since its initial formulation by [1], the problem of association rule mining -and the underlying problem of the frequent set mining- has focused many works. Our paper focuses on association rule mining, especially on classification rule mining as proposed by [2] i.e. with one predetermined target like in supervised learning.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 705–712, 2009.
© Springer-Verlag Berlin Heidelberg 2009

An association rule is a rule $\mathtt{A} \rightarrow \mathtt{B}$, where $\mathtt{A}$ and $\mathtt{B}$ are two sets of items (also called itemsets) such that $\mathtt{A} \neq \emptyset$, $\mathtt{B} \neq \emptyset$ and $\mathtt{A} \cap \mathtt{B} = \emptyset$, meaning that given a database $\mathcal{D}$ of transactions (where each transaction is a set of items) whenever a transaction $\mathtt{T}$ contains $\mathtt{A}$, then $\mathtt{T}$ probably contains $\mathtt{B}$ also [1]. The problem of mining association rules is then to generate all association rules that have support and confidence greater than a user-specified minimum support and minimum confidence respectively. Support is defined as the proportion of transactions containing $\mathtt{A}$ and $\mathtt{B}$ in $\mathcal{D}$ (noted $\mathbb{P}(\mathtt{AB})$ or $supp(\mathtt{AB})$), while confidence is the proportion of transactions containing $\mathtt{A}$ and $\mathtt{B}$ inside the set of transactions containing $\mathtt{A}$ in $\mathcal{D}$ (noted $\mathbb{P}(\mathtt{B}|\mathtt{A})$ or $conf(\mathtt{A} \rightarrow \mathtt{B})$). Association rule mining is a two-step process: minimum support constraint is firstly applied to find all frequent itemsets in $\mathcal{D}$ and secondly, the frequent itemsets and the minimum confidence constraint are used to form rules.

Finding all frequent itemsets in $\mathcal{D}$ is computationally expensive since there are 2^k possible itemsets where k is the number of items in $\mathcal{D}$. However using the downward-closure property of support (also called antimonotonicity), very efficient algorithms [3,4] can find all frequent itemsets. An interesting survey may be found in [5]. This step gives in any circumstances an important role to the support constraint and mining interesting rules without the support requirement has been identified as an important problem. Some researches try to avoid the use of support [6] or to avoid the task of threshold fixation [7].

Finding all and only all interesting rules is also a problem because the generated rule sets are quite large, especially within the support-confidence framework while the percentage of interesting rules is often only a very small fraction of the all rules. A first strategy to reduce the number of mined rules consists of increasing the user-specified minimum support. Unfortunately with this strategy many interesting rules and especially nuggets will be missed. A second strategy consists of increasing the user-specified minimum confidence. This will favor rules with large consequent which may give many uninteresting rules.

An another popular strategy is to rank the rules in a post-analysis phase with additional objective measures of interest. A large number of interestingness measures were proposed. Interesting surveys and comparisons may be found in [8,9,10]. An another way of reducing the rule sets is to allow the user to specify which items have to or cannot be part of the left-hand-side or the right-hand-side of the rules. This is the case for associative classification which focuses on association rules whose right-hand-side are restricted to a class attribute [2,11]. Unfortunately these strategies are always subject to the dictatorship of support. A more efficient approach is to apply additional constraints [12] on item appearance or to use additional interestingness measures as soon as possible to reduce both the time to mine databases and the number of founded itemsets [13]. In particular we are here interested in measures that can reduce the search space and also be useful for the evaluation of the quality of mined patterns. We believe that this approach is certainly the most promising one. To overcome the

previous mentioned problems with the dictatorship of the support constraint different solutions were proposed at the algorithmic level and mainly for the confidence measure.

The rest of the paper is organized as follows. In Section 2, we present an overview of recent works that focus on the algorithmic properties of measures. In Section 3, we present a new formal framework which allows us to make the link between analytic and algorithmic properties of the measures. We apply in Section 4 this framework to optimal rules, and we demonstrate a necessary and sufficient condition of existence for applying a pruning strategy with a large set of measures. We conclude in Section 5.

2 State of Art

To get rid of support constraint, several authors proposed to focus on the algorithmic properties of confidence. The goal is to succeed in getting all the high confident rules, even the nuggets of knowledge. We here briefly review some of these works. Some of these works are mainly based on algorithmic properties of confidence, while others focus on different measures, by exploiting their intrinsic properties.

In [14] the authors introduce the *h-confidence* a new measure over itemsets mathematically identical to *all-confidence* proposed by [15]. Both of these measures are antimonotone. They also introduce the concept of cross-support patterns i.e. - uninteresting - patterns involving items with substantially different support levels. The authors thus propose *hyperclique miner*, an efficient algorithm that utilizes both the cross-support and anti-monotone properties of the h-confidence measure. In [16] the authors introduce the *Universal Existential Upward Closure* property based on a certain monotonicity of the confidence. This property applies to classification rules and allows to examine only confident rules of larger size for generating confident rules of smaller size. The authors deduce from this property a top-down confidence-based pruning strategy.

In [17] the authors adapt a technique by [18] to propose a branch-and-bound algorithm for associative classification, based on an antimonotonic property of convexity of the χ^2. In [19] the authors introduce a new type of antimonotony, called Loose Anti-Monotony, that can be applied for statistical constraints. This concept is then efficiently exploited in an Apriori-like algorithm. In [20] the author introduces the notion of optimal rule set for classification rules. A rule set is optimal if it contains all rules, except those with no greater interestingness than one of its more general rules. This concept defines a pruning strategy that applies to a large set of measures. The author gives an individual proof for 12 measures.

In the following, we introduce a new framework and use it to link together analytic properties of measures and the antimonotone property of [20].

3 A Framework for Measure's Study

3.1 Adapted Functions of Measure

Firstly, we will precise the concept of associated measure introduced in [21]. In this article, measures are considered as functions of $\mathbb{R}^3 \longrightarrow \mathbb{R}$. The authors only focus on the parametrization of interestingness measures in function of the number of examples, antecedents, and consequents. Since the antimonotone property of optimal rule mining relies on the number of counter-examples, we will study the behavior of measures according to this quantity. Similar approaches can be found in [22,9,23,10]. We here give a general framework to study analytic properties of measures.

Definition 1. *(adapted function of measure) Let call function of measure adapted to a given measure of interest m a couple $(\mathcal{D}, \phi_m)$, where $\mathcal{D} \subset \mathbb{Q}^3$ and $\phi_m : \mathcal{D} \longrightarrow \mathbb{R}$, verifying:*
(1) $\forall$ association rule r, $\exists\ X \in \mathcal{D}$ such that $\phi_m(X) = m(r)$;
(2) $\forall\ X \in \mathcal{D}$, $\exists$ an association rule r such that $m(r) = \phi_m(X)$.
The domain $\mathcal{D}$ is then called an adapted domain.

We thus define a surjection from the space of association rules, over all the databases, in $\mathcal{D}$. The specification of $\mathcal{D}$ allows to make the difference between all the parametrizations: support of examples, support of counterexamples, or confidence of the rule. The adapted function depends of the parametrization. The second condition assures of the usability of $\mathcal{D}$ by assuming that every point in $\mathcal{D}$ corresponds to a real situation.

Example 1. (Recall Measure) $Rec(\mathtt{A} \rightarrow \mathtt{B}) = \frac{y-x}{z}$ where $x = \mathbb{P}(\mathtt{A}\neg\mathtt{B})$, $y = \mathbb{P}(\mathtt{A})$ and $z = \mathbb{P}(\mathtt{B})$. If we study its variations according to z, we find that it increases if $y - x < 0$ and decreases if $y - x > 0$. But constraints of support let us assume that $y > x$, thus *Rec* decreases with z.

3.2 A Domain for Counterexamples

We present now a domain according to counterexamples. Using the constraints $0 < \mathbb{P}(\mathtt{A}), \mathbb{P}(\mathtt{B}) < 1$, $0 \leq \mathbb{P}(\mathtt{A}\neg\mathtt{B}) \leq \mathbb{P}(\mathtt{A})$ and $\mathbb{P}(\mathtt{A}) - \mathbb{P}(\mathtt{B}) \leq \mathbb{P}(\mathtt{A}\neg\mathtt{B}) \leq \mathbb{P}(\neg\mathtt{B})$, we define the following domain, and prove that it is adapted:

$$\mathcal{D} = \left\{ \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{Q}^3 \middle| \begin{array}{ccc} 0 & < y < & 1 \\ 0 & < z < & 1 \\ \max\{0, y-z\} & \leq x \leq & \min\{y, 1-z\} \end{array} \right\}.$$

Table 1. Database for the domain $\mathcal{D}$

	1		$x \cdot n$			$y \cdot n$			$(z+x) \cdot n$			n
A	1	$\cdots$	$\cdots$	$\cdots$	$\cdots$	1	0	$\cdots$	$\cdots$	$\cdots$	$\cdots$	0
B	0	$\cdots$	0	1	$\cdots$	$\cdots$	$\cdots$	$\cdots$	1	0	$\cdots$	0

Property 1. Let m be an interestingness measure such that there exists ϕ_m with $m(\mathtt{A} \rightarrow \mathtt{B}) = \phi_m(\mathbb{P}(\mathtt{A}\neg\mathtt{B}), \mathbb{P}(\mathtt{A}), \mathbb{P}(\mathtt{B}))$. Then the couple $(\mathcal{D}, \phi_m)$ is an adapted function of measure for m.

Proof. The first condition comes immediately from definitions by taking $X = \big(\mathbb{P}(\mathtt{A}\neg\mathtt{B}), \mathbb{P}(\mathtt{A}), \mathbb{P}(\mathtt{B})\big)$. For the second one, considering an element (x, y, z) of $\mathcal{D}$, we need to construct a database $\mathcal{DB}$ containing an association rule $\mathtt{A} \rightarrow \mathtt{B}$, such that $m(\mathtt{A} \rightarrow \mathtt{B})$ equals to $\phi_m(x, y, z)$. We define n as an integer such that $(x \times n, y \times n, z \times n) \in \mathbb{N}^3$. Our database should verify the following equalities $\mathbb{P}(\mathtt{A}\neg\mathtt{B}) = x$, $\mathbb{P}(\mathtt{A}) = y$, $\mathbb{P}(\mathtt{B}) = z$. The constraints on the domain assure that $x \leq y \leq x + z \leq 1$ holds. We can thus construct the database of table 1 satisfying the equalities.

We now benefit of a rigorous formal framework for the study of interestingness measures. We will use it in the following to establish a necessary and sufficient condition (NSC) for existence of an antimonotone property. We thus focus on the parametrization with counterexamples.

4 General Optimal Rule Discovery

4.1 A Condition of Optimonotony

From now, we use the same notations as in [20]: $\mathtt{P}$ is an itemset, $\mathtt{X}$ is an item not in $\mathtt{P}$ and $\mathtt{c}$ a class item in a supervised context.

Theorem 1. *(Antimonotone property, [20]) If* $supp(\mathtt{PX}\neg\mathtt{c}) = supp(\mathtt{P}\neg\mathtt{c})$ *then rule* $\mathtt{PX} \rightarrow \mathtt{c}$ *and all its more-specific rules will not occur in an optimal rule set defined by confidence, odds ratio, lift, gain, added-value, Klosgen, conviction, p-s, Laplace, cosine, certainty factor, or Jaccard.*

In addition Li et al. [24] prove that the Relative Risk verifies this property. We introduce the general notion of *optimonotony*, the antimonotone property for optimal rule mining, and formulate a NSC for it.

Definition 2. *(optimonotony) A measure of interest* m *is optimonotone if, given a rule* $\mathtt{P} \rightarrow \mathtt{c}$ *and a specification* $\mathtt{PX} \rightarrow \mathtt{c}$*, we have* $(supp(\mathtt{PX}\neg\mathtt{c}) = supp(\mathtt{P}\neg\mathtt{c}) \implies m(\mathtt{PX} \rightarrow \mathtt{c}) \leq m(\mathtt{P} \rightarrow \mathtt{c}))$.

An optimonotone measure can be clearly added to the list of theorem 1.

Theorem 2. *(NSC of optimonotony) Let* m *be an interestingness measure, and* $(\mathcal{D}, \phi_m)$ *its adapted function of measure.* m *is optimonotone iff* ϕ_m *increases with the second variable (associated with* $supp(\mathtt{P})$*).*

Proof. Sufficiency. Let ϕ_m increases with the second variable, $\mathtt{P}$ be an itemset, $\mathtt{c}$ a class attribute, $\mathtt{X}$ an item $\notin \mathtt{P}$, and $supp(\mathtt{P}\neg\mathtt{c}) = supp(\mathtt{PX}\neg\mathtt{c})$, we show

$m(\mathtt{PX} \to \mathtt{c}) \leq m(\mathtt{P} \to \mathtt{c})$. Since $supp(\mathtt{P}) \geq supp(\mathtt{PX})$, we write:

$$\begin{aligned} m(\mathtt{PX} \to \mathtt{c}) &= \phi_m(supp(\mathtt{PX}\neg\mathtt{c}), supp(\mathtt{PX}), supp(\mathtt{c})) \\ &\overset{\text{hypothesis}}{=} \phi_m(supp(\mathtt{P}\neg\mathtt{c}), supp(\mathtt{PX}), supp(\mathtt{c})) \\ &\overset{\text{monotony}}{\leq} \phi_m(supp(\mathtt{P}\neg\mathtt{c}), supp(\mathtt{P}), supp(\mathtt{c})) = m(\mathtt{P} \to \mathtt{c}) \end{aligned}$$

Necessity. We want to prove that, given two elements (x, y, z) and (x, y', z) of $\mathcal{D}$ such that $y' \leq y$, we have $\phi_m(x, y', z) \leq \phi_m(x, y, z)$. Let n be an integer such that $n \times x$, $n \times y$, $n \times y'$ and $n \times z$ are integers. We construct the database $\mathcal{DB}$ of table 2, in which the supports are the followings: $supp(\mathtt{P}\neg\mathtt{c}) = x = supp(\mathtt{PX}\neg\mathtt{c})$, $supp(\mathtt{c}) = z$, $supp(\mathtt{P}) = y$, $supp(\mathtt{PX}) = y'$. Thus, we do have the equality $supp(\mathtt{PX}\neg\mathtt{c}) = supp(\mathtt{P}\neg\mathtt{c})$, and deduce, following the hypothesis of implication, that $m(\mathtt{PX} \to \mathtt{c}) \leq m(\mathtt{P} \to \mathtt{c})$. Since (D, ϕ_m) is a function of measure

Table 2. Database $\mathcal{DB}$

	1		$(z-(y-x))\times n$			$(z-(y'-x))\times n$			$z \times n$			$(z+x)\times n$			n
P	0	⋯	0	1	⋯	⋯	⋯	⋯	⋯	⋯ ⋯		1	0	⋯	0
X	0	⋯	⋯	⋯	⋯	0	1	⋯	⋯	⋯ ⋯		1	0	⋯	0
c	1	⋯	⋯	⋯	⋯	⋯	⋯	⋯	1	0	⋯	⋯	⋯	⋯	0

Table 3. Expression of measures in $\mathcal{D}$ and relation to optimonotony.

measure	$(x, y, z) \in \mathcal{D}$	optimonotone	measure	$(x, y, z) \in \mathcal{D}$	optimonotone
support	$y - x$	Y	**confidence**	$1 - \frac{x}{y}$	Y
cover	y	Y	**prevalence**	z	Y
recall	$\frac{y-x}{z}$	Y	**specificity**	$\frac{1-z-x}{1-y}$	Y
relative specificity	$\frac{1-z-x}{1-y} - 1 + y$	Y	**precision**	$y - 2x + 1 - z$	Y
lift	$\frac{1}{z} - \frac{x}{yz}$	Y	**leverage**	$1 - \frac{x}{y} - yz$	N
centered confidence	$1 - \frac{x}{y} - z$	Y	**relative risk**	$1 + \frac{(1-z)-x}{1-y}$	Y
Jaccard	$\frac{y-x}{z+x}$	Y	**positive confidence**	$1 - \frac{x}{y(1-z)}$	Y
interest	$\lvert y(1-z) - x\rvert$	N	**odds ratio**	$1 + \frac{y(1-z)-x}{x(z-y+x)}$	Y
Klosgen	$\sqrt{y-x}(1-z-\frac{x}{y})$	N	**added value**	$\max(1-z-\frac{x}{y}, \frac{y(1-z)-x}{z})$	Y
conviction	$\frac{y(1-z)}{x}$	Y	**one way support**	$(1-\frac{x}{y})\log\frac{y-x}{yz}$	N
J-measure	$(y-x)\log\frac{y-x}{yz}$	N	**Piatetsky-Shapiro**	$y(1-z) - x$	Y
cosine	$\frac{y-x}{\sqrt{yz}}$	Y	**Loevinger**	$1 - \frac{x}{y(1-z)}$	Y
information gain	$\log\frac{y-x}{yz}$	Y	**Sebag Shoenauer**	$\frac{y-x}{x}$	Y
contramin	$\frac{y-2x}{z}$	Y	**Bayesian factor**	$\frac{(y-x)(1-z)}{zx}$	Y
example-counter-example rate	$1 - \frac{x}{y-x}$	Y	**Zhang**	$\frac{y(1-z)-x}{\max((y-x)(1-z), zx)}$	Y
Kappa	$2\frac{y(1-z)-x}{y+z-2yz}$	Y	**Interestingness Weighting Dependency**	$\left(\left(\frac{y-x}{y\cdot z}\right)^k - 1\right) \times (y-x)^m$	N

adapted to m, $\phi_m(x, y', z) \leq \phi_m(x, y, z)$. Therefore, ϕ_m does increase with the second variable.

4.2 Classification of the Measures

We thus now have a NSC for the optimonotony. This condition is applied to 32 different measures. The results are presented in table 3. These results show that 26 measures are compatible with the optimonotony, making this pruning strategy very interesting.

Example 2. In table 3 we can notice that, in contrary to the original theorem 1 [20], Klosgen measure K doesn't verify the property. A 3-attributes-database containing the three following transactions: $\{\mathtt{P}, \mathtt{X}\}$, $\{\mathtt{P}, \mathtt{c}\}$, $\{\mathtt{c}\}$ gives a counterexample. We have $\mathbb{P}(\mathtt{P}\mathtt{X}\neg\mathtt{c}) = \mathbb{P}(\mathtt{P}\neg\mathtt{c})$ but $K(\mathtt{P} \rightarrow \mathtt{c}) = -0.096 < 0$ and $K(\mathtt{P}\mathtt{X} \rightarrow \mathtt{c}) = 0$ i.e. $K(\mathtt{P}\mathtt{X} \rightarrow \mathtt{c}) \not\leq K(\mathtt{P} \rightarrow \mathtt{c})$.

5 Conclusion

Measures may possess many algorithmic properties, that have not been exploited. We here propose a formal framework for the analytical study of measures and use it to study the optimonotony property we define. We then demonstrate a NSC for optimonotony. Finally we show that many measures are optimonotone and thus could be used with the underlying pruning strategy.

References

1. Agrawal, R., Imieliski, T., Swami, A.: Mining association rules between sets of items in large databases. In: ICMD, pp. 207–216 (1993)
2. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: KDD, pp. 80–86 (1998)
3. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: VLDB, pp. 478–499 (1994)
4. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: ICMD, pp. 1–12 (2000)
5. Goethals, B.: Frequent set mining. In: The Data Mining and Knowledge Discovery Handbook, pp. 377–397 (2005)
6. Cohen, E., Datar, M., Fujiwara, S., Gionis, A., Indyk, P., Motwani, R., Ullman, J.D., Yang, C.: Finding interesting associations without support pruning. In: TKDE, pp. 64–78 (2001)
7. Koh, Y.S.: Mining non-coincidental rules without a user defined support threshold. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS, vol. 5012, pp. 910–915. Springer, Heidelberg (2008)
8. Tan, P.N., Kumar, V., Srivastava, J.: Selecting the right objective measure for association analysis. In: IS, pp. 293–313 (2004)
9. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: A survey. ACM Computing Surveys 38(3, Article 9) (2006)

10. Lenca, P., Meyer, P., Vaillant, B., Lallich, S.: On selecting interestingness measures for association rules: user oriented description and multiple criteria decision aid. In: EJOR, pp. 610–626 (2008)
11. Li, W., Han, J., Pei, J.: CMAR: Accurate and efficient classification based on multiple class-association rules. In: ICDM, pp. 369–376 (2001)
12. Pei, J., Han, J.: Can we push more constraints into frequent pattern mining? In: KDD, pp. 350–354 (2000)
13. Bayardo Jr., R.J., Agrawal, R., Gunopulos, D.: Constraint-based rule mining in large, dense databases. In: ICDE, pp. 188–197 (1999)
14. Xiong, H., Tan, P.N., Kumar, V.: Mining strong affinity association patterns in data sets with skewed support distribution. In: ICDM, pp. 387–394 (2003)
15. Omiecinski, E.: Alternative interest measures for mining associations in databases. In: TKDE, pp. 57–69 (2003)
16. Wang, K., He, Y., Cheung, D.W.: Mining confident rules without support requirement. In: IKM, pp. 89–96 (2001)
17. Zimmermann, A., De Raedt, L.: CorClass: Correlated association rule mining for classification. In: Suzuki, E., Arikawa, S. (eds.) DS 2004. LNCS, vol. 3245, pp. 60–72. Springer, Heidelberg (2004)
18. Morishita, S., Sese, J.: Transversing itemset lattices with statistical metric pruning. In: PODS, pp. 226–236 (2000)
19. Bonchi, F., Lucchese, C.: Pushing tougher constraints in frequent pattern mining. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS, vol. 3518, pp. 114–124. Springer, Heidelberg (2005)
20. Li, J.: On optimal rule discovery. In: TKDE, pp. 460–471 (2006)
21. Hébert, C., Crémilleux, B.: A unified view of objective interestingness measures. In: Perner, P. (ed.) MLDM 2007. LNCS, vol. 4571, pp. 533–547. Springer, Heidelberg (2007)
22. Lallich, S., Vaillant, B., Lenca, P.: Parametrised measures for the evaluation of association rule interestingness. In: AMSDA, pp. 220–229 (2005)
23. Lallich, S., Vaillant, B., Lenca, P.: A probabilistic framework towards the parameterization of association rule interestingness measures. In: MCAP, pp. 447–463 (2007)
24. Li, J., Chee Fu, A.W., He, H., Chen, J., Jin, H., McAullay, D., Williams, G., Sparks, R., Kelman, C.: Mining risk patterns in medical data. In: KDD, pp. 770–775 (2005)

Discovering Action Rules That Are Highly Achievable from Massive Data

Einoshin Suzuki

Department of Informatics, ISEE, Kyushu University
suzuki@i.kyushu-u.ac.jp

Abstract. In this paper, we propose a novel algorithm which discovers a set of action rules for converting negative examples into positive examples. Unlike conventional action rule discovery methods, our method AARUDIA (Achievable Action RUle DIscovery Algorithm) considers the effects of actions and the achievability of the class change for disk-resident data. In AARUDIA, effects of actions are specified using domain rules and the achievability is inferred with Naive Bayes classifiers. AARUDIA takes a new breadth-first search method which manages actionable literals and stable literals, and exploits the achievability to reduce the number of discovered rules. Experimental results with inflated real-world data sets are promising and demonstrate the practicality of AARUDIA.

1 Introduction

Suppose a census data set of which positive and negative classes correspond to income no less than 50 K dollars per year and the rest, respectively (i.e. adult.data [2]). We try to discover action rules each of which suggests actions for changing the negative class into positive such as obtaining a bachelor degree and becoming an employee of the federal government.

Conventional methods [6,9,10,11,12,13,15,16] either neglect or unsatisfactorily deal with the achievability of an action rule i.e. how many examples are likely to succeed in making the class change. Firstly, the action rule is less useful if a condition "native-country = Trinadad and Tobago" is added in the premise as there are only 25 negative examples which correspond to the rule among 37,154 negative examples. Secondly, the action rule is also less useful if most of the negative examples who can obtain a bachelor degree and become an employee of the federal government have many disadvantages which would prevent the class change. Thirdly, the action rule is less useful if most of the negative examples are 57 years old, it takes 4 years to obtain a bachelor degree, and the age of retirement for an employee of the federal government is 60. In short, negative examples associated with a promising action rule should be large in number, should become similar to the positive examples once the actions are accomplished, and should not violate the domain rules in their class change. From the viewpoint of rule discovery, a discovered rule suggesting actions for the class change is likely to be

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 713–722, 2009.
© Springer-Verlag Berlin Heidelberg 2009

of little use if the number of examples which would succeed in the class change by taking the actions is small.

We argue that a discovered action rule should have enough achievability i.e. it should have a sufficient number of negative examples which can make the class change. We propose AARUDIA (Achievable Action RUle DIscovery Algorithm), which discovers a set of action rules each of which is associated with a set of relevant examples, as a practical solution. To realize the motivation, we include a set of domain rules which describe the effect of actions and a minimum achievability threshold in the input, and associate a set of relevant examples to a discovered action rule in the output. Our method judges the relevance of an example based on the Naive Bayes classifier, which is associated to each candidate of a discovered action rule.

2 Discovering Action Rules That Are Achievable

2.1 Action Rule

A table format data T consists of n examples $e_1, e_2, \ldots, e_n$, each of which is described with m attributes $f_1, f_2, \ldots, f_m$. An attribute is either nominal, ordinary, or continuous and not necessary binary. An ordinary attribute assumes a total order among its values unlike a nominal attribute and its number of possible values is small unlike a continuous attribute. Unlike association rule discovery, we do not assume a specific type of distribution on attribute values.

An attribute is classified as either actionable or stable. One can change the value v_1 of an actionable attribute f to another value v_2 through action e.g. "Sales" for "occupation" can be changed to "Exec-managerial". The value of a stable attribute remains the same e.g. "native-country = United-States" remains the same for a person.

We assume that the table format data contains a class attribute and one of its values represents a positive class. Table 1 shows an example of the table format data, which consist of three examples and have five attributes. Age, Education, and Capital loss are ordinary attributes while Income is a class attribute and Income $\geq$ 5K is the positive class.

We call a condition in the form of "an attribute = a value" an atom and we define that a literal represents either a single atom or a conjunction of atoms. An atom is called actionable or stable if its attribute is actionable or stable, respectively. A literal which has no actionable attribute is called a stable literal. A non-stable literal is called an actionable literal.

Table 1. Example data set

Age	Occupation	Education	Capital loss	Native country	Income
32	Exec-manegerial	Master	0-1K	US	$\geq$5K
53	Exec-manegerial	7th	0-1K	India	<5K
27	Clerk	Bachelor	1-10K	Japan	$\geq$5K

An action rule is a rule $Y \rightarrow \mathbb{P}$ where Y is an actionable literal and $\mathbb{P}$ represents "class = positive". For instance, "native country = US, education level = Bachelors $\rightarrow$ $\mathbb{P}$" suggests that those who were born in the U.S. and belong to the negative class had better change their education level to bachelor to turn into the positive class. A negative example which has a good possibility to satisfy Y by changing its value(s) of actionable attribute(s) and become a positive example is called relevant to the action rule. Let the new state of a negative example e by performing such a change be e', we call the class change achievable with the action rule for e if e satisfies the stable atoms in Y, e' satisfies the actionable atoms in Y, and the classifier γ associated with the action rule predicts e' as positive. In this case, we also call e to be relevant to $Y \rightarrow \mathbb{P}$.

We define that an action rule $Y \rightarrow \mathbb{P}$ satisfies the minimum support condition (1) for a minimum support threshold θ and the minimum confidence condition (2) for a minimum confidence threshold η as in association rule discovery [1], where $n(\mathbb{P}, Y)$ and $n(Y)$ are the number of positive examples which satisfy Y and the number of examples which satisfy Y, respectively.

$$\frac{n(\mathbb{P}, Y)}{n} \geq \theta \tag{1}$$

$$\frac{n(\mathbb{P}, Y)}{n(Y)} \geq \eta \tag{2}$$

The left-hand sides of (1) and (2) are called the support and the confidence of $Y \rightarrow \mathbb{P}$ and are denoted with $sup(Y)$ and $conf(Y)$, respectively. A crucial difference to association rule discovery is the minimum achievability condition (3) for a minimum achievability threshold ξ, where $a(Y \rightarrow \mathbb{P})$ represents the number of relevant examples to $Y \rightarrow \mathbb{P}$.

$$\frac{a(Y \rightarrow \mathbb{P})}{n} \geq \xi \tag{3}$$

We call the left-hand side of (3) the achievability of an action rule $Y \rightarrow \mathbb{P}$ and denote it with $ach(Y)$. We call an action rule which satisfies (1), (2), and (3) highly achievable.

[Action rule discovery problem]: given a table format data T, the values of thresholds for the minimum support θ, the minimum confidence η, and the minimum achievability ξ, find all action rules $Y \rightarrow \mathbb{P}$ that satisfy (1), (2), (3).

2.2 Domain Rules

A domain rule describes either the kind of an attribute, the change of values of an actionable attribute, and the number of bins in discretizing a continuous attribute. The last kind of rules are obvious thus we explain the first two.

For the kind of an attribute, we consider "ordered actionable" and "partially actionable" in addition to "actionable" and "stable" in the previous section. The first two kinds are special types of actionable attributes and are invented

to facilitate a specification of changes of an actionable attribute. By specifying that an attribute is an ordered actionable attribute, any change of values in the wrong direction is prohibited. For instance, the final education level is an ordered actionable attribute because its values follow a total order (e.g. "10th" $<$ "11th" $<$ "12th" $<$ "HS-grad") and a value is only allowed to increase. On the other hand, by specifying that an attribute is a partially actionable attribute, any change of values not specified explicitly is prohibited. For instance, the marital status is a partially actionable attribute because some of the recommendations such as divorce to a married person are inappropriate and thus are not specified.

For a change of values of an actionable attribute, we specify the old value, the new value, the cost of the change, the effects of the change, and the condition of the change. We do not allow multiple possibilities for a change of values for the same condition. Conflicts between effects of actions are handled by user-specified rules. For an old value or a new value, the user can specify either one value, a set of values, or a wild card which matches any value. For the cost of a change, we allow either "easy", "normal", or "difficult" for ease of specification. These costs are ordinary and cannot be added. We admit that a quantitative index can be more accurate but believe that it is not available in most domains. For the effects of a change, the user can specify a list of (attribute, new value)s. Here a new value is either an attribute value, an addition/deletion of a value (e.g. age 5 years older), or an average value of the attribute (e.g. working hour = the average value of the successful executive manegerials). For the condition of a change, we allow a literal which specifies a set of examples.

We assume that an example follows the recommended action if the class change is achievable. The suggestion made by our discovered rule can be called individual since each example consults the classifier and domain rules before taking the suggested action. Most of previous research [6,9,10,11,12,13,15,16] assume that all examples that can take the suggested action (with affordable cost if any but consulting neither a classifier nor domain rules) follow the suggestion thus can be called total. We believe that our formalization fits the reality unlike total as one would not follow a suggestion if the class change seems impossible.

3 Achievable Action Rule Discovery Algorithm

3.1 Motivation and Overview

AARUDIA finds all actionable literals Y that satisfy (1) à la Apriori [1] then outputs all $Y \rightarrow \mathbb{P}$ that satisfy (2) and (3) from disk-resident data. AARUDIA employs the Naive Bayes classifier [4], which considers both positive examples and negative examples and is effective even for high-dimensional data, to each actionable literal, which represents a candidate of an action rule. The Naive Bayes classifier can be constructed with one scan of the data, requires only $(2\mu+2)$-double-precision-number space where μ represents the number of possible atoms, and is robust against noise. Many experiments show its high accuracy despite of its simplicity and its assumption on the independence of attributes [4].

It is well-known that the support of an itemset follows the anti-monotonicity property unlike the confidence [1]. The achievability on the other hand does not follow the anti-monotonicity property because adding an actionable atom α to a premise Y may add some of the examples each of which satisfies $\overline{\alpha} \wedge Y$ into the set of relevant examples i.e. $a(\alpha Y \rightarrow \mathbb{P})$ can be greater than $a(Y \rightarrow \mathbb{P})$.

The above analysis has led us invent an algorithm which handles the set of actionable literals and the set of stable literals separately. The set of actionable and stable literals with d atoms are denoted with $arl[d]$ and $nrl[d]$, respectively. The algorithm requires $2 + 2\nu$ scans of the disk-resident data where ν represents the maximum length of investigated literals. The number of scans is nearly twice of Apriori but we will show its practical performance in the next section.

```
algorithm AARUDIA
    T' = globalDiscretization(T) // 2 scans
    d = 1
    do // 2 scans for each loop
      obtainRulesCandidates(d, arl[d], nrl[d], res[d], T', θ, η)
      printAchievableActionRules(res[d], ξ, T')
      d = d + 1
    while((arl[d − 1] ≠ ∅ or nrl[d − 1] ≠ ∅) and d < m)
```

```
                        actionable literal                  stable literal
rule 1:
 workclass = Private, education = Bachelors, race = White, native country = US -> class = positive
rule 2:
 workclass = Private, education = Bachelors, sex = Male, native country = US -> class = positive
example 1:
  55,  Private,   Some-college, Married-civ-spouse, Sales,      Husband,      White,
 (age, workclass, education,    marital status,     occupation, relationship, race)
  Male, 1877,        425,          43,              United-States,  negative
 (sex,  capital gain,capital loss, hours per week, native country, class)
```

naive Bayes classifier for rule 1
Pr(class=positive)= 0.152, Pr(class=negaitve)= 0.847

	age	workclass	education	marital status	occupation	relationship	race
Pr(class=positive|att=val)	0.630	0.996	0.993	0.883	0.145	0.814	0.998
Pr(class=negative|att=val)	0.432	0.999	0.998	0.544	0.144	0.512	0.999

	sex	capital gain	capital loss	hours per week	native country
Pr(class=positive|att=val)	0.910	0.694	0.305	0.757	0.981
Pr(class=negative|att=val)	0.779	0.880	0.174	0.586	0.996

0.152*0.630*0.996*0.993*0.883*0.145*0.814*0.998*0.910*0.694*0.305*0.757*0.981
  > 0.847*0.432*0.999*0.998*0.544*0.144*0.512*0.999*0.779*0.880*0.174*0.586*0.996
→ classified as positive

naive Bayes classifier for rule 2
Pr(class=positive)= 0.172, Pr(class=negaitve)= 0.827

	age	workclass	education	marital status	occupation	relationship	race
Pr(class=positive|att=val)	0.634	0.996	0.992	0.895	0.143	0.855	0.993
Pr(class=negative|att=val)	0.446	0.999	0.998	0.603	0.148	0.599	0.987

	sex	capital gain	capital loss	hours per week	native country
Pr(class=positive|att=val)	0.999	0.695	0.304	0.770	0.980
Pr(class=negative|att=val)	0.999	0.875	0.180	0.622	0.995

0.172*0.634*0.996*0.992*0.895*0.143*0.855*0.993*0.999*0.695*0.304*0.770*0.980
  < 0.827*0.446*0.999*0.998*0.603*0.148*0.599*0.987*0.999*0.875*0.180*0.622*0.995
→ classified as negative

Fig. 1. Examples of how to judge whether an example is relevant to an action rule

AARUDIA first performs a global discretization for each continuous attribute to obtain a discretized data set T' then iterates loops in each of which the set of all action rules with a premise-length d is discovered and d is incremented. In each iteration of the loop, the sets of actionable and stable literals which correspond to $arl[d]$ and $nrl[d]$, respectively as well as the candidates $res[d]$ of the premises of discovered rules are obtained with obtainRulesCandidates. Then AARUDIA outputs all action rules of which length of the premise is d with printAchievableActionRules. The iteration continues at least one of $arl[d]$ and $nrl[d]$ is nonempty and $d < m$. We omit the details of the algorithm due to space constraint.

Figure 1 shows two examples of action rules, an example, the Naive Bayes classifiers associated with the two action rules, and their predictions to the example. Laplace estimates [4] are used for the conditional probabilities. The Naive Bayes classifier associated with action rules 1 and 2 judges example 1 positive and negative, respectively. Let $\mathbb{N}$ represent "class = negative". In the experiments in the next section, rule 1 had $n\Pr(\mathbb{P}Y) = 2{,}219$, $n\Pr(\mathbb{N}Y) = 12{,}362$, $a(Y \to \mathbb{P}) = 3{,}489$ (including this example) while rule 2 had $n\Pr(\mathbb{P}Y) = 2{,}030$, $n\Pr(\mathbb{N}Y) = 9{,}747$, $a(Y \to \mathbb{P}) = 2{,}080$ (not including this example) for $n = 1,000,000$. As we used $\theta = 0.01, \eta = 0.15, \xi = 0.03$, rule 1 satisfies (1), (2), (3) while rule 2 does not satisfy (3). Thus only rule 1 is output as the discovery result.

4 Experiments

4.1 Conditions

The Census data set (i.e. adult.data) [2] consists of 48,842 examples described with 15 attributes and the maximum number of values that an attribute can take is 42. We believe that the data is at the same time ideal and inadequate for testing AARUDIA because it is comprehensible but small. To circumvent this problem, we have "inflated" the data set by randomly generating "reasonable" examples. A reasonable example is generated by choosing a specified number (i.e. 5) of examples randomly then adopting the most frequent value for each non-continuous attribute and the average value for each continuous attribute. We have generated three inflated data sets of 100,000, 500,000, and 1,000,000 examples and call them 100K, 500K, and 1M, respectively.

In the experiments, age, capital-gain, and capital-loss are considered as continuous attributes. The attribute fnlwgt is ignored because it is determined by the sampling weight thus does not describe a property of the corresponding person. The attribute education-num was used to define the order of values for the education attribute then was deleted because it is just a numeric representation of the education attribute.

We specified workclass and occupation as actionable. Education and marital-status were specified as ordered actionable and partially actionable, respectively. Five domain rules for actions were provided based on a simple statistical analysis[1]. We neglected the cost information because it is an open problem and is not

[1] The five rules use wild cards, effects, and conditions so the possible actions are reasonably complex.

the main focus of our research. For the number of discretized values, we specified 3 for age and 2 for others. We used a workstation Dell Precision 690 with double 3.0GHz-CPUs and 16G byte memory. In the implementation, we used gcc 3.4.6 under Linux OS 2.6.9 and an optimization flag -O. The maximum number of the Naive Bayes classifiers were set to 65,536 though the workstation can hold even 1,048,576 Naive Bayes classifiers.

4.2 Results and Discussions

We show the results of the experiments for the data sets 1M, 500K, and 100K in the top, middle, and bottom rows of Figure 2, respectively. The figure consists of three kinds of plots, in which θ, η, and ξ are varied in the left, middle, and right plots, respectively. We have varied the range of θ for the data sets as 0.015, 0.016, ..., 0.024 for 1M, 0.01, 0.011, ..., 0.019 for 500K, and 0.005, 0.006, ..., 0.014 for 100K to keep the execution time reasonable. The values of η and ξ

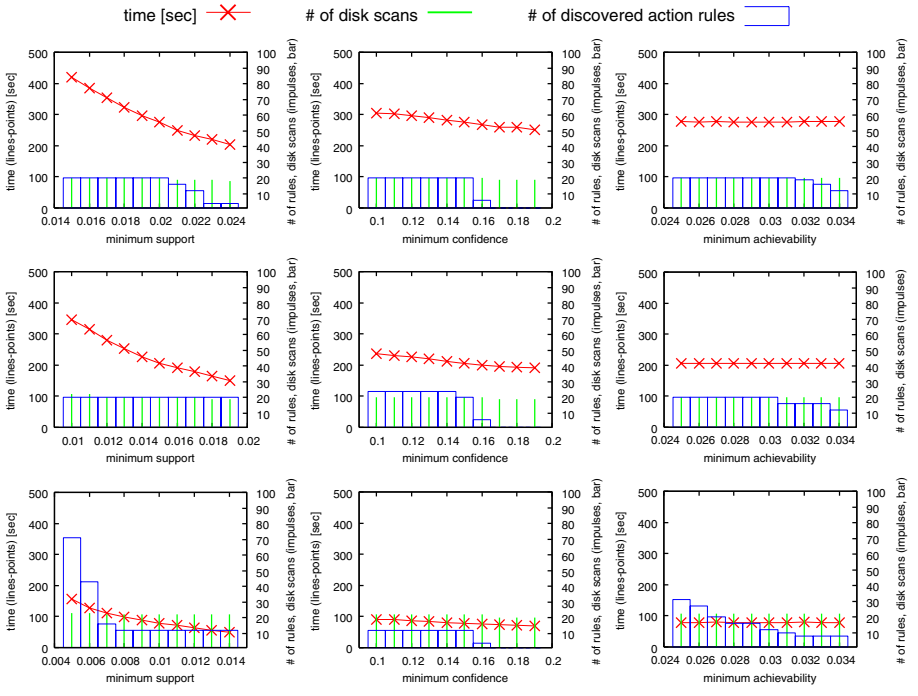


Fig. 2. Results for 1M, 500K, and 100K data in terms of minimum support, minimum confidence, and minimum achievability. The line graph, the bar graph, and the impulse graph correspond to the time of execution, the number of disk scans, and the number of discovered action rules, respectively.

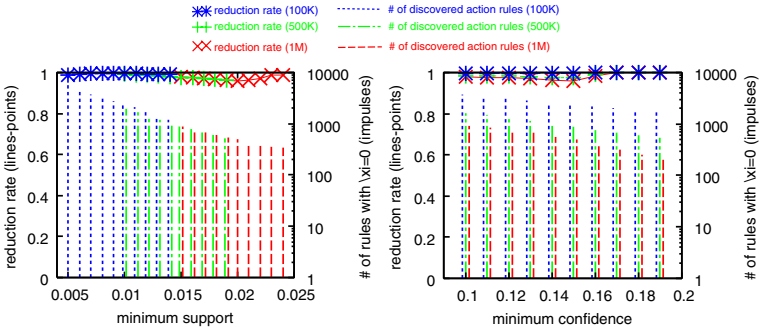


Fig. 3. Results of experiments for the effectiveness of achievability i.e. comparison of $\xi =$ (the standard value) and $\xi = 0$

were varied as 0.10, 0.11, ..., 0.19 and 0.025, 0.026, ..., 0.034, respectively. Note that ξ is not as small as it looks since for instance $\xi = 0.025$ for 1M means that the class change is achievable for at least 25,000 persons with an action rule. A standard condition was chosen for each data set: $(\theta, \eta, \xi) = (0.02, 0.15, 0.03)$ for 1M, $(\theta, \eta, \xi) = (0.015, 0.15, 0.03)$ for 500K, and $(\theta, \eta, \xi) = (0.01, 0.15, 0.03)$ for 100K. In each plot, we fixed the values of the remaining two parameters to those of the respective standard condition.

From the figure, we see that AARUDIA runs in a reasonable time: for 1M, 500K, and 100K, the execution times are between 204 and 419, 150 and 345, and 48 and 155 seconds, respectively. These results are related with the number of examples, the number of disk scans, which is typically about 20, the distributions of values in the data, and the values of the thresholds.

Unlike a typical rule discovery algorithm, AARUDIA discovers at most 71 rules thus the inspection of the discovered rules by the user is easy. A higher threshold value yields in a reduced execution time but a smaller number of discovered action rules, which is natural due to a more strict constraint on the discovered action rules. Under several combinations of threshold values, no action rules were discovered, which means that the threshold values were too high. In practice, the values of the thresholds may be determined with trial and error, which is common in a KDD process [5]. Most of the discovered action rules recommend actions in terms of education, workclass, and occupation, which seems reasonable. No action rules which recommend an action on marital status were discovered, which shows that AARUDIA allows inadequate domain rules[2].

Figure 3 shows the result of our investigation on the effectiveness of (3) in reducing the number of discovered action rules. The left and right plots show the results of varying θ and η, respectively, where $\xi = 0$ and the fixed threshold takes the value of the standard condition. The reduction rate is defined as 1 - (the number of discovered action rules with $\xi =$ (the standard value))/(the number of discovered action rules with $\xi = 0$). We see that (3) eliminates more than

[2] By common sense such action rules are inadequate in obtaining high income.

95 % of discovered action rules and without it their numbers become hundreds or thousands. Clearly the achievability condition (3) is useful in reducing the workload of the user in inspecting the discovered action rules. Note that each of the eliminated action rules has a small number of relevant examples thus are less worth of inspection.

5 Related Work

Raś et al.'s are the pioneers of the action rule mining [9,10,11,12,13,15,16] though they neglect the effect of an action and the achievability. They mostly use a heuristic method to discover a set of rules then to make pairs of a rule which predicts the positive class and a related rule which predicts the negative class. Unlike an exhaustive method such as ours, their method can miss important rules.

[17] also aims at converting individuals from an undesirable class to a desirable class. The work is based on the case-base reasoning and it states to "identify typical positive cases to form a small and highly representative case base, then use the case base as "role models" to formulate the marketing actions". As it discovers recommended actions to individuals instead of action rules, the recommendation lacks of generality and are easily influenced by outliers. They use three classifiers for finding the role model: 1-nearest neighbor, 1-nearest cluster centroid[3], and the support vector machines. The first and the second classifiers neglect negative examples and are ineffective for high-dimensional data due to the curse of dimensionality. The last classifier is prohibitive to disk-resident data and noisy data.

Though the discovered pattern of [7] is a deviation, the motivation of the work is related to ours as it states "the interestingness of a finding is the estimated benefit from a possible action connected to it". Consideration on the effects and the achievability of the class change are left to the user. It should be also noted that [7] relies on a DBMS system thus neglects inventing data mining algorithms for disk-resident data. [3] tries to obtain the set of products that yields the maximum cross-selling profits in the framework of association rule discovery. Though the work also considers the effects of actions and puts emphasis on decision making, it neglects the achievability, takes a totally different approach, and relies on accurate quantitative profits and costs.

6 Conclusions

In a practical data mining application, the achievability of the objective and the effect of the actions suggested by a data mining method are often crucial. Most of data mining methods leave the consideration on them to the user while this paper can be considered as an initial step toward supporting such activities.

[3] 1-nearest cluster centroid returns the class of the nearest cluster centroid as its prediction.

Acknowledgments

This work was partially supported by the grant-in-aid for scientific research on fundamental research (B) 18300047 from the Japanese Ministry of Education, Culture, Sports, Science and Technology.

References

1. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In: Proc. VLDB, pp. 487–499 (1994)
2. Blake, C., Merz, C.J., Keogh, E.: UCI Repository of Machine Learning Databases, `http://www.ics.uci.edu/~mlearn/MLRepository.html`
3. Brijs, T., Goethals, B., Swinnen, G., Vanhoof, K., Wets, G.: A Data Mining Framework for Optimal Product Selection in Retail Supermarket Data: The Generalized PROFSET Model. In: Proc. KDD, pp. 20–23 (2000)
4. Domingos, P., Pazzani, M.: On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. Machine Learning 29(2/3), 103–130 (1997)
5. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P.: From Data Mining to Knowledge Discovery: An Overview. In: Advances in Knowledge Discovery and Data Mining, pp. 1–34. AAAI/MIT Press, Menlo Park, Calif. (1996)
6. Jiang, Y., Wang, K., Tuzhilin, A., Fu, A.W.-C.: Mining Patterns that Respond to Actions. In: Proc. ICDM, pp. 669–672 (2005)
7. Piatetsky-Shapiro, G., Matheus, C.J.: The Interestingness of Deviations. In: AAAI 1994 Workshop on Knowledge Discovery in Databases, pp. 25–36 (1994)
8. Quinlan, J.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
9. Rás, Z.W., Gupta, S.: Global Action Rules in Distributed Knowledge Systems. Fundamenta Informaticae 51(1-2), 175–184 (2002)
10. Rás, Z.W., Tsay, L.-S.: Discovering Extended Action-Rules (System DEAR). In: Proc. International IIS, pp. 293–300 (2003)
11. Rás, Z.W., Tzacheva, A.A., Tsay, L.-S., Gürdal, O.: Mining for Interesting Action Rules. In: Proc. IAT, pp. 187–193 (2005)
12. Raś, Z.W., Wieczorkowska, A.: Action-Rules: How to Increase Profit of a Company. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS, vol. 1910, pp. 587–592. Springer, Heidelberg (2000)
13. Rás, Z.W., Wieczorkowska, A.: Mining for Action-Rules in Large Decision Tables Classifying Customers. Intelligent Information Systems, 55–63 (2000)
14. Russell, S., Norvig, P.: Artificial Intelligence. Prentice-Hall, Englewood Cliffs (1995)
15. Tsay, L.-S., Rás, Z.W., Wieczorkowska, A.: Tree-based Algorithm for Discovering Extended Action-Rules (System DEAR2). Intelligent Information Systems, 459–464 (2004)
16. Tzacheva, A.A., Rás, Z.W.: Action Rules Mining. International Journal of Intelligent Systems 20(6), 719–736 (2005)
17. Yang, Q., Cheng, H.: Mining Case Bases for Action Recommendation. In: Proc. ICDM, pp. 522–529 (2002)

Extracting Fuzzy Rules for Detecting Ventricular Arrhythmias Based on NEWFM

Dong-Kun Shin[1], Sang-Hong Lee[2], and Joon S. Lim[3,*]

[1] Division of Computer, Sahmyook University, Seoul 139-742, Korea
dkshin@syu.ac.kr
[2,3] Division of Software, Kyungwon University, Sungnam 461-701, Korea
shleedosa@kyungwon.ac.kr, jslim@kyungwon.ac.kr

Abstract. In the heart disease, the important problem of ECG arrhythmia is to discriminate ventricular arrhythmias from normal cardiac rhythm. This paper presents novel method based on the neural network with weighted fuzzy membership functions (NEWFM) for the discrimination of ventricular tachycardia (VT) and ventricular fibrillation (VF) from normal sinus rhythm (NSR). This paper uses two pre-processes, the Haar wavelet function and extraction feature method are carried out in order. By using these methods, six features can be generated, which are the input data of NEWFM. NEWFM classifies NSR and VT/VF beats by the trained bounded sum of weighted fuzzy membership functions (BSWFMs) using six input features from the Creighton University Ventricular Tachyarrhythmia Data Base (CUDB). The results are better than Amann's phase space reconstruction (PSR) algorithm, accuracy and specificity rates of 90.4% and 93.3%, respectively.

Keywords: Fuzzy Neural Networks, NSR, VT, VF, Wavelet Transform.

1 Introduction

Detecting of cardiac rhythm abnormalities are based on adaptive decision support tool and analysis of time series of the electrocardiograms (ECG). Support vector machines (SVM) and fuzzy neural networks (FNN) can be effectively used for this type of tool as a major pattern classification and predictive rule generation tool for cardiac pattern analysis [1] [2] [3] [4]. Since the ECG signal includes noise, baseline drift, and abnormal behavior, the Wavelet Transforms (WT) as filtering process is needed.

Ventricular tachycardia (VT) is a potentially lethal disruption of normal heartbeat (arrhythmia) that may cause the heart to become unable to pump adequate blood through the body. If the VT appears a period of time, VT will induce a Ventricular Fibrillation (VF), the most dangerous type of heart arrhythmia. The VF is represented by fast rhythm, abnormal and ineffective contractions of the ventricles and it finishes in a systole. The VF within a few minutes or a few days will lead to cardiac sudden

* Corresponding author.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 723–730, 2009.
© Springer-Verlag Berlin Heidelberg 2009

death. The survival probability for a human who has a VF attack outside the hospital ranges between 2-25% [5].

In this paper, six input features are obtained from two steps. In the first step, ECG beasts are transformed by Haar wavelet function, and then 256 coefficients of d3 at levels 3 are obtained. In the second step, the first input feature are obtained by phase space reconstruction (PSR) method which is based on detail coefficients d3 at level 3, the second feature are obtained by extraction peaks number (EPN) method which is based on detail coefficients d3 at level 3. The other input features are obtained by extraction characteristic features (ECF) method. By using these methods, 6 features can be generated which are the input data of NEWFM.

Amann used filtering transformed and the threshold d_0 by PSR algorithm, as characteristic features for VF classification with 85.1% of VF classification accuracy [6]. This paper is shown that the 6 generalized features are extracted on the data sets used by Creighton University Ventricular Tachyarrhythmia Data Base (CUDB) with reliable accuracy and specificity rates of 90.4% and 93.3%, respectively.

2 Pre-process of ECG Signals

2.1 Wavelet Transform

In this paper, the Haar wavelet function is used as a mother wavelet. Detail 256 coefficients d3 at levels 3 are selected for PSR, EPN, and ECF method, since high-frequency inference artifacts in level 1 and 2, and baseline variations in level 5 and 6 may have an influence on the extracting characteristics of ECG signal. The sampling size of the window of ECG signals from the 250 Hz CUDB is 8s.

2.2 Extraction Input Features

The phase space reconstruction (PSR) method is based on reconstruct the phase space [6]. In the paper of Amann, there is a detailed explanation of PSR algorithm. Used this method can identify signals which a dynamic law or random behavior. Calculating a measure d is defined by formula (1): [6]

$$d = \frac{number\ of\ visited\ boxes}{number\ of\ all\ boxes} \tag{1}$$

Fig. 1 shows typical NSR, VF, VT 8s episodes from the CUDB and these signals by wavelet transform are illustrated in Fig. 2 and the corresponding phase space plot is illustrated in Fig. 3. So NSR from VT/VF is differentiated by the d value of visited boxes in this paper. The d value is used as the NEWFM first input feature.

The extraction peaks number (EPN) method is based on detail coefficients d3 at level 3 of Haar wavelet transform. Calculating the number of the peak points whose value are greater than the average of all data as NEWFM second input feature in Fig. 4.

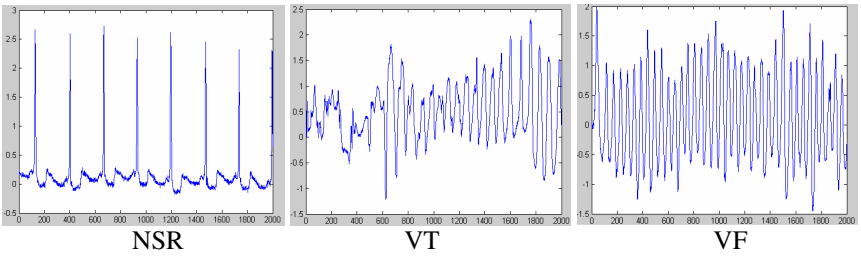


Fig. 1. NSR, VT, VF 8s episodes in the ECG signal from the CUDB

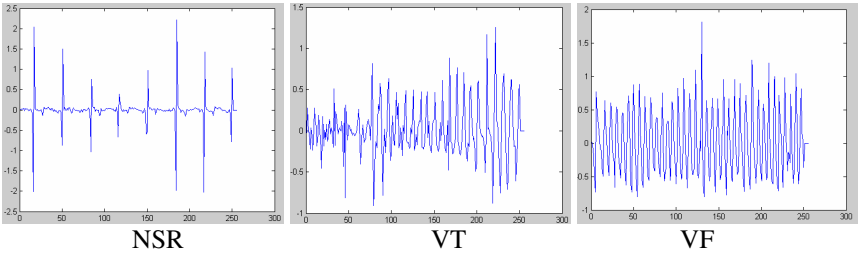


Fig. 2. NSR, VT, VF 8 sec episodes by wavelet transforms

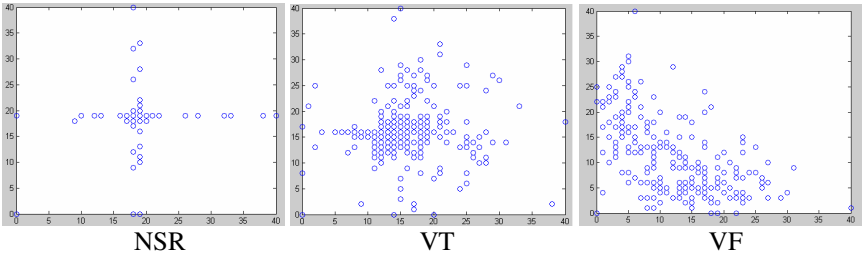


Fig. 3. NSR, VT, VF 8 sec episodes visited boxes visualized

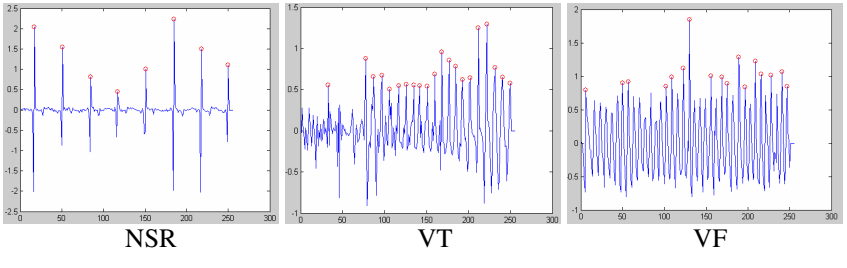


Fig. 4. The peaks of NSR, VT, VF are recognized

Extraction characteristic features (ECF) of front sides of the maximum peak with detail coefficients d3. In this method, sort the peaks in extracted peaks result of d3 by values, then select 4 coefficients (b1, b2, b3, b4) before first 3 maximum peaks, and

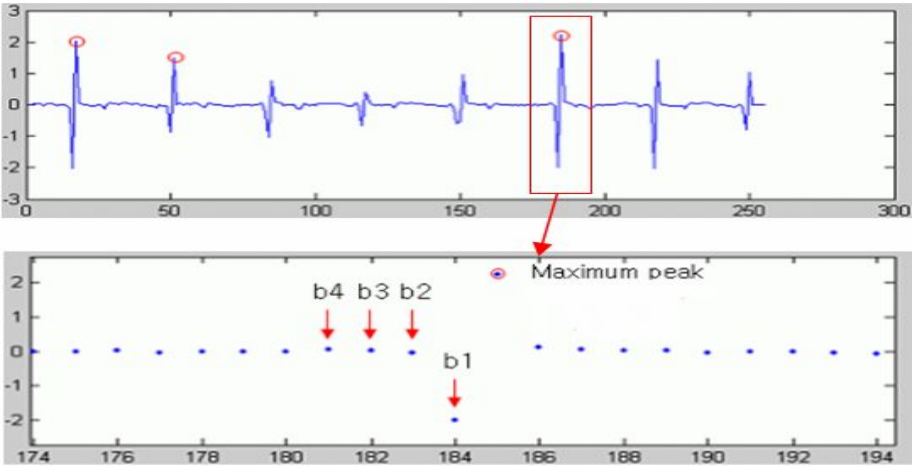


Fig. 5. For example, the first 3 maximum peaks of NSR are recognized and select 4 coefficients before the maximum peak

calculate the average of them(AV_{b1}, AV_{b2}, AV_{b3}, and AV_{b4}). Fig. 5 shows first 3 maximum peaks of the NSR sample detail coefficients d3 are recognized. And then selects 4 coefficients (b1, b2, b3, b4) before the maximum peak.

3 Neural Networks with Weighted Fuzzy Membership Function (NEWFM)

3.1 The Structure of NEWFM

Neural network with weighted fuzzy membership function (NEWFM) is a supervised classification neuro-fuzzy system using bounded sum of weighted fuzzy membership functions (BSWFM in Fig. 7) [7][2]. The structure of NEWFM, illustrated in Fig. 6, comprises three layers namely input, hyperbox, and class layer. The input layer contains n input nodes for an n featured input pattern. The hyperbox layer consists of m hyperbox nodes. Each hyperbox node B_l to be connected to a class node contains n BSWFMs for n input nodes. The output layer is composed of p class nodes. Each class node is connected to one or more hyperbox nodes. An hth. Input pattern can be recorded as I_h={A_h=(a_1, a_2, … , a_n), $class$}, where class is the result of classification and A_h is n features of an input pattern.

The connection weight between a hyperbox node B_l and a class node C_i is represented by w_{li}, which is initially set to 0. From the first input pattern I_h, the w_{li} is set to 1 by the winner hyperbox node B_l and class i in I_h. C_i should have one or more than one connections to hyperbox nodes, whereas B_l is restricted to have one connection to a corresponding class node. The B_l can be learned only when B_l is a winner for an input I_h with class i and w_{li} = 1.

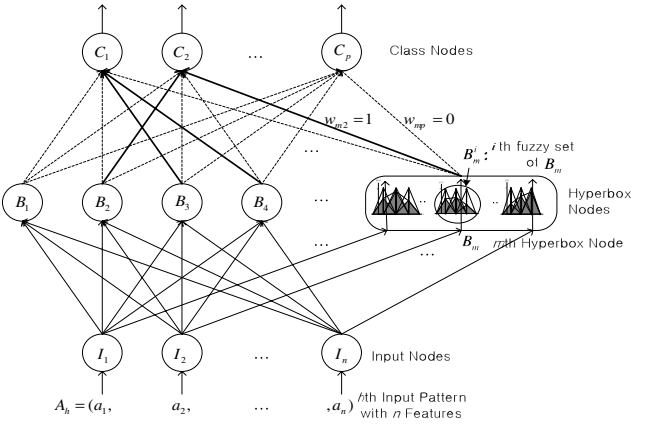


Fig. 6. Structure of NEWFM

3.2 Learning Scheme

A hyperbox node B_l consists of n fuzzy sets. The ith fuzzy set of B_l, represented by B_l^i, has three *weighted fuzzy membership functions* (WFM, grey triangles $\omega_{l1}^i, \omega_{l2}^i$, and ω_{l3}^i in Fig. 7) which randomly constructed before learning. Each ω_{lj}^i is originated from the original membership function μ_{lj}^i with its weight W_{lj}^i in the Fig. 7. The *bounded sum of three weighted fuzzy membership functions* (BSWFM is depicted by the bold line in Fig. 7) of B_l^i combines the fuzzy characteristics of the three WFMs. The BSWFM value of B_l^i, denoted as $BS_l^i(.)$, and is calculated by formula (2) where a_i is an ith feature value of an input pattern A_h for B_l^i.

$$BS_l^i(a_i) = \sum_{j=1}^{3} \omega_{lj}^i(a_i), \tag{2}$$

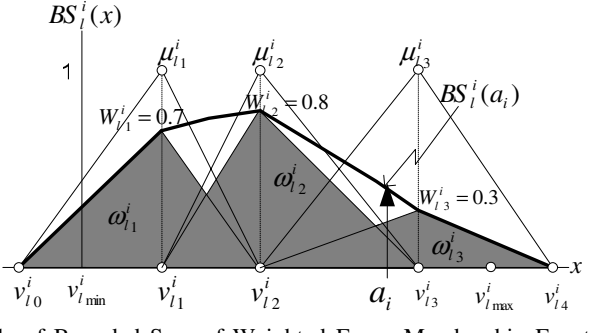


Fig. 7. An example of Bounded Sum of Weighted Fuzzy Membership Functions (BSWFM, bold line) of B_l^i and $BS_l^i(a_i)$

The winner hyperbox node B_l is selected by the *Output* (B_l) operator. Only the B_l, that has the maximum value of *Output* (B_l) for an input I_h wtih class i and w_{li} = 1, among the hyperbox nodes can be learned. For the hth input A_h= (a_1, a_2… a_n) with n features to the hyperbox B_l, output of the B_l is obtained by formula (3)

$$Output\ \ (B_l) = \frac{1}{n}\sum_{i=1}^{n} BS_l^i(a_i). \tag{3}$$

Then, the selected winner hyperbox node B_l is learned by the *Adjust* (B_l) operation. This operation adjusts all B_l^is according to the input a_i, where i=1, 2… n. The membership function weight W_{lj}^i (where $0 \le W_{lj}^i \le 1$ and j=1, 2, 3) represents the strength of ω_{lj}^i. Then a WFM ω_{lj}^i can be formed by ($v_{lj-1}^i, W_{lj}^i, v_{lj+1}^i$). As a result of *Adjust* (B_l) operation, the vertices v_{lj}^i, and weights W_{lj}^i in Fig. 8, are adjusted by formulas (4) expressions:

$$\begin{aligned} v_{lj}^i &= v_{lj}^i + s \times \alpha \times E_{lj}^i \times \omega_{lj}^i(a_i) \\ &= v_{lj}^i + s \times \alpha \times E_{lj}^i \times \mu_{lj}^i(a_i) \times W_{lj}^i, where \end{aligned}$$

$$\begin{cases} s=-1, E_{lj}^i = \min(\left|v_{lj}^i - a_i\right|, \left|v_{lj-1}^i - a_i\right|), \ if\ v_{lj-1}^i \le a_i < v_{lj}^i \\ s=1, E_{lj}^i = \min(\left|v_{lj}^i - a_i\right|, \left|v_{lj+1}^i - a_i\right|), \ if\ v_{lj}^i \le a_i < v_{lj+1}^i \\ E_{lj}^i = 0, \ otherwise \end{cases} \tag{4}$$

$$W_{lj}^i = W_{lj}^i + \beta \times (\mu_{lj}^i(a_i) - W_{lj}^i)$$

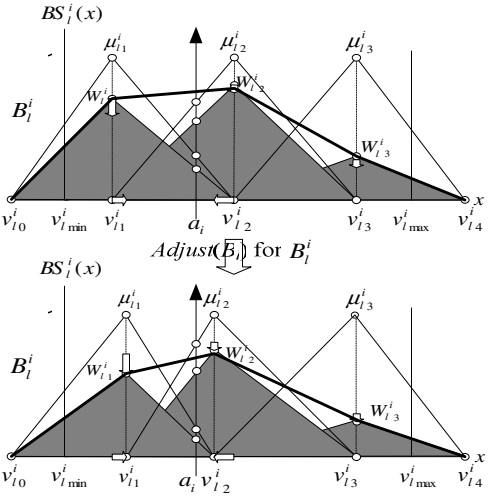


Fig. 8. An example of before and after adjust (B_l) operation for B_l^i

Where the α and β are the learning rates for $v_{l\,j}^{i}$ and $W_{l}^{i}{}_{j}$ respectively in the range from 0 to 1 and j=1,2,3.

Fig. 8 shows BSWFMs before and after *Adjust* (B_l) operation for B_l^i with an input a_i. The weights and the centers of membership functions are adjusted by the *Adjust* (B_l) operation, e.g., $W_{l\,1}^{i}$, $W_{l\,2}^{i}$, and $W_{l\,3}^{i}$ are moved down, v_{l1}^{i} and $v_{l\,2}^{i}$ are moved toward a_i, and $v_{l\,3}^{i}$ remains in the same location.

The Adjust (B_l) operations are executed by a set of training data. If the classification rate for a set of test data is not reached to a goal rate, the learning scheme with Adjust(B_l) operation is repeated from the beginning by randomly reconstructing all WFMs in B_l s and making all connection weights to 0 (w_{li} = 0) until the goal rate is reached.

4 Experimental Results

The sampling data sets are taken from the CUDB. This data base consists of thirty-five files, and one channel per file which is with 508 seconds long is stored in one file. The sampling size of the window of ECG signals from the 250 Hz CUDB is 8s. No pre-selection of certain ECG signals was made to simulate the case of an onlooker more accurately [6].

The graphs in Fig. 9 are the BSWFMs of the six input features: the 1st feature is *d* value, 2nd feature is peaks number, 3rd-6th features which are coefficients average of front side of the maximum peak with detail coefficients d3. The solid lines and dotted lines visually represent NSR, VT/VF characteristics of ECG visually, which enables the features to interpret explicitly.

Table 1 shows the data set used to evaluate the performance of the NEWFM model is the Amann data set of the CUDB. The sensitivity rates, specificity rates, positive productivity rates, and accuracy rates of NSR, VF/VT is 79.1%, 93.3%, 75.7%, and 90.36%, respectively.

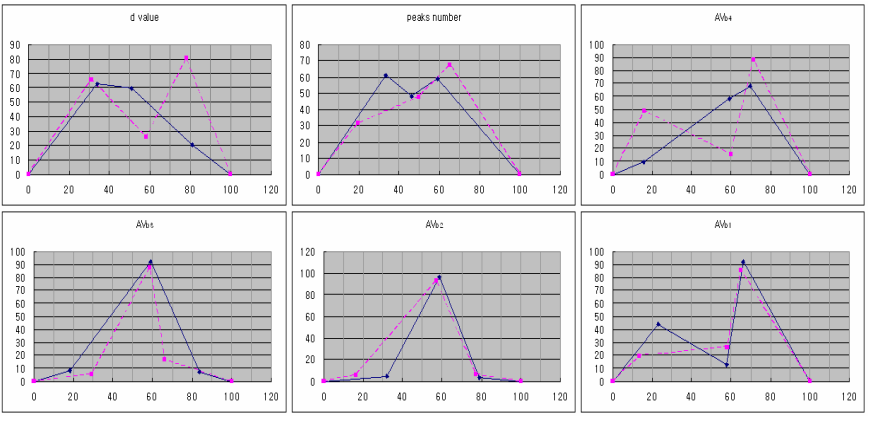


Fig. 9. Trained BSWFMs of the 1- 6th Features for NSR, VT/VF Classification

Table 1. The results of evaluating the NSR, VT/VF detection algorithm: sensitivity (Se), specificity (Sp), positive productivity (Pp), accuracy (Ac)

algorithm	Se	Sp	Pp	Ac
Amann	70.2	89.3	65.0	5
NEWFM	79.1	93.3	75.7	90.36

5 Concluding Remarks

The BSWFMs of the six features trained by NEWFM are shown visually, which makes the features interpret explicitly. Since each BSWFM combines multiple weighted fuzzy membership functions into one using bounded sum, the six small-sized BSWFMs can realize real-time NSR, VT/VF detection in mobile environment. These algorithms are pivotal component in Automated External Defibrillators (AED).

References

1. Engin, M.: ECG beat classification using neuro-fuzzy network. Pattern Recognition Letters 25, 1715–1722 (2004)
2. Lim, J.S., Wang, D., Kim, Y.-S., Gupta, S.: A neuro-fuzzy approach for diagnosis of antibody deficiency syndrome. Neurocomputing 69(7-9), 969–974 (2006)
3. Minami, K., Nakajima, H., Toyoshima, T.: Real-Time Discrimination of Ventricular Tachyarrhythmia with Fourier-Transform Neural Network. IEEE Trans. on Biomedical Engineering 46(2), 176–185 (1999)
4. Osowski, S., Linh, T.H.: ECG beat recognition using fuzzy hybrid neural network. IEEE Trans. on Biomedical Engineering 48(4), 1265–1271 (2001)
5. Fernandez, A.R., Folgueras, J., Colorado, O.: Validation of a set of algorithms for ventricular fibrillation detection: experimental results. In: Proceedings of the 25th Annual International Conference of the IEEE, September 17-21, vol. 3, pp. 2885–2888 (2003)
6. Amann, A., Tratnig, R., Unterkofler, K.: Detecting Ventricular Fibrillation by Time-Delay Methods. IEEE Trans. on Biomedical Engineering 54(1), 174–177 (2007)
7. Lim, J.S., Ryu, T.-W., Kim, H.-J., Gupta, S.: Feature Selection for Specific Antibody Deficiency Syndrome by Neural Network with Weighted Fuzzy Membership Functions. In: Wang, L., Jin, Y. (eds.) FSKD 2005. LNCS, vol. 3614, pp. 811–820. Springer, Heidelberg (2005)

Trace Mining from Distributed Assembly Databases for Causal Analysis

Shohei Hido[1], Hirofumi Matsuzawa[2],
Fumihiko Kitayama[2], and Masayuki Numao[3]

[1] IBM Research, Tokyo Research Laboratory
hido@jp.ibm.com
[2] Global Business Services, IBM Japan
{matuzawa,ktym}@jp.ibm.com
[3] Department of Computer Science, The University of Electro-Communications
numao@cs.uec.ac.jp

Abstract. Hierarchical structures of components often appear in industry, such as the components of cars. We focus on association mining from the hierarchically assembled data items that are characterized with identity labels such as lot numbers. Massive and physically distributed product databases make it difficult to directly find the associations of deep-level items. We propose a top-down algorithm using virtual lot numbers to mine association rules from the hierarchical databases. Virtual lot numbers delegate the identity information of the subcomponents to upper-level lot numbers without modifications to the databases. Our pruning method reduces the number of enumerated items and avoids redundant access to the databases. Experiments show that the algorithm works an order of magnitude faster than a naive approach.

Keywords: Distributed data mining, causal analysis, product recall.

1 Introduction

Distributed and parallel data mining has become important due to the increased amounts of data stored in various storage devices, as accessed by high-speed networking infrastructures [1]. Along this line of research, we focus on association mining from hierarchical distributed databases. In traditional data mining, a data record basically corresponds to an itemset, which is a set of items of a single conceptual level, such as milk and egg in a basket. The purpose is to find the associations between the items which are frequently included in the same itemsets. At the same time, we assume that the given top-level element can be separated into a set of lower level elements. For example, complex industrial products such as cars consist of thousands of parts which are hierarchically assembled. Association mining among hierarchically assembled products is important in practice, such as when a manufacturing company faces a safety recall of defective products. It is useful to find the set of suspect subitems (parts) that have high correlation rates with the list of failed products, since the company

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 731–740, 2009.
© Springer-Verlag Berlin Heidelberg 2009

wants to reduce the number of recalled products and minimize the costs. Each component is assigned its name (parts ID) and the product information is managed with identity labels such as *lot* numbers. This allows us to regard the causal analysis of recalls as the association rule mining between failed products and the parts of a specific lot number. Note that an association with failed product might not reflect the true cause of the failure. However, the causal items should show high associations and we should benefit from knowing the revealed associations.

Contemporary final products consist of tens of thousands of parts manufactured by other companies (called parts suppliers), which means that the assembly information is stored in physically distributed databases. Mining across such databases can be difficult and expensive since the queries must be transmitted over a network or through human processes. This forces us to address the real-world requirements of industrial information management where we cannot modify the transmission of assembly information between the suppliers. Though integrating the hierarchical databases into one data table allows using existing mining algorithms, it is generally unrealistic due to confidentiality issues and the massive size of the subassemblies of complex products. In most cases each company communicates only with one customer company (higher-level supplier) and with its own parts suppliers (lower-level suppliers). In addition, the anti-monotonicity of associations between items might be violated since an item can be included in multiple upper-level items. For example, two sets of electric circuits with different lot numbers may share capacitors with the same lot number. This makes it impossible to apply existing hierarchical mining algorithms that assume the anti-monotonic property.

The key of our approach is to introduce virtual lot numbers that connect current lot numbers with lower-level assembly information. The virtual lot numbers allow us to calculate the upper-bounds of the association rates with the descendant subitems and to prune the search space effectively in the tree structure of assembly. This approach also has a significant advantage in that we do not require any modifications to the design of the existing assembly databases, since the virtual lot number is still a kind of lot number (a natural number) so that it is stored and transmitted in the same way as a normal lot number within the distributed manufacturing system. We evaluate the efficiency of our algorithm using artificial datasets generated by a simulator of a manufacturing process.

2 Industrial Background

2.1 Product Recall

Product recalls are causing increasingly large problems for manufacturing industries. For example, major automotive companies have suffered from the safety recall problems of their cars. In the U.S. the number of recalled vehicles increased by 25% in 2007. Since the costs for recalls increase linearly with the number of recalled products, the company wants to recall the smallest number of suspicious products that might have the dangerous defect.

2.2 Hierarchical Assembly

In this paper we address association rule discovery in hierarchical assemblies of components. A top-level component (final product) contains a set of subcomponents. Since the subcomponents might also be separated into lower-level subcomponents, we regard the products as hierarchical-structured sets of subcomponents. In terms of the hierarchical production (assembly), the tree structure of the components also appears in the entire production system. We consider *suppliers* as manufacturers of components which form a hierarchical architecture like a supply chain of the subcomponents. For simplicity, we assume that each supplier produces only one component (part) without loss of generality. The assembly information transmitted and stored among the suppliers can also feed hierarchically connected databases. Note that these databases may be physically distributed and not connected via any network so that access to another database is exceedingly time-consuming. Therefore, we want to prune unnecessary database scans for efficient searches. These hierarchies of components, suppliers, and databases directly correspond to the structure of the parts assembly, supply chain and information used in the manufacture of an industrial product. Figure 1 shows the relationships between the composition of product A, its supply chain and the distributed assembly databases. The product consists of parts A1, A2, A3 according to its design as shown in Figure 1(a). Since each part can also has its own subcomponents such as A11 and A12, the structure is a tree. Figure 1(b) shows the hierarchical supply chain involving the manufacturer and the parts suppliers. The hierarchical structure also appears between the databases in Figure 1(c).

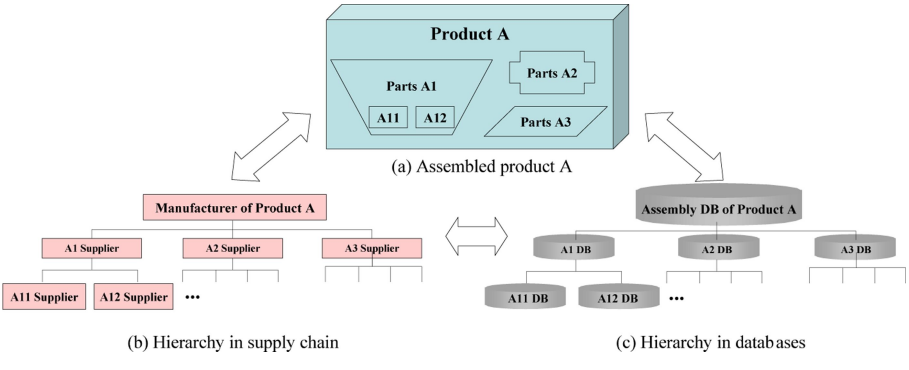


Fig. 1. Hierarchical assembly and database in manufacturing

2.3 Lot Representation

We define *item* (the minimum unit of a data instance) to be considered in our mining problem. Note that each component cannot be a data item by itself since the co-occurrence of different subcomponents is generally fixed by the design of

the final product. Instead, our interest is on a subset of objects of the parts, since the independent objects of the same parts might have different qualities and a fatal defect may be caused by a specific subset of the parts. Therefore we use a production label such as a lot number with its part ID as the data item. Lot-based management of assembly information is common in the manufacturing industry and there are some standards and infrastructure support such as the EPC (Electronic Product Code) [2] for information management across supply chains. The lot number tends to be updated when changes happen in the production environment such as an assembly line, workers, or production dates. Therefore we can assume that a set of components assigned to a lot number have almost the same quality due to their simultaneous production in the same environment.

Definition 1. *Each object in a hierarchically assembled product corresponds to an item* (id, ℓ) *where* id *indicates the part ID and* ℓ *represents the production label such as the lot number of the object.*

2.4 Assembly Database

We introduce a simplified design for hierarchically distributed databases of assembly information, relying on the structure of the supply chains. There are two types of database that each supplier maintains. The first is a product information database that stores production information of the assembly components. The product information database is typically a relational database in which each row represents a production object of the component identified by its lot number. The columns can include any information related to the products such as serial numbers or production dates. Let us note that the product databases are typically heterogeneous among suppliers and the column elements may differ. Table 1 represents an example of a production information database for product A. The column *Serial* denotes the serial number of each object. The columns *Lot* and *VirtualLot* show the original and virtual lot numbers that we define in Section 4.

The second type of supplier database is an assembly information database. The assembly information is basically the relationships between a supplier's own products at level i and the descendant lot numbers of their own subcomponents

Table 1. Product Information Database

Serial	Lot	VirtualLot
1	1	1
2	1	2
3	1	2
4	1	2
5	1	2
6	1	2

Table 2. Assembly Information Database

Serial	PartsID	SupplierID	VirtualLot
1	A1	A1-maker	103
1	A2	A2-maker	23
1	A3	A3-maker	299
2	A1	A1-maker	103
2	A2	A2-maker	23
2	A3	A3-maker	300

at level $i-1$. We can obtain the assembly information for all products easily as long as all of the sub-products are delivered and used in the order of their lot numbers. This connection across the assembly databases relates the properties of the high-level components to the quality of the descendant lower-level subcomponents. However, in general each supplier cannot directly access the databases of lower-level suppliers. Table 2 shows a corresponding assembly information database. Each row represents a subcomponent as shown in *PartsID* and *SupplierID* for an object specified with the serial number. Note that the assembly information databases of the deepest suppliers of the basic parts may be empty. The column *VirtualLot* denotes the virtual lot numbers of the subcomponents. We assume that every supplier has these two databases in our experiments.

Definition 2. *We have two types of manufacturing database for each supplier of a component. The product information database D^P stores the production information such as a lot number for each produced object. The assembly information database D^A represents the relationships between the product and its subcomponents represented as lot numbers.*

3 Mining Problem for Assembly Databases

Under the above setting, the only information we can use is the link of the lot numbers between a product and its sub-product parts. Therefore we calculate the association between two items based on the frequency of co-occurrences of their lot numbers. An entire set of items $\{(id_j, \ell_j)\}_{j=1}^N$ represents every combination of products and their lot numbers. $\sigma(I)$ is the number of the occurrences of an item $I \in \{(id_j, \ell_j)\}_{j=1}^N$. Given two items $A = (id_a, \ell_a)$ and $B = (id_b, \ell_b)$, $\sigma(A \wedge B)$ computes the number of co-occurrences of A and B. We can regard them as a pair with strong association if and only if both items appear in the same top-level products frequently more than threshold θ. We calculate the confidence of the association between A and B as $\sigma(A \wedge B)^2/\sigma(A)$, rather than the common definition $\sigma(A \wedge B)/\sigma(A)$. This weighted confidence allows us to emphasis the associations of smaller parts since single lot number is assigned to hundreds of objects for deep level parts. For example, 10% co-occurrence rate in $10,000$ objects with a lot number is meaningful than the same co-occurrence rate in only 100 objects that might be occurred by chance.

Definition 3. *For two items $A = (id_a, \ell_a)$ and $B = (id_b, \ell_b)$ in the itemset $\{(id_j, \ell_j)\}_{j=1}^N$, such that $\sigma(A \wedge B)^2/\sigma(A) \geq \theta$, we say that there is an association rule $A \rightarrow B$ in which B occurs frequently when A occurs.*

Finally we define the problem of the association rule mining from the hierarchically distributed databases.

Definition 4. *Given an entire itemset $\{(id_j, \ell_j)\}_{j=1}^N$, a set of focused top-level components R and threshold θ, enumerate all items which satisfy $R \rightarrow I$.*

We try to identify the suspect components and their lot numbers based on a set of failed top-level products $R = (id_{top}, \ell_{fail})$. Given a limited number of high-association items we can easily inspect each of them one-by-one to find the root cause of the failures. This allows the manufacturer to limit the number of recalled final products and minimize the cost of the safety recall.

4 Algorithm

4.1 Virtual Lot Number

We introduce the virtual lot numbers for efficient transmission of assembly information along the supply chain. Figure 2 shows an example of the virtual lot numbers of product A. The top row is the original lot numbers defined during the sequential production of A. The middle rows represent the lot splits of the sub-components of A. Double lines show the change points of their own lot numbers. The bottom row is the consequent virtual lot numbers. The value of the virtual lot number is updated when the original lot number changes or the lot split of a subcomponent occurs to combine the assembly information. The point is that virtual lot numbers can also be represented and stored as single numbers. Therefore we can replace the original lot numbers with the virtual numbers without modification to the existing system for transmitting lot numbers. This allows us to delegate the assembly information of lower-level subcomponents to the higher-level suppliers so that the aggregated assembly information of final products does not grow exponentially.

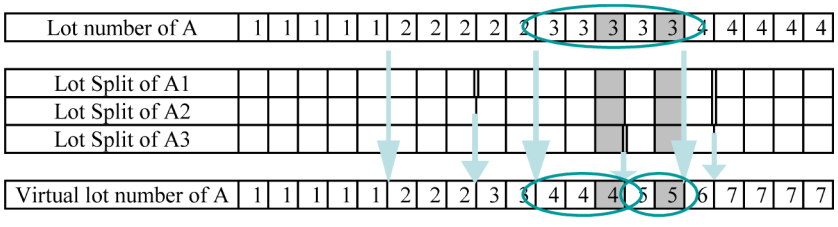


Fig. 2. Virtual lot number and upper bound of the confidence in assembly

4.2 Upper Bound of Association

We describe how to compute the association between failures and an item using virtual lot numbers efficiently. The point is that the objects of the parts with an original lot number are always assigned one or more virtual lot numbers. For example, in Figure 2, five objects of product A with the lot number 3 contain the virtual lot numbers 4 and 5. They include two failed objects so that the weighted confidence of the item is $(2*2)/5 = 0.8$. On the other hand, the weighted confidence with the virtual lot numbers 4 and 5 are $(1*1)/3 = 0.333\cdots$ and $(1*1)/2 = 0.5$. Then the sum of confidence $(0.833\cdots)$ is larger than 0.8. In fact, the virtual lot numbers provide the following theorem.

Theorem 1. *The sum of the weighted confidences with the virtual lot numbers is an upper bound of the weighted confidence of the corresponding original lot number.*

In the same way, we can compute the upper bound on a subcomponent since all lot numbers of a subcomponent also contain one or more virtual lot numbers. For example, the corresponding weighted confidence rates with items of subparts A1, A2 and A3 is not larger than $0.833\cdots$.

Theorem 2. *The sum of the weighted confidences with the virtual lot numbers is an upper bound of the weighted confidence of the corresponding items at lower-level.*

These theorems can be proved quite simply using the definition of the weighted confidence. Therefore, we use the associations with the virtual lot numbers as the upper bounds of weighted associations between the failures and the original lot number or the subcomponents. Based on these theorems we prune the redundant access to databases of descendant parts if the sum of the confidence rates of virtual lot numbers is smaller than the threshold. This is the key advantage of our trace mining algorithm compared to naive approach to compute all associations without pruning.

4.3 Trace Mining Algorithm

We present the pseudo code of the trace mining algorithm in Figure 3. First we choose the failed final product as the root on line 1. Next we calculate the upper bound of the confidence using virtual lot numbers on line 3. If the upper bound exceeds given threshold θ, we compute the actual confidence with original lot

Algorithm: Trace Mining
INPUT:
· Set of all databases D^P and D^A
· Set of failed final products R
· threshold $\theta > 0$
1. Let target item I_t be the final product R.
2. For each subcomponent I_{sub} of I_t.
3. $\hat{C}$ = the sum of confidence of with virtual lot numbers of I_{sub}.
4. If $\hat{C} > \theta$ then
5. C = the confidence of with original lot number of I_{sub}.
6. If $C > \theta$ then
7. Add I_sub to the set of defective items.
8. End If
9. Let I_{sub} be I_t and iterate the loop.
10. End If
11. End For
12. Output the set of detected items.

Fig. 3. Trace mining algorithm for assembly databases

number on line 5. If the actual confidence rate is larger than θ, we append the item to the set of output on line 7. Next we repeat the same procedure at deeper-level starting from line 2 for the subcomponents. Thus the search continues in a depth-first manner along the hierarchy of the supply chain with pruning on line 4. The calculation of the confidence is based on the databases D^P and D^A. Note that the databases may not be integrated in practice so that the access is exceedingly time-consuming. Based on the algorithm, we can obtain all defective items of which confidence of the association is higher than θ without loss.

5 Related Work

Cheung et al. [3] first addressed the problem and proposed Fast Distributed Mining of association rules (FDM). Zaki et al. utilized itemset clustering for efficient partitioning of the database into the subsets [4]. Schuster et al. proposed communication-efficient algorithm for distributed association mining [5]. Since these algorithms only handle traditional data items of single-level so that they are not directly applicable to the mining from hierarchically distributed databases.

Li et al. introduced an approach for mining higher-order association rule from distributed text databases of which schema is different each other [6]. First the method find out linkable records of linguistic features between local databases by Apriori using global ID assigned to the linkage items such as social security number. Next it exchanges data entities about linkable records between all databases and applying Apriori again to the merged local database. In addition, multi-level association rule mining was addressed by Han et al. [7]. The method considers taxonomic level of items and apply existing mining approach in a top-down manner. For example, association between wheat bread and milk is more informative than that of bread and milk. The point is to represent each item as digits of fixed length where the value of each digit corresponds to a category at a conceptual level. This is equivalent to integrate all hierarchically distributed databases into single data table, which is not sufficient in practice for our problem.

Traceability has become a key technology in manufacturing especially for investigating the relationship between defective products and parts in recall. Robson et al. addressed a problem for the management of suppliers' databases to support reverse lookups of parts [8]. The method is based on a table of Component InDex (CID) which maintains the relationship between each final product and its parts in a similar way to ours. However, the construction of CID table requires complete survey of all databases at least once. This may be inefficient for complex products of which assembly information is stored in hundreds of databases. In addition, their main concern is the volatile network of supply chain where the supplier of a parts is often replaced, which is different from ours.

6 Experiment

We evaluate the proposed algorithm and compare its performance with naive approach.

Table 3. Result of trace mining algorithm

Size	θ	# item	# DB	# access	sec
500	0.2	81	158	174	9.81
	0.4	8	99	99	7.20
	0.6	0	2	2	0.42
1000	0.2	108	163	184	25.61
	0.4	12	140	140	19.88
	0.6	2	103	103	16.91
1500	0.2	117	193	266	62.50
	0.4	14	146	149	34.34
	0.6	2	136	136	31.06
2000	0.2	122	207	290	195.42
	0.4	12	155	165	107.11
	0.6	1	141	143	68.39

Table 4. Comparison between brute force approach and trace mining algorithm

Size	Algorithm	# DB	# access	sec
500	Naive	212	759	23.23
	Proposed	99	99	7.20
1000	Naive	212	775	81.05
	Proposed	140	140	19.88
1500	Naive	212	743	144.03
	Proposed	146	149	34.34
2000	Naive	212	791	778.25
	Proposed	155	165	107.11

First we implement a simulation of producing an artificial product of which structure is defined in a XML-formatted BOM. The BOM file specifies the hierarchical structure of size, production size, supplier of the product and the defect rate. During the simulation, we compute the assembly information of objects of every component and obtain the production information database and assembly information database for each supplier. In the BOM we embed a defective parts of which defect rate is much higher than other parts. Every object of parts is defective with the probability of its defect rate and assigned failed if it or its parts is defective. Thus the failures are propagated from deep-level components to final product along the tree-structured hierarchy. Therefore, some of final products are determined as failed objects mainly caused by the defective parts.

Next we perform mining algorithms to identify the defective parts given the suppliers' databases, the list of the failed final products with changing the production size and the threshold. We compare the trace mining algorithm with a naive approach to compute all associations without pruning.

Table 3 shows the result over 10 trials. Column *#item* represents the number of detected items of which association with failed products exceeds the threshold. In most cases, the embedded defective parts is successfully detected as the most suspicious item. Column *#DB* and *#item* represents the number of reached DB and the access to the DB respectively during the calculation. Thus these numbers correspond to the efficiency of the algorithm. As shown in Column *sec*, the execution time fluctuates depending on the size of production and the threshold. Table 4 compares our algorithm with the naive approach with threshold $\theta = 0.4$. Note that the outputs of two algorithms are exactly the same since our algorithm never miss the defective parts. The table indicates that the number of access on the proposed algorithm is much smaller than that of the naive approach. Thus the trace mining algorithm one order of magnitude faster than the naive approach. Note that the difference between the execution times should be more extreme in practice due to the physically distributed databases though the databases are all in main memory in this experiment.

7 Conclusion

We focused on the association mining for hierarchically assembled items. We proposed the trace mining algorithm for identifying the subitems which have high associations with a set of top-level items. The point is to implement the virtual lot numbers that represent level i lot number and the split of level $(i-1)$ lot numbers at the same time. This allows us to compute the upper-bound of association rate for possible subitems and to avoid unnecessary access to distributed databases. The algorithm can be applied to the manufacturing databases of complex products such as cars without modification to the databases. Experimental evaluation using simulated assembly databases shows that our algorithm can identify embedded failure items correctly and perform with one order of magnitude fewer access to the databases.

References

1. Zaki, M.J.: Parallel and distributed association mining: A survey. IEEE Concurrency 7(4), 14–25 (1999)
2. Brock, D.: The Electronic Product Code (EPC), a naming scheme for physical objects. White Paper, MIT Auto-ID Center (2001)
3. Cheung, D., Han, J., Ng, V.T., Fu, A.W., Fu, Y.: A fast distributed algorithm for mining association rules. In: Proceedings of the PDIS, p. 0031 (1996)
4. Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: Parallel algorithms for discovery of association rules. Data Mining and Knowledge Discovery 1(4), 343–373 (1997)
5. Schuster, A., Wolff, R.: Communication-efficient distributed mining of association rules. In: Proceedings of the 2001 ACM SIGMOD international conference on Management of data, vol. 8(2), pp. 171–196 (2004)
6. Li, S., Wu, T., Pottenger, W.M.: Distributed higher order association rule mining using information extracted from textual data. SIGKDD Explorations Newsletter 7(1), 26–35 (2005)
7. Han, J., Fu, Y.: Discovery of multiple-level association rules from large databases. In: Proceedings of the 21nd VLDB, pp. 420–431 (1995)
8. Robson, C., Watanabe, Y., Numao, M.: Parts traceability for manufacturers. In: Proceedings of the 23rd IEEE ICDE, pp. 1212–1221 (2007)

Let's Tango – Finding the Right Couple for Feature-Opinion Association in Sentiment Analysis

Kam Tong Chan and Irwin King

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
{ktchan,king}@cse.cuhk.edu.hk

Abstract. One approach in opinion mining is to perform sentiment classification at the sentence level. User's view on a discovered product feature is predicted by the opinion words, e.g. adjectives, appeared in the same sentence. A number of previous works has been proposed and these approaches typically treat the feature and word relations identically. Blindly using sentiments of all opinion words to perform classification would lead to false results. In this paper, we investigate the relationship between features and opinion words using the corpus-based approach. We proposed a Feature-Opinion Association (FOA) algorithm to match these two in sentences to improve sentiment analysis results. We construct a feature-based sentiment lexicon using the proposed algorithm in the sentiment identification process. Extensive experiments based on a commercial product review site show that our method is quite effective in obtaining a more accurate result.

Keywords: Opinion mining, sentiment analysis.

1 Introduction

One task in opinion mining is to perform feature-based summarization (FBS), which identifies target features mentioned in the reviews and determine their sentiment. Early approaches in FBS use a predefined sentiment lexicon to check against the extracted terms in reviews [2]. The problem of this approach is that sentiments of opinion words are context sensitive. Improvements have been made by generating lexicons with are domain-specific [3] or feature-specific [1].

An accurate and robust Feature-Opinion Association (FOA) method is crucial for both lexicon generation and sentiment classification. That is because more than one feature and opinion word may be mentioned in a sentence. It is not necessary that all opinion words appeared in the sentence are used to describe every feature. An accurate FOA algorithm tells us which feature an opinion word is describing and thus the correct *sense* of that word can be used.

In this paper, we propose to use a function to compute the relevance score between features and opinion words. We then propose an FOA algorithm to

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 741–748, 2009.
© Springer-Verlag Berlin Heidelberg 2009

match features and opinion words by maximizing the sum of the relevance scores of sentences. The algorithm can be used to generate a sentiment lexicon and be used in sentiment analysis. Experiment results show that our method is useful in improving the sentiment classification accuracy.

2 Proposed Approach

In this section, we give definitions to the Feature-Opinion Association problem and present our proposed solution to the problem.

2.1 Feature-Opinion Association

We define *Feature* and *Opinion words* as follows:

Feature can be a component of the product, e.g. Flash, Lenses, or it can be an attribute of the product, e.g. Weight, Size. Products in the same category share a similar set of common features.

Opinion words in general can be anything that is used to describe a feature. However, due to the difficulties in natural language understanding, it is not easy to perform sentiment analysis on all types of opinion. As with most of the existing works, we limit our scope to handle opinion words that are in the from of adjectives and adverbs.

Here, we define the Feature-Opinion Association Problem (FOA) as follows:

Definition 1. *Given a sentence that contains a non-empty set of features* $F = \{f_0, f_1, ..., f_n\}$ *and opinion words* $W = \{w_0, w_1, ..., w_m\}$*, find out a matching* M *that matches product features with their related opinion words so that the following function is maximized:*

$$foa(F, M) = (\sum_{i=0}^{n} \sum_{j=0}^{|M(fi)|} rel(f_i, M(f_i)(j)). \quad (1)$$

In Eq. (1), $M(f)$ is the set of opinion words matched to feature f and $M(f)(i)$ is the i-th opinion word in the set. The function $rel(f, w)$ returns the relevance score of opinion word w to feature f. The key to this association problem is to define a good $rel(f, w)$ function.

2.2 *Rel* Function

Here is a list of possible *rel* functions for the FOA problem:

- **Nearest Opinion Word (DIST)**
  The simplest solution to the FOA problem is to associate opinion words that are nearest to features. In this case, $rel(f, w)$ is defined as the inverse of the distance between the opinion word w and feature f.

– **Co-Occurrence Frequency (COF)**
Another approach to the FOA problem is to define $rel(f, w)$ as the Co-Occurrence Frequency (COF) between feature f and opinion word w. The intuition is that the higher the COF score, the more likely that the opinion word is related to the feature.
– **Co-Occurrence Ratio (COR)**
COF has the problem that common terms (i.e., "good", "great", "excellent") dominate the top of the rank. To get a more feature-specific ranking, we should also take into accounts the corpus frequency (CF) (i.e. the number of times a term appears in the corpus) of opinion words. In this case, the *rel* function becomes:

$$ref(f,w) = \begin{cases} \frac{COF(f,w)}{CF(w)} & \text{if COF(f,w)} > \text{threshold} \\ 0 & \text{otherwise} \end{cases}$$

– **Likelihood-Ratio Test (LHR)**
Using the method proposed by Yi [4], assuming the association between feature and opinion word is a Bernoulli event and it follows a binomial distribution, we can compute the likelihood ratio as follows:

$$-2\log\lambda = \begin{cases} -2 * lr & \text{if } r_2 < r_1 \\ 0 & \text{if } r_1 < r_2 \end{cases}$$

where

$$\begin{aligned} lr =& (C_{11} + C_{21})\log(r) + (C_{12} + C_{22})\log(1 - lr) \\ & - C_{11}\log(r_1) - C_{12}\log(1 - r_1) - C_{21}\log(r_2) \\ & - C_{22}\log(1 - r_2) \end{aligned}$$

$$\begin{aligned} r_1 =& \frac{C_{11}}{C_{11} + C_{12}} \\ r_2 =& \frac{C_{21}}{C_{21} + C_{22}} \\ r =& \frac{C_{11} + C_{21}}{C_{11} + C_{12} + C_{21} + C_{22}} \end{aligned}$$

$$\begin{aligned} C_{11} =& COF(f,w) \\ C_{12} =& CF(w) - COF(f,w) \\ C_{21} =& CF(f) - COF(f,w) \\ C_{22} =& SentenceCount - C_{11} - C_{12} - C_{21} \end{aligned}$$

The computed log likelihood ratio $-2\log\lambda$ is used as *rel* score in this case. A larger value represents a higher chance that opinion word w should be associated with feature f.

– **Combined Method**
In here, we combine DIST, a distance based measurement, with statistical measurements COR and LHR. This is to combine the best of both types of measurements. They are defined as follows:
COR+DIST (C+D):

$$rel(f,w) = \frac{COF(f,w)}{CF(w) * dist(f,w)}. \tag{2}$$

LHR+DIST (L+D):

$$rel(f,w) = \frac{-2\log\lambda}{dist(f,w)}. \tag{3}$$

2.3 Feature-Opinion Association Algorithm

We define the Feature-Opinion Association Algorithm (FOAA) to associate opinion words to features subject to the *rel* function in Algorithm 1:

Algorithm 1. Feature-Opinion Association Algorithm

```
F ← features in S
W ← opinion words in S
for each w in opinion word list W do
   score ← highest rel(f, w) for all f ∈ F
   if score ≥ threshold then
      if the same word is already assign to f then
         Try another f with the next highest foa score
      else
         associate w to f
      end if
   end if
end for
```

In general, each opinion word will be associated to the feature with the highest *rel* score except in sentences like: *"Good lenses, good pictures!"*. Depending on the *rel* function used, it is possible that the scores for the same pair of feature and opinion word are the same for different appearances of the same opinion word. The algorithm will check if the same opinion word is already associated to a feature, and if so, it tries the next feature until no more features exist. A threshold value is used to prune opinion words that have low *rel* scores to all features appeared in the sentence.

2.4 Sentiment Lexicon Expansion

With the Feature-Opinion Association algorithm, steps for sentiment lexicon expansion become straightforward. Two sets of opinion words (positive and negative) are defined initially as seeds. FOA is applied on each sentence appeared

in the training corpus. Same opinion words associated to different feature are treated as different words. Using the linguistic rules proposed by Ding [1], we can compute the orientation score for each word as follows:

$$orientation(w) = \frac{c_{+ve} - c_{-ve}}{CF(w)}, \tag{4}$$

where c_{+ve} and c_{-ve} is the number of times word w is in conjunction with the known positive and negative set respectively. The higher the score, the more likely that the word should be in the positive set and vice vera. Words with absolute score smaller than a threshold T should not be treated as either polarity. In each iteration, Words with the highest and lowest score are added to the two sets respectively.

3 Evaluation

In order to verify our ideas, we collected two data sets from the internet. These two data sets are used to conduct our experiments. NLTK[1] is used to perform natural language processing tasks such as sentence splitting and part-of-speech (POS) tagging. Feature list is constructed based on the glossary page of the Digital Photography Review[2]. Synonyms are grouped together and treated as the same feature.

There are two data sets in our experiment:

Corpus Data Set: User reviews of all cameras of popular brands are crawled from Digital Photography Review. This data set contains 400+ different camera models, 17,000+ user reviews and 250,000+ sentences. This data set is used as a statistical database for computing the relevance scores discussed in Section 2.

Test Data Set: We used another data set [2,1] for testing. The reviews of 4 different camera models are used. The data set contains over 100 reviews and 1500 sentences. The reviews are re-tagged so that each sentence is attached with the mentioned features and their associated opinion words.

3.1 Feature-Opinion Association Accuracy

For each tagged sentence in the test data set, we use the FOA algorithm to match appeared features and opinion words. Therefore, for each feature, there will be a list of opinion words tagged by human and the FOA algorithm respectively. We have defined the following values to count the number of opinion-feature pair that are tagged by either Human or the FOA algorithm:

- C_{AA}: Both Human and FOA tagged
- C_{AN}: Only Human tagged

[1] http://nltk.org/
[2] http://www.dpreview.com/

Table 1. FOA Accuracies

	DIST	**COF**	**COR**	**LHR**	**C+D**	**L+D**
th	0.20	1.00	0.01	0.00	0.02	9.00
p	0.58	0.45	0.46	0.49	0.59	0.66
r	0.64	0.75	0.71	**0.79**	0.65	0.64
f	0.61	0.57	0.56	0.60	**0.62**	**0.65**

- C_{NA}: Only FOA tagged
- C_{NN}: Neither Human nor FOA tagged

Then, the association accuracy can be computed by the traditional precision (p), recall (r) and F-score (f):

$$p = \frac{C_{AA}}{C_{AA} + C_{NA}}, r = \frac{C_{AA}}{C_{AA} + C_{AN}}, f = \frac{2 * p}{p + r}.$$

We evaluate the accuracy of FOA using each *rel* function with a range of possible threshold *th*. For COR, DIST and COR+DIST, the range is from 0 to 1 with step size 0.01. Range for other functions is from 0 - 100 with step size 1. The results of each data set with the best F-Score is presented in Table 1. We only report the average results of the 4 data set due to space limitations.

From the results we can see that non-combined *rel* functions are capable of achieving good recalls (around $65-79\%$). However, their precisions are generally quite low. Among all the non-combined *rel* functions, LHR and DIST performed the best, reaching an average F-Score of 60%. DIST achieved the highest average precision, but it is mainly due to the outliner camera 1 (not shown here). LHR has the highest average recall (nearly 80%).

The combined *rel* functions, in general, perform better in terms of the F-Score measurements with LHR+DIST slightly outperforming COR+DIST. The results here indicate the importance of using both types of measurement in the feature-opinion association process. Missing either information will lead to worse association results.

3.2 Sentiment Classification Accuracy

We used LHR and LHR+DIST to generate two sentiment lexicons by the algorithm discussed in section 2.4. The reason for choosing them is that they performed the best in terms of average Precision, Recall and F-Score in the FOA process. The FOA thresholds (*th*) 0, and 9 are selected for LHR and LHR+DIST respectively. They are thresholds that each methods achieve the best average F-Score in the FOA process. Opinion words {"excellent", "good"} and {"poor", "bad"} are used as the initial seed words for the positive and negative sentiment respectively. In our experiment, the orientation threshold T for lexicon generation is set to 0.2.

We conducted two set of experiments with identical settings except that one includes FOA while the other does not. The sentiment classification process is

Table 2. Sentiment Classification Accuracy

		LHR	**L+D**		**LHR**	**L+D**		**LHR**	**L+D**		**LHR**	**L+D**		**LHR**	**L+D**
All	**Camera 1**	0.745	0.668	**Camera 2**	0.703	0.695	**Camera 3**	0.799	0.672	**Camera 4**	0.663	0.626	**Average**	0.727	0.665
Human		0.761	0.719		0.740	0.740		0.797	0.709		0.656	0.660		0.739	0.707
Human*		0.772	0.710		0.749	0.748		0.845	0.721		0.653	0.667		0.755	0.711
FOA		0.692	0.662		0.685	0.647		0.785	0.656		0.629	0.585		0.698	0.638
FOA*		**0.755**	0.695		0.729	**0.736**		**0.828**	0.716		**0.694**	0.636		**0.752**	0.696

as follows: For each tagged sentences, all the appeared features and opinion words are extracted. The algorithm computes the sentiment score for each feature mentioned in the sentence solely based on the associated opinion words. Features that are not associated to any opinion words will have their sentiments inferred using two different methods. The first method (*Human* and *FOA*) uses the majority sentiment of other features that appeared at the same sentence. The second method (*Human** and *FOA**) falls back to use all opinion words of the same sentence to infer sentiments of these features. This is the same as the case where FOA is not used. Under both cases, we use the opinion aggregation function [1] for sentiment scoring:

$$score(f) = \sum_{w_i : w_i \in M(f)} \frac{w_i.SO}{dist(w_i, f)}. \tag{5}$$

$M(f)$ is the set of opinion words associated to feature f. For the case where FOA is not used (*All*), $M(f)$ will be replaced by the set of all opinion words within the same sentence as feature f. (*All*) represents the opinion aggregation method proposed in [1]. This give us a direct comparison of FOA versus the state of the art method in sentiment analysis. $dist(w_i, f)$ is the distance between opinion word w_i and feature f. $w_i.SO$ is the sentiment score of opinion word w_i. Positive words have a score of +1 and negative words have a score of -1. Sentiments of opinion words are retrieved from sentiment lexicons generated in the above steps. The predicted sentiment orientations are compared against human tags to calculate accuracies. Table 2 summarizes the results of our sentiment classification experiments.

Classification results using human tags. We first compare the sentiment classification accuracies of all opinion words (*All*) and human association (*Human* and *Human**). We can see that the overall accuracies increase if we limit the sentiment classifier to use only opinion words that are tagged by human. These agree with our intuitions that blindly using all opinion words actually produces false results.

An interesting observation is that *Human** performs better than *Human*. The reason is that a sentence usually mentions only 1 or 2 features. When there are no associated opinion words, we either cannot find another feature, or the remaining features are not enough to help inferring its sentiment correctly. Falling back to use all opinion words actually helps in this case.

Classification results using FOA algorithm. Solely using opinion words that are associated by the FOA algorithm for sentiment analysis (*FOA* in Table 6) actually produces poorer results. This is reasonable, given that the accuracies of using human FOA alone (*Human*) are just slightly better than the case where all opinion words are used (*All*). However, if we use the second method to deal with the case where no opinion words are associated to a feature (*FOA**), the overall accuracy improves and it consistently outperform *All* in our experimental data sets. In fact, we can interpret FOA* as a pruning heuristic that tries to tighten the set of opinion words used for sentiment classification whenever possible. The results show that this heuristic is effective in improving the sentiment classification accuracies.

Finally, we observe that using LHR to perform FOA actually performed better. A possible reason is that LHR is good at achieving high recalls. It extracts most of the feature-related opinion words in the lexicon building process. Although its precision is not high, the incorrectly associated opinion words are unlikely to be inferred to carry a sentiment because of the threshold limitations.

4 Conclusion

We have presented a Feature-Opinion Association (FOA) algorithm to improve the sentiment analysis results. The algorithm maximizes the sum of the relevance scores between features and opinion words. We proposed 6 relevance measures that make use of the structural information of sentences as well as the statistical information collected from a commercial review web site. The proposed FOA algorithm can be used in both the lexicon generation and sentiment classification process. The evaluation results show that it is effective in improving the sentiment analysis accuracy over the traditional methods where classification is done using all opinion words.

Acknowledgements

This work is fully supported by two grants (Project No. CUHK 2050379 and Project No. CUHK 6902498) and is affiliated with the Microsoft-CUHK Joint Laboratory for Human-centric Computing and Interface Technologies.

References

1. Ding, X., Liu, B.: A holistic lexicon-based approach to opinion mining. In: WSDM 2008, pp. 231–240 (2008)
2. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: KDD 2004, pp. 168–177 (2004)
3. Kanayama, H., Nasukawa, T.: Fully Automatic Lexicon Expansion for Domain-oriented Sentiment Analysis. In: EMNLP 2006, pp. 355–363 (2006)
4. Yi, J., Nasukawa, T., Bunescu, R., Niblack, W.: Sentiment Analyzer: Extracting Sentiments about a Given Topic using Natural Language Processing Techniques. In: ICDM 2003, p. 427 (2003)

An Efficient Candidate Pruning Technique for High Utility Pattern Mining

Chowdhury Farhan Ahmed, Syed Khairuzzaman Tanbeer,
Byeong-Soo Jeong, and Young-Koo Lee

Department of Computer Engineering, Kyung Hee University
1 Seochun-dong, Kihung-gu, Youngin-si, Kyunggi-do, 446-701, Republic of Korea
{farhan,tanbeer,jeong,yklee}@khu.ac.kr

Abstract. High utility pattern mining extracts more useful and realistic knowledge from transaction databases compared to the traditional frequent pattern mining by considering the non-binary frequency values of items in transactions and different profit values for every item. However, the existing high utility pattern mining algorithms suffer from the level-wise candidate generation-and-test problem and need several database scans to mine the actual high utility patterns. In this paper, we propose a novel tree-based candidate pruning technique HUC-Prune (high utility candidates prune) to efficiently mine high utility patterns without level-wise candidate generation-and-test. It exploits a pattern growth mining approach and needs maximum three database scans in contrast to several database scans of the existing algorithms. Extensive experimental results show that our technique is very efficient for high utility pattern mining and it outperforms the existing algorithms.

Keywords: Data mining, knowledge discovery, frequent pattern mining, high utility pattern mining, association rules.

1 Introduction

Frequent pattern mining [1], [5], [8] plays an important role in data mining and knowledge discovery techniques such as association rule mining, classification, clustering, time-series mining, graph mining, web mining etc. The initial solution of frequent pattern mining, level-wise candidate set generation-and-test paradigm of *Apriori* [1] has revealed many drawbacks that it requires multiple database scans and generates lots of candidate patterns. FP-growth [5] solved this problem by introducing a prefix-tree (FP-tree) based algorithm without candidate set generation-and-test. However, frequent pattern mining has two principal limitations. First, it treats all the items have the same importance/weight/price and second, in one transaction each item appears in a binary (0/1) form, i.e. either present or absent. But in our real world scenarios, each item in the supermarket has different importance/price and one customer may buy multiple copies of an item. Therefore, frequent patterns only reflect the correlation between items, and it does not reflect the semantic significance of the items.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 749–756, 2009.
© Springer-Verlag Berlin Heidelberg 2009

A high utility mining [2], [3], [4], [6], [7] model was defined to solve the above limitations of frequent pattern mining. It allows the user to conveniently measure the importance of an itemset by the utility value. By utility mining, several important decisions in business area like maximizing revenue, minimizing marketing or inventory costs can be taken and more important knowledge about itemsets/customers contributing to the majority of the profit can be discovered. Other application areas, such as biological gene database, web click streams, stock tickers, network traffic measurements, web-server logs, data feeds from sensor networks and telecom call records can have similar solutions.

The existing high utility pattern mining algorithms [2], [3], [4], [7] suffer from the level-wise candidate generation-and-test problem. Therefore, they generate a huge number of candidates and need several database scans to mine the actual high utility patterns. Moreover, their number of database scans is directly dependent on the maximum length of the candidate patterns. In this paper, we propose a novel tree-based candidate pruning technique HUC-Prune (high utility candidates prune) to efficiently mine high utility patterns without level-wise candidate set generation-and-test. It prunes a large number of unnecessary candidates during the mining process. It exploits a pattern growth mining approach and needs a maximum of three database scans in contrast to several database scans of the existing algorithms. Extensive experimental results show that our technique is very efficient for high utility pattern mining and it outperforms the existing algorithms.

The remainder of this paper is organized as follows. In Section 2, we describe the related work. In Section 3, we describe the high utility pattern mining problem. In Section 4, we describe our proposed pruning technique HUC-Prune to efficiently mine high utility patterns. In Section 5, our experimental results are presented and analyzed. Finally, in Section 6, conclusions are drawn.

2 Related Work

The theoretical model and definitions of high utility pattern mining were given in [2], named MEU (mining with expected utility). They cannot maintain the *downward closure* property of *Apriori*. They used a heuristic to determine whether an itemset should be considered as a candidate itemset. It usually overestimates, especially at the beginning stages, where the number of candidates approaches the number of all the combinations of items. This is impractical whenever the number of distinct items is large and utility threshold is low. Later, the same authors proposed two new algorithms UMining and UMining_H [3] to calculate the high utility patterns. In UMining they have used a pruning strategy based on utility upper bound property. UMining_H has been designed with another pruning strategy based on a heuristic method. But, some high utility itemsets may be erroneously pruned by their heuristic method. Moreover, these methods do not satisfy the *downward closure* property of *Apriori* and also suffer from the level-wise candidate generation-and-test methodology.

The Two-Phase [4] algorithm was developed based on the definitions of [2] to find high utility itemsets using the *downward closure* property. They have defined "transaction weighted utilization", and proved it can maintain the *downward closure* property. For the first database scan, their algorithm finds all the 1-element transaction weighted utilization itemset and based on that generates the candidates for 2-element transaction weighted utilization itemsets. In the second database scan, it finds all the 2-element transaction weighted utilization itemset and based on that generates the candidates for 3-element transaction weighted utilization itemsets and so on. At the last scan, it finds out the actual high utility itemsets from high transaction weighted utilization itemsets. This algorithm suffers from the same problem of the level-wise candidate generation-and-test methodology. CTU-Mine [6] proposed an algorithm that is efficient over Two-Phase algorithm only in dense database when the minimum utility threshold is very low.

The isolated items discarding strategy (IIDS) [7] for discovering high utility itemsets was proposed to reduce some candidates in every pass of databases. They developed efficient high utility itemset mining algorithm FUM and DCG+ and showed that their work is better than all previous high utility pattern mining works. But still their algorithms suffers from the level-wise candidate generation-and-test problem and needs multiple database scans depending on the maximum length of the candidate patterns. Therefore, we propose a novel tree-based pruning technique to remove these problems of the existing works.

3 Problem Definition

We have adopted similar definitions presented in the previous works [2], [3], [4]. Let $I = \{i_1, i_2, ...i_m\}$ be a set of items and D be a transaction database $\{T_1, T_2, ...T_n\}$ where each transaction $T_i \in D$ is a subset of I. The local transaction utility value $l(i_p, T_q)$, represents the quantity of item i_p in transaction T_q. For example, in Fig. 1(a), $l(b, T_2) = 6$. The external utility $p(i_p)$ is the unit profit value of item i_p. For example, in Fig. 1(b), $p(c) = 3$. Utility $u(i_p, T_q)$, is the quantitative measure of utility for item i_p in transaction T_q, defined by

$$u(i_p, T_q) = l(i_p, T_q) \times p(i_p) \tag{1}$$

For example, $u(b, T_1) = 2 \times 6 = 12$ in Fig. 1. The utility of an itemset X in transaction T_q, $u(X, T_q)$ is defined by,

$$u(X, T_q) = \sum_{i_p \in X} u(i_p, T_q) \tag{2}$$

where $X = \{i_1, i_2, ...i_k\}$ is a k-itemset, $X \subseteq T_q$ and $1 \leq k \leq m$. For example, $u(bc, T_1) = 2 \times 6 + 8 \times 3 = 36$ in Fig. 1. The utility of an itemset X is defined by,

$$u(X) = \sum_{T_q \in D} \sum_{i_p \in X} u(i_p, T_q) \tag{3}$$

TID \ ITEM	a	b	c	d	e	Trans. Utility($)
T_1	0	2	8	2	0	52
T_2	4	6	0	0	0	44
T_3	0	3	0	0	2	38
T_4	2	2	7	0	0	37
T_5	6	5	0	4	0	74
T_6	4	4	0	5	3	102
T_7	0	0	0	3	4	64
T_8	0	0	0	2	0	16

(a) Transaction database

ITEM	PROFIT($) (per unit)
a	2
b	6
c	3
d	8
e	10

(b) Utility table

Fig. 1. Example of a transaction database and utility table

For example, $u(ab) = u(ab, T_2) + u(ab, T_4) + u(ab, T_5) + u(ab, T_6) = 44 + 16 + 42 + 32 = 134$ in Fig. 1. The transaction utility of transaction T_q denoted as $tu(T_q)$ describes the total profit of that transaction and it is defined by,

$$tu(T_q) = \sum_{i_p \in T_q} u(i_p, T_q) \tag{4}$$

For example, $tu(T_1) = u(b, T_1) + u(c, T_1) + u(d, T_1) = 12 + 24 + 16 = 52$ in Fig. 1. The minimum utility threshold δ, is given by the percentage of total transaction utility values of the database. In Fig. 1, the summation of all the transaction utility values is 427. If δ is 25% or we can also express it as 0.25, then the minimum utility value can be defined as

$$minutil = \delta \times \sum_{T_q \in D} tu(T_q) \tag{5}$$

Therefore, in this example $minutil = 0.25 \times 427 = 106.75$ in Fig. 1. An itemset X is a high utility itemset, if $u(X) \geq minutil$. Finding high utility itemsets means determining all itemsets X having criteria $u(X) \geq minutil$. The main challenge of facing high utility pattern mining areas is the itemset utility does not have the *downward closure* property. For example, if $minutil = 106.75$ in Fig. 1, then "e" is a low utility item as $u(e) = 90$. However, "de" is a high utility itemset as $u(de) = 134$. So, the *downward closure* property is not satisfied. We can maintain the *downward closure property* by transaction weighted utilization. Transaction weighted utilization of an itemset X, denoted by $twu(X)$, is the sum of the transaction utilities of all transactions containing X.

$$twu(X) = \sum_{X \subseteq T_q \in D} tu(T_q) \tag{6}$$

For example, $twu(bc) = tu(T_1) + tu(T_4) = 52 + 37 = 89$ in Fig. 1. Here, for $minutil = 106.75$ in Fig. 1 as $twu(bc) < minutil$, any super pattern of "bc" cannot be a high *twu* itemset (candidate itemset) and obviously cannot be a high utility itemset. X is a high transaction weighted utilization itemset (i.e. a candidate itemset) if $twu(X) \geq minutil$.

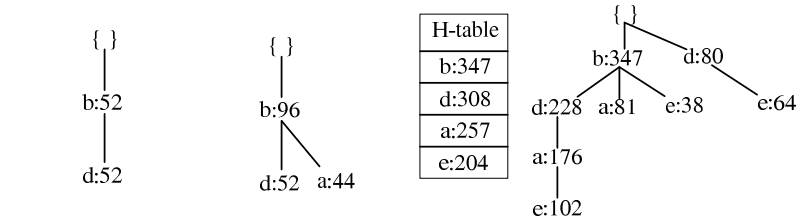


Fig. 2. Tree construction process

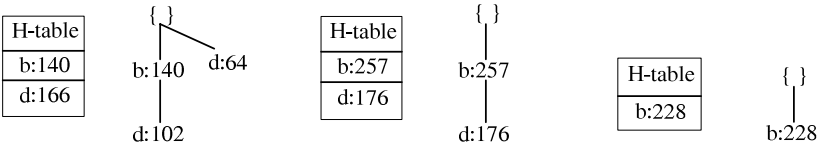
4 HUC-Prune: Our Proposed Technique

At first, we describe the construction process of our tree structure. Header table is maintained in our tree structure. Each entry in a header table and node of the tree explicitly maintain item-id and *twu* (transaction weighted utilization) value for each item. To facilitate the tree traversals adjacent links are also maintained (not shown in the figures for simplicity) in our tree structure. In the first database scan HUC-Prune captures the *twu* value of all the items in order to prune unnecessary candidates. We have explained in Section 3 that the *downward closure* property can be maintained by using the *twu* value. Therefore, by pruning the single-element items having *low-twu* value with respect to the given threshold, HUC-Prune achieves huge gain in tree-based candidate generation process.

Consider the database shown in Fig. 1 and $minutil = 106.75$. After the first database scan, the twu values are <a:257, b:347, c:89, d:308, e:204>. As a result, item "c" is a low *twu* item and according to the *downward closure* property its any superset cannot be high utility itemset. Therefore, we prune this item. Next, we sort the header table in descending order according to their *twu* values and the new order is <b:347, d:308, a:257, e:204>. In the second database scan, we take only high *twu* items from the transactions, sort them according to the header table order, and insert them into the tree. For the first transaction T_1, which contains item "b", "c" and "d", we discard the low *twu* item "c" and arrange it according to the header table order. Both "b" and "d" get the *tu* value of T_1 (the *tu* value of T_1 is 52, shown in Fig. 1). Fig. 2(a) shows the tree after inserting T_1. After that T_2 is inserted in the tree (shown in Fig. 2(b)). Before insertion the items of T_2 are arranged (at first "b" then "a") in the header table order. Item "b" gets the prefix sharing with the existing node containing item "b". Its *twu* value becomes 52+44=96, and item "a" becomes its child with *twu* value of 44. Fig. 2(c) shows the final tree with the header table for the full database presented at Fig. 1. The following property is true for our tree structure.

Property 1. The total count of *twu* value of any node in the tree is greater than or equal to the sum of total counts of *twu* values of its children.

Now, we describe the mining process of our proposed HUC-Prune technique. As our tree-structure has the important property of FP-tree stated in property



(a) Conditional-tree for item "e" (b) Conditional-tree for item "a" (c) Conditional-tree for item "d"

Fig. 3. Mining process of HUC-Prune

1, pattern growth mining algorithm can be directly applicable to it by using the *twu* value. Consider the database of Fig. 1. The final tree is created for that database is shown in Fig. 2(c). If we take $\delta = 0.25$ in that database, then *minutil* = 106.75 according to equation 5. As like FP-growth, we have started from the bottom-most item.

At first the conditional tree of the bottom-most item "e" (shown in Fig. 3(a)) is created by taking all the branches prefixing the item "e" and deleting the nodes containing an item which cannot be a candidate pattern (high *twu* pattern) with the item "e". Obviously, item "a" cannot be a candidate itemset with item "e" as it has low *twu* value with the item "e". So, the conditional tree of item "e" does not contain the item "a". However, candidate patterns (1) {b, e: 140}, (2) {d, e: 166}, (3) {e: 204} are generated for the item "e". In the similar fashion, conditional tree for item "a" is created in Fig. 3(b) and candidate patterns (4) {a, b: 257}, (5) {a, d: 176}, (6) {a, b, d: 176}, (7) {a: 257} are generated. After that, conditional tree for item "d" is created in Fig. 3(c), and candidate patterns (8) {b, d: 228}, (9) {d: 308} are generated. The last candidate pattern (10) {b: 347} is generated for the topmost item "b". Third database scan is required to find high utility itemsets from these 10 candidate high twu itemsets. The high utility 6 itemsets are {a, b: 134}, {a, b, d: 146}, {b: 132}, {b, d: 154}, {d: 128} and {d, e: 134}.

5 Experimental Results

To evaluate the performance of our proposed technique, we have performed several experiments on IBM synthetic T10I4D100K dataset and real-life mushroom dataset from frequent itemset mining dataset repository (`http://fimi.cs.helsinki.fi/data/`). These datasets do not provide the profit values or quantity of items in transactions. As like the performance evaluation of the previous utility based pattern mining [4], [6], [7] we have generated random numbers for the profit values of each item and quantity of each item in each transaction, ranging from 0.01 to 10.0 and 1 to 10 respectively. Observed from real world databases that most of the items carry low profit, we generate the profit values using a lognormal distribution [4], [6], [7]. Our programs were written in Microsoft Visual C++ 6.0 and run with the Windows XP operating system on a Pentium dual core 2.13 GHz CPU with 1GB main memory.

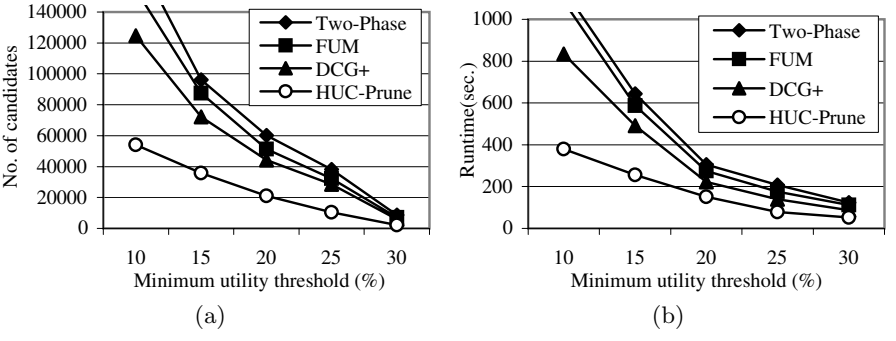


Fig. 4. Comparison on the mushroom dataset (a) Number of candidates (b) Runtime

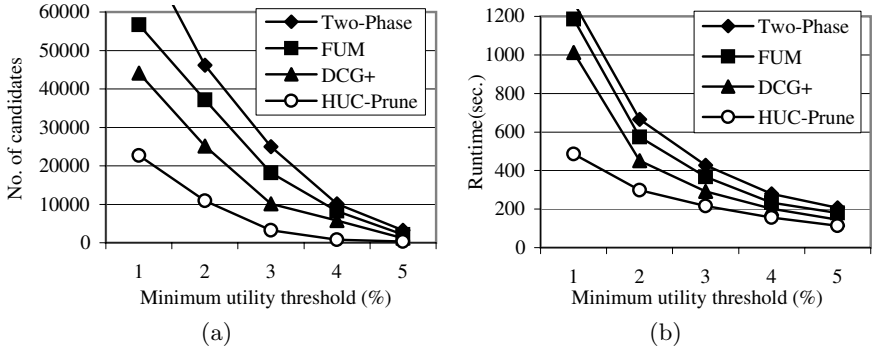


Fig. 5. Comparison on the T10I4D100K dataset (a) Number of candidates (b) Runtime

Mushroom (8,124 transactions, 119 distinct items) is a dense dataset having transaction length 23 for its every transaction. Almost 20% ((23/119) × 100) of its total items are present in every transaction. Dense datasets have too many long frequent as well as high utility patterns. Because of the probability of an item's occurrence is very high in every transaction, the number of candidate patterns and the maximum length of the candidate patterns sharply increase when the minimum threshold decreases in dense datasets. The number of candidates comparison in mushroom dataset is shown in Fig. 4(a). The number of candidates rapidly increases below the utility threshold 20%. For utility threshold 10% and 15% its amount is remarkable larger from our candidate patterns. Fig. 4(b) shows the running time comparison in mushroom dataset. As lower threshold has too many long candidate patterns and several database scans are needed for the huge number of long candidate patterns, time difference between existing algorithms and our technique becomes larger as the δ decreases. So, it is obvious that our technique is better than the existing algorithms in dense datasets.

T10I4D100K (100,000 transactions, 870 distinct items) is a sparse dataset containing average transaction length 10. Sparse datasets normally have too many distinct items. Although in the average case their transactions length is

small, but they normally have many transactions. Handling too many distinct items is a serious problem for level-wise candidate generation-and-test methodology. Therefore, scanning sparse datasets with many candidates several times is a severe problem of the existing algorithms. The number of candidate patterns and runtime comparison in this dataset are shown in Fig. 5(a) and Fig. 5(b) respectively. These figures demonstrate that our technique also outperforms the existing algorithms in sparse datasets.

6 Conclusions

The key contribution of this paper is to provide a very efficient tree-based candidate pruning technique for high utility pattern mining. Our technique prunes a huge number of candidates during tree creation time by eliminating non-candidate single-element patterns and also during mining time by using a pattern growth approach. Its maximum number of database scans is totally independent of the maximum length of candidate patterns. It needs maximum three database scans in contrast to several database scans needed for the existing algorithms. Extensive performance analyses show that our technique is very efficient in high utility pattern mining and it outperforms the existing algorithms in both dense and sparse datasets.

References

1. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In: 20th Int. Conf. on Very Large Data Bases (VLDB), pp. 487–499 (1994)
2. Yao, H., Hamilton, H.J., Butz, C.J.: A Foundational Approach to Mining Itemset Utilities from Databases. In: Third SIAM Int. Conf. on Data Mining, pp. 482–486 (2004)
3. Yao, H., Hamilton, H.J.: Mining itemset utilities from transaction databases. Data & Knowledge Engineering 59, 603–626 (2006)
4. Liu, Y., Liao, W.-K., Choudhary, A.: A Two Phase algorithm for fast discovery of High Utility of Itemsets. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS(LNAI), vol. 3518, pp. 689–695. Springer, Heidelberg (2005)
5. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data Mining and Knowledge Discovery 8, 53–87 (2004)
6. Erwin, A., Gopalan, R.P., Achuthan, N.R.: CTU-Mine: An Efficient High Utility Itemset Mining Algorithm Using the Pattern Growth Approach. In: 7th IEEE Int. Conf. on Computer and Information Technology (CIT 2007), pp. 71–76 (2007)
7. Li, Y.-C., Yeh, J.-S., Chang, C.-C.: Isolated items discarding strategy for discovering high utility itemsets. Data & Knowledge Engineering 64, 198–217 (2008)
8. Tanbeer, S.K., Ahmed, C.F., Jeong, B.-S., Lee, Y.-K.: CP-tree: A tree structure for single pass frequent pattern mining. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS(LNAI), vol. 5012, pp. 1022–1027. Springer, Heidelberg (2008)

Grouped ECOC Conditional Random Fields for Prediction of Web User Behavior

Yong Zhen Guo, Kotagiri Ramamohanarao, and Laurence A.F. Park

Department of Computer Science and Software Engineering
University of Melbourne, Australia
yzguo@csse.unimelb.edu.au

Abstract. Web page prefetching has shown to provide reduction in Web access latency, but is highly dependent on the accuracy of the Web page prediction method. Conditional Random Fields (CRFs) with Error Correcting Output Coding (ECOC) have shown to provide highly accurate and efficient Web page prediction on large-size websites. However, the limited class information provided to the binary-label sub-CRFs in ECOC-CRFs will also lead to inferior accuracy when compared to the single multi-label CRFs. Although increasing the minimum Hamming distance of the ECOC matrix can help to improve the accuracy of ECOC-CRFs, it is still not an ideal method. In this paper, we introduce the grouped ECOC-CRFs that allow us to obtain a prediction accuracy closer to that of single multi-label CRFs by grouping the binary ECOC vectors. We show in our experiments that by using the grouping method, we can maintain the efficiency of the ECOC-CRFs while providing significant increase in Web page prediction accuracy over ECOC-CRFs.

Keywords: Web Page Prediction, Conditional Random Fields, Error Correcting Output Coding, Grouping.

1 Introduction

Many Internet users have turned to wireless devices such as mobile phones or PDAs due to their mobility and convenience, even though the connection speed of these wireless devices are usually slow. Meanwhile, there are still many dial-up users with low bandwidth access to the Internet. On account of the limited bandwidth and low-speed connections, many wireless and dial-up Internet users need to spend long periods of time waiting for the requested Web pages to be transferred to them through Internet, which may lead to intolerable delays.

Web page *prefetching* is an effective way to reduce the access latency for Web users. However, if most of the prefetched Web pages are not visited by the users in their subsequent accesses, the limited network bandwidth and server resources will not be used efficiently, and hence may worsen the access latency problem. Therefore, the success of a prefetching method relies heavily on the Web page prediction accuracy.

Conditional Random Fields (CRFs) [1] are powerful probabilistic framework for classifying sequential data. Although the training complexity of a CRF is

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 757–763, 2009.
© Springer-Verlag Berlin Heidelberg 2009

very high, by using Error Correcting Output Coding (ECOC) [4] to decompose a multi-label CRF training into many binary-label sub-CRF trainings, the overall training complexity can be decreased. Therefore, ECOC-CRFs can be efficiently used on data sets containing many unique labels and are ideal for predicting an Internet user's future Web access patterns on large scale websites [5]. However, because the class information given to each sub-CRF is drastically decreased, the prediction accuracy of ECOC-CRFs is inferior to that of single multi-label CRFs. In this paper, we propose the grouped ECOC-CRFs, which can maintain the advantages of ECOC-CRFs while yielding a higher accuracy, thus can be used to improve the performance of ECOC-CRFs based Web page prediction further.

The rest of this paper is organized as follows: in Section 2, we briefly describe the ECOC-CRFs for Web page prediction, which is followed by the comparison of ECOC-CRFs and single CRFs in Section 3. We propose our novel grouped ECOC-CRFs to improve the prediction accuracy in Section 4 with experiments. Finally, we conclude this paper in Section 5.

2 ECOC-CRFs for Web Page Prediction

The complexity of a CRF training is proportional to the square of the number of labels (in Web page prediction scenario, unique Web pages), when used on large-scale websites containing thousands of unique Web pages, the CRF training will become highly resource intensive and even infeasible. If we can decompose a multi-label CRF training into a series of sub-CRF trainings, each with much fewer labels, the total training complexity will be reduced significantly. Error correcting output coding (ECOC) can be used to decompose a multi-label classification task into a set of binary-label classification sub-tasks [3], it consists of two steps: firstly, construct a code matrix for the original multi-label classification task and train a series of binary sub-classifiers for each column of the code matrix; secondly, combine the decoding results of each binary sub-classifier to obtain the possible result for the original classification task. Therefore, we can employ ECOC to decompose a multi-label CRF training into many binary-label sub-CRF trainings. Since all the sub-CRFs are binary, they can be trained very efficiently: an ECOC-CRF training can be completed in a fraction of the time of a single multi-label CRF training. In addition, all sub-CRFs can be trained in parallel to save the overall training time. It has been shown in [5] that we are able to efficiently predict Web pages that a user may access using ECOC-CRFs in large-size websites, in [5] it also explains how to use *Search Coding* to design an ECOC code matrix with good row separation.

Although by using ECOC-CRFs the total training complexity can be reduced, the class information given to each sub-CRF is drastically decreased as well. As a result, the prediction accuracy of ECOC-CRFs is lower than that of the single multi-label CRFs, which can be seen from the experimental results in next section. In this paper, we propose the grouped ECOC-CRFs to improve the accuracy of ECOC-CRFs based Web page prediction, while maintain the advantages of ECOC-CRFs.

3 Comparison of ECOC-CRFs and Single CRFs

In this section, we evaluate the Web page prediction performance of ECOC-CRFs and single CRFs on the publicly accessible *msnbc* dataset [6]. In this dataset, all the user visits are recorded in session format at the level of page categories. There are 17 different page categories which can also be treated as 17 distinct pageviews. We randomly selected 50,000 distinct sessions with length more than 5 and less than 100 from the preprocessed dataset and divided them into two subsets: 45,000 sessions for training and 5,000 for testing.

For the implementation of CRF training, we use the CRF++ toolkit [8]. We used three CRF++ feature templates in our experiments. In the first template (referred to as CRF1), we define the current and previous one observation and their combination as the unigram features; for the second template (CRF2), we use the current and previous two observations and their combinations as the unigram features; we define the third template (CRF4) similarily. These templates share a same bigram feature, which will automatically generate the combination of the current label and previous label as the feature functions. The more abundant features in use, the more powerful CRF models can be obtained.

As a baseline, we trained 3 single multi-label CRFs on the *msnbc* dataset using the feature functions of CRF1 ,CRF2 and CRF4, and evaluated their page prediction accuracies by calculating the ratio of the number of correctly predicted labels to the total number of predicted labels. We also implemented the ECOC-CRFs by using *Search Coding* to design the code matrix. Figure 1 shows the accuracies of single multi-label CRFs and their corresponding ECOC-CRFs with ECOC code matrix parameters: code word length $n = 16$ and minimum Hamming distance between each code word $d = 8$, in which the prediction accuracies of the 1st and 2nd-order Markov Chain (referred to as 1st-MC and 2nd-MC respectively) are shown for comparison.

From Figure 1 we can see that all single CRFs produce much higher prediction accuracy than Markov Chains, indicating the superiority of CRFs to Markov Chains in predicting labels of sequences. We can also see that the accuracy of an ECOC-CRF is lower than its corresponding single multi-label CRF. This

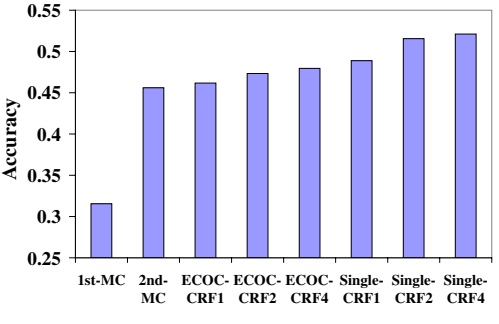


Fig. 1. Web page prediction accuracy of Markov Chains, single CRFs and ECOC-CRFs with code length $n = 16$ and minimum Hamming distance $d = 8$ on the *msnbc* dataset.

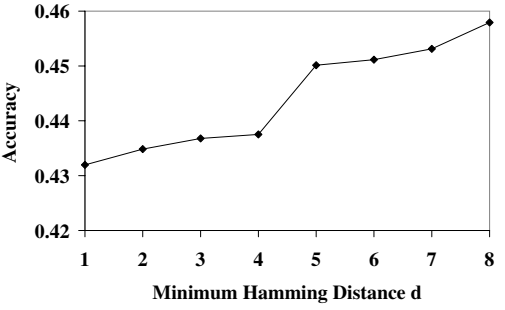


Fig. 2. Accuracy of ECOC-CRF1 with different ECOC code matrices on the *msnbc* dataset

reduction in accuracy is due to the limited class information provided to the binary sub-CRFs when compared to the single multi-label CRFs. However, we will show later that by using several approaches, the prediction accuracy of ECOC-CRFs can be improved further.

By choosing the ECOC code matrix with a larger minimum Hamming distance d, we can increase the number of sub-classification errors that can be corrected. Therefore, we can improve the accuracy of ECOC-CRFs by using an ECOC code matrix with larger minimum Hamming distance. We show this by an experiment on the *msnbc* dataset, in which we design a series of ECOC code matrices, whose minimum Hamming distances d are ranging from 1 to 8 and code lengths n vary from 5 to 16 accordingly(the minimum code length that satisfies the given minimum Hamming distances d and the number of code words m=17 by using Search Coding). Then we evaluated the accuracies of ECOC-CRF1 by using these 8 ECOC code matrices, the results in Figure 2 show that the prediction accuracy of ECOC-CRF1 increases as d increases.

However, as the minimum Hamming distances d increases, the length of code words n increases as well. Although long code words can lead to good row-separation, they increase the chance that two matrix columns are similar to each other as well, in which case the two corresponding sub-classifiers will learn similar concepts. Furthermore, the increase in the length of code words will also result in the increase in the number of sub-classifiers to be trained, which will add up the overall training time. Therefore, improve the accuracy of ECOC-CRFs by simply enlarging the minimum Hamming distances d is not an ideal method, we should find a tradeoff between the length of ECOC code words and accuracy. In the following section we will propose a grouping method which can improve the prediction accuracy of ECOC-CRFs without adding the length of ECOC code words.

4 ECOC-CRFs with Grouping

By decomposing a multi-label CRF classification problem into many independent binary classification problems, we achieved faster training times, but we

also drastically reduced the class information given to each sub-CRF. If we can provide each sub-CRF with more abundant class information, the prediction accuracy can be improved. This leads to the consideration of grouping, which divides the columns of the ECOC code matrix into several groups and uses each group to train a sub-CRF. For example, an ECOC code matrix containing 12 columns can be divided into 4 groups containing 3 columns each, in which case the grouping size (the number of possible labels for each sub-CRF to learn) is $2^3 = 8$.

By grouping, the number of sub-CRFs needed will be cut down from n to $\lceil n/\log_2 G \rceil$, where n is the length of ECOC code words and G is the grouping size; meanwhile, the number of possible labels for each sub-CRF to learn will increase from 2 to G. Therefore, the training complexity can be reduced from $O(L^2)$ to $O(\lceil n/\log_2 G \rceil \times G^2)$, where L is the total number of labels. When the grouping size increases, each sub-CRF will obtain more refined information about the class labels, and hence, the accuracy should also increase. A grouping size of L (meaning all the columns are put into one group) implies that the ECOC-CRFs will reduce to single CRFs. An example grouping process is shown in Figure 3.

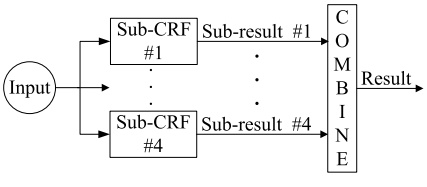


Fig. 3. Example grouping process when code length $n = 8$ and grouping size $G = 4$ (2 columns per group), therefore 4 sub-CRFs are needed

We performed experiments on the *msnbc* dataset to measure the effect of grouping on ECOC-CRF4 using two different experimental setups: (1) code length $n = 8$, minimum Hamming distance $d = 2$; and (2) $n = 16$, $d = 8$. These results can be found in Figure 4, from which we can see that as the grouping size of ECOC-CRFs grows, the overall trend of prediction accuracy increases. When the grouping size becomes big enough to include all L labels, the ECOC-CRFs can perform as well as the single multi-label CRFs. Therefore, by incorporating grouping, the accuracy of ECOC-CRFs can be improved further without increasing the code length.

In Figure 4 we also illustrate the change in training time of ECOC-CRFs (measured by adding the training time of all the sub-CRFs) versus the grouping size using the identical experimental setups. We can see that when the grouping size becomes larger, the total training time of ECOC-CRFs will increase and even exceed the training time of single multi-label CRFs. This is because the number of labels that a sub-CRF needs to learn increases as the grouping size increases, which in turn increases the training time dramatically, and since there are numerous sub-CRFs to be trained, the total training time can be longer than

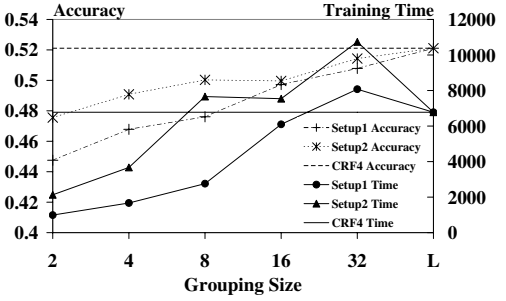


Fig. 4. Relationships between accuracy, training time (in seconds) and grouping size of Setup1 ($n = 8$, $d = 2$) and Setup2 ($n = 16$, $d = 8$) on *msnbc* dataset, where the accuracy and training time of single CRF4 are shown as a baseline. When grouping size is L, ECOC-CRFs reduce to single CRFs.

that of single CRFs. Therefore, it is important to select an appropriate grouping size to provide high accuracy and fast training.

We also conducted experiments on the CSSE dataset [7] which contains $3,829$ unique Web pages to evaluate the scaling performance of grouped ECOC-CRFs on Web page prediction. After preprocessing, we randomly select $2,723$ sessions as the training data and 544 sessions as the testing data. We conducted the experiments on this dataset by using different grouping sizes on two experimental setups with ECOC-CRF1: (1)code number $m = 3,829$, code length n=16, minimum Hamming distance d=1, and (2) $m = 3,829$, n=24, d=8. We recorded the prediction accuracy and the training time for different grouping size in Table 1. Due to the large number of labels ($3,829$ unique Web pages) and the limitation of our computational resources, the training of single multi-label CRFs and ECOC-CRFs with grouping size of L (in this case the grouped ECOC-CRFs will reduce to single CRFs) on this dataset are infeasible.

Table 1. Prediction accuracy and training time of Setup1 ($m = 3,829$, n=16, d=1) and Setup2 ($m = 3,829$, n=24, d=8) on CSSE dataset

Grouping Size	Setup1		Setup2	
(Label Numbers)	Accuracy	Time	Accuracy	Time
2	64.0%	51s	69.2%	75s
4	63.9%	63s	70.0%	84s
8	64.8%	108s	70.4%	146s
16	64.7%	267s	70.7%	343s
32	66.6%	736s	72.2%	942s
64	68.2%	1,995s	74.0%	3,815s
128	70.1%	8,151s	74.9%	12,042s
256	72.9%	35,302s	75.0%	49,629s
$3,829$(Single CRF)	infeasible		infeasible	

From Table 1 we can observe that on the CSSE dataset, with a bigger d, ECOC-CRFs can yield more accurate Web page prediction accuracy. We can also see that the grouping method can help to improve the accuracy of ECOC-CRFs significantly. When we compare the accuracy of grouping size of 256 and grouping size of 2 (no grouping), the improvements in accuracy are 8.9% for Setup1 and 5.8% for Setup2 respectively. Additionally, while the grouping size enlarges, the training time of both setups increase dramatically, and it is infeasible to train a single multi-lable CRF on this dataset. Therefore, we can use grouped ECOC-CRFs to obtain satisfactory Web page prediction performance in large scale websites.

5 Conclusion

By using ECOC-CRFs to decrease the training complexity we also reduce the class information provided to each sub-CRF, thus the prediction accuracy of ECOC-CRFs is lower than the corresponding single CRFs. In this paper, we proposed the grouped ECOC-CRFs. By dividing the columns of ECOC matrix into several groups, each sub-CRF can learn more class information which will lead to higher accuracy. Using grouped ECOC-CRFs we are able to obtain a prediction accuracy closer to that of single multi-label CRFs while maintain the advantages of ECOC-CRFs. Our experiments have shown that the grouped ECOC-CRF Web page prediction is highly accurate and scalable, and hence ready for use with large scale Web sites.

References

1. Lafferty, J., McCallum, A., Pereir, F.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data, In: Proceedings of the 18th International Conference on Machine Learning, pp. 282–289 (2001)
2. Srivastava, J., Cooley, R., Deshpande, M., Tan, P.N.: Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. SIGKDD Explorations 1, 12–23 (2000)
3. Berger, A.: Error-correcting output coding for text classification. In: Proceeding of IJCAI: Workshop on machine learning for information filtering (1999)
4. Dietterich, T., Bakiri, G.: Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research (1995)
5. Guo, Y.Z., Ramamohanarao, K., Park, L.: Error Correcting Output Coding-Based Conditional Random Fields for Web Page Prediction. In: Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 743–746 (2008)
6. UCI KDD Archive, `http://kdd.ics.uci.edu/databases/msnbc/msnbc.html`
7. The Department of Computer Science and Software Engineering, University of Melbourne, `http://csse.unimelb.edu.au`
8. CRF++: Yet another CRF toolkit, `http://crfpp.sourceforge.net`

CLHQS: Hierarchical Query Suggestion by Mining Clickthrough Log

Depin Chen[1,*], Ning Liu[2], Zhijun Yin[3], Yang Tong[2], Jun Yan[2], and Zheng Chen[2]

[1] University of Science and Technology of China
[2] Microsoft Research Asia
[3] University of Illinois at Urbana-Champaign
charjoe@ustc.edu.cn, {ningl, i-yatong, junyan, zhengc}@microsoft.com, zyin3@uiuc.edu

Abstract. Most commercial search engines provide query suggestion in a ranked list for more effective search. However, a ranked list may not be an ideal way to satisfy users' various information demands. In this paper, we propose a novel query suggestion method named CLHQS (Clickthrough-Log based Hierarchical Query Suggestion). It organizes the suggested queries into a well-structured hierarchy. Users can easily generalize, extend or specialize their queries within the hierarchy. The query hierarchy is mined from the clickthrough log data in the following way. First, we generate a candidate set through the query-url graph analysis. Second, the *pair-wise* relationships are inspected for each pair of candidate queries. Finally, we construct the suggested query hierarchy using these relationships. Experiments on a real-world clickthrough log validate the effectiveness of our proposed CLHQS approach.

Keywords: Query Suggestion, Clickthrough Log, Hierarchy.

1 Introduction

Query suggestion has been widely used in commercial search engines (Google [1], Live Search [2], Yahoo [3] etc.) to improve user experience. When a user submits a query, the search engine returns the search results, together with a small number (up to 8 by Google [1]) of suggested queries if available. The suggested queries are usually laid at the top or bottom of the result page.

Query suggestion has been studied intensively in research community. Traditional algorithms are based on content similarity between the originated query and the candidate query. These methods are limited in computational cost and also do not take user feedback information into account. Many following works [4, 5, 7, 9, 11] make use of the clickthrough logs. Some of them [4, 5, 7] are based on query-url graph or session analysis. Other works [11] combine the log-based method and the content-based method. What is common in these methods is that they all adopt a ranked query list as output and try to put the most appealing ones on top. We argue that a single list composing of a small number of queries may not be suitable, since users may have

[*] This work is done during the internship of the first author at Microsoft Research Asia.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 764–771, 2009.
© Springer-Verlag Berlin Heidelberg 2009

various information demands during search and hence different types of suggested queries (generalization, extension and specialization) are needed.

In this paper, we propose a novel method CLHQS, aiming to facilitate the usage of query suggestion and improve user experience. The rest of our paper is organized as follows. In Section 2, we describe the query suggestion problem. Section 3 introduces our method for query suggestion in detail. Then in Section 4, we report our experimental results, followed by Section 5 which concludes this paper.

2 Problem Description

We summarize three categories of suggested queries according to different kinds of user intentions.

(1) Generalization. The user only knows a specific expression, while wants to get information about a general related concept. For example, a user would like to search information about the manufacturer of the software suite *"office"*. Since the user does not know the existence of *"microsoft"*, he uses *"office"* as the query. If the search engine could suggest query *"microsoft"* and tell the user that it is a general concept of *"office"*, it would accelerate the search process and improve user experience.

(2) Specialization. Conversely, in some cases, the user only knows the general concept, while wants to get information about a specialized thing. For example, the user searches *"google"* to find out the services provided by Google [1]. If the search engine could suggest queries such as *"Google Earth"* and *"Google Image"* that are the sub-concepts of *"google"*, the user can easily find the most interesting service to him.

(3) Extension. Besides the two scenarios above, users may also be interested in some related queries which belong to the same concept with the originated query. For example, users searching *"xbox"* may be also interested in information about *"ps3"*, since both of them are within the concept of *"game player"*.

Observing this, we propose to present the suggested queries in a hierarchy structure instead of a ranked list. Doing this, users can get an overview of the suggested queries. Higher level indicates more general queries, and lower level indicates more specialized ones. Users can explore the hierarchy and locate the ideal query at ease.

3 Clickthrough-Log Based Hierarchical Query Suggestion

3.1 Candidate Set Generation

To generate the candidate set, we adopt the assumption that related queries are likely to share some common clicked urls in the clickthrough log data [4]. Fig. 1 gives the flow chart of candidate set generation for a specific query q.

First, find all the clicked urls after query q. Sort these urls by the click times: $(u_1, f_1), (u_2, f_2), ..., (u_n, f_n)$, where $f_1 \geq f_2 \geq ... \geq f_n$. To determine the size of frequent url set, we define $H(i)$, which can empirically produce reasonable frequent urls. We calculate $H(i)$ from $i=1$ increasingly till $H(k+1)-H(k)<\varepsilon$. The top k mostly clicked urls $u_1, u_2, ..., u_k$ are selected as frequent urls.

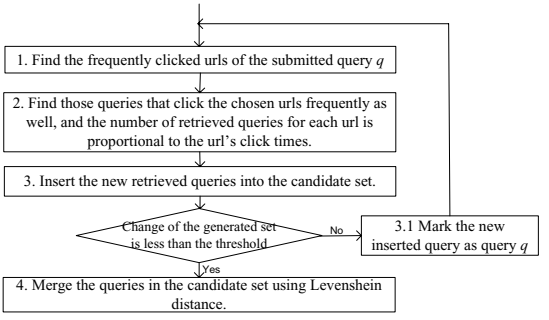


Fig. 1. The flow chart of candidate set generation

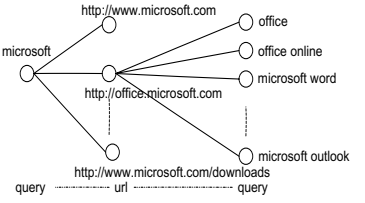


Fig. 2. The query-url graph of *"microsoft"*

$$H(i)=\left|\log\frac{\sum_{l=1}^{i}f_i}{\sum_{l=1}^{n}f_i}\right|+\left|\log\frac{\sum_{l=i+1}^{n}f_i}{\sum_{l=1}^{n}f_i}\right| \tag{1}$$

Second, for each of the *k* frequent urls u_i, we retrieve n_i queries which also clicked u_i frequently and add them into the candidate set. Here, n_i is proportional to the click times f_i. Fig. 2 gives an example in which we add query *"office"* into the candidate set of query *"microsoft"*.

Third, we find the related queries recursively with the candidate set getting larger and larger until certain stop condition is met. Finally, we merge some redundant queries in the candidate set. For instance, the candidate set of query *"microsoft"* may contain *"microsoft.com"*, *"www.microsoft.com"* and *"mircosoft"*. It is notable that *"mircosoft"* is a misspelled version of *"microsoft"*. These queries mean exactly the same with *"microsoft"* and should be merged. We use the Levenshtein distance [6] to automatically detect queries which should be merged. Levenshtein distance is used to compute the similarity between two strings in information theory. It is defined as the minimum number of operations needed to transform one string into the other, where an operation can be insertion, deletion, or substitution of a single character. For example, *L*(*"hierarchy"*,*"hierarchical"*) = 4. We use a normalized version of Levenshtein distance,

$$D(a,b)=\frac{L(a,b)}{\max(Length(a),Length(b))} \tag{2}$$

We merge query *a* and query *b*, if and only if $D(a,b)<\sigma$, where σ is a predefined threshold. Query *a* and query *b* will be merged into the more frequent one between them.

3.2 Pair-Wise Relationship Building

After getting the candidate set, we examine the pair-wise relationships between each two queries in the set. We find out whether there is a *parent-child* relation between two queries. The parent is a general concept of the child.

Intuitively, the frequent urls of a parent query should subsume the ones of a child query. Suppose there are two queries, query a and query b, together with their frequent urls,

$$U_a = \{(u_{a1}, f_{a1}), (u_{a2}, f_{a2}), ..., (u_{am}, f_{am})\} \; U_b = \{(u_{b1}, f_{b1}), (u_{b2}, f_{b2}), ..., (u_{bn}, f_{bn})\} \quad (3)$$

First, we normalize f_{a*} and f_{b*} as follows,

1. For each url-frequency pair (u_{ai}, f_{ai}) in U_a,

$$f_{ai} \leftarrow \frac{f_{ai}}{\sum_{j=1}^{m} f_{aj}}$$

2. For each u_{ai},

if (u_{ai} denotes the same url as u_{bj} in U_b)

$$f_{ai} \leftarrow (f_{ai} + f_{bj})/2$$

else

$$f_{ai} \leftarrow f_{ai}/2$$

3. Apply 1 and 2 on U_b.

Then, we compute the $P(a \mid b)$ and $P(b \mid a)$, which represent how probable query a subsumes query b and query b subsumes query a, respectively.

$$P(a \mid b) = \frac{\sum_{i|u_i \in U_a \cap U_b} f_{bi}}{\sum_{i=1}^{n} f_{bi}}, \qquad P(b \mid a) = \frac{\sum_{i|u_i \in U_a \cap U_b} f_{ai}}{\sum_{i=1}^{m} f_{ai}} \quad (4)$$

As in [11], we judge that query a subsumes query b if $P(a|b)>\theta$ and $P(a|b)$-$P(b|a)>\alpha$. For two query a and query b having little overlap, we first merge them into a merged query *merge*(a,b). Then we indirectly examine their relationship by considering the relationship between query a and query *merge*(a,b) and between query b and query *merge*(a,b) respectively. If query a is a parent of query b, the meaning of *merge*(a,b) should be much close to b than a, and a should still be a parent of *merge*(a,b). For example, we merge query *"harvard"* and *"university"* into *"harvard university"*. Query *"harvard university"* has the same meaning of *"harvard"* and is a child of *"university"*. Therefore, *"university"* is a parent concept of *"harvard"*. The predicate is: query a is the parent concept of query b only if $P(merge(a,b)|b)>\theta_m$ and $P(merge(a,b)|b)$- $P(merge(a,b)|a)>\alpha_m$, and vice versa. If none of these conditions is satisfied, we consider that there is no subsumption relationship.

3.3 Query Hierarchy Construction

In this step, we first directly generate the hierarchy from the pair-wise relationships. Then, we trim and merge redundant edges conforming to the following two rules.

Rule 1. Query a_1 subsumes query a_2, query a_2 subsumes query a_3,…, query a_{n-1} subsumes query a_n. Meanwhile, query a_1 subsumes query a_n. If a_{i+1} is the substring of a_i for all $1 < i <$ n-1, trim the edge between a_1 and a_n. If not, do not trim the edge.

Rule 2. Query a subsumes query b and query a subsumes query c. If b is the substring of c and a is the substring of c, then move the children of c under the query a, remove the edge between a and c and remove query c.

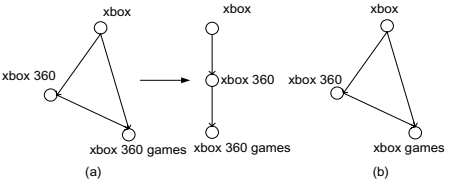


Fig. 3. Two examples of **Rule 1**. The graph in **(a)** can be trimmed and the one in **(b)** cannot be trimmed.

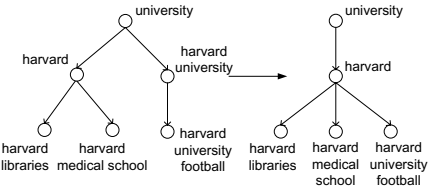


Fig. 4. Example of **Rule 2**. Merge *"harvard university"* into *"harvard"*

Fig. 3 and 4 give examples of the two rules. In Fig. 3(a), the edge between *"xbox"* and *"xbox 360 games"* can be removed, since it conforms to Rule 1. In Fig. 3(b), edge between *"xbox"* and *"xbox games"* should not be trimmed, since *"xbox 360"* is not the substring of *"xbox games"*. In Fig. 4, query *"university"* subsumes query *"harvard"* and query *"harvard university"*, and *"harvard"* and *"harvard university"* have the same meaning. Hence we merge them into *"harvard"* according to Rule 2.

4 Experiments

We design a series of experiments to demonstrate the effectiveness and efficiency of our method. The dataset used in our experiments is randomly sampled by 5% from a three-month clickthrough log (Oct. 2006 ~ Dec. 2006), which is collected by a commercial search engine.

4.1 Candidate Set Evaluation

In order to evaluate the quality of the generated candidate set, we recruit five volunteers to do the labeling work. 100 queries are randomly selected from the top 10% queries with highest click frequency. We generate 20 candidate queries for each of them using our method. The volunteers are asked to label each candidate query as "relevant" or "not relevant" to the originated query. It is notable that a candidate query with no concept overlap, containing redundant information or misspelling terms will be labeled as "not relevant". The final label for each candidate query is decided by majority voting among the five volunteers.

Finally, we calculate the precision, recall, $MacroF_1$ and $MicroF_1$ [10] with different σ and depict the results in Fig. 5. The precision exceeds 0.9 when we merge the queries in the candidate set using Levenshtein distance [6] with the threshold $\sigma \geq 0.1$. In the following, we set σ to be 0.1 to get optimal performance.

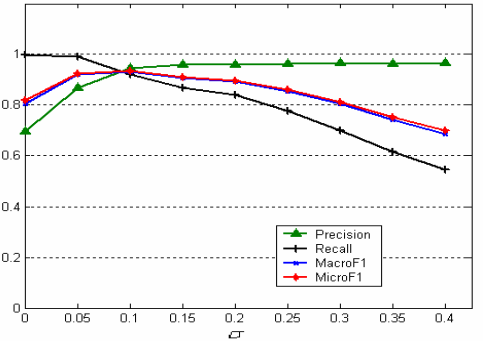


Fig. 5. The impact on precision, recall, $MacroF_1$ and $MicroF_1$ by the threshold σ

4.2 Pair-Wise Relationship Evaluation

In this section, we compare the pair-wise relationship calculated by our method CLHQS with the ones generated by the Substring method. The assumption of the Substring method is that query *a* subsumes query *b* if query *a* is the substring of query *b* for the reason that there is more constraint on *b* than on *a*.

In our method CLHQS, we set the parameters as $\sigma = 0.1$, $\theta = \theta_m = 0.8$, $\alpha = \alpha_m = 0.2$ and calculate the pair-wise relationship on the candidate set generated for the 100 queries in Section 4.2. We again ask the five volunteers to label the pair-wise subsumption relationships and the final label is generated by majority voting among them. To evaluate the performance of CLHQS, we calculate the precision, recall and F_1. Precision P is the percentage of correct ones labeled by people. Recall R is calculated as follows. $C_{substring}$ is the correct pairs generated by substring method. C_{CLHQS} is the correct pairs generated by CLHQS.

$$R_{substring} = \frac{|C_{substring}|}{|C_{substring} \cup C_{CLHQS}|} \qquad R_{CLHQS} = \frac{|C_{CLHQS}|}{|C_{substring} \cup C_{CLHQS}|} \tag{5}$$

Table 1 summarizes the evaluation result. Although the Substring method reaches higher precision, it can only find the straightforward subsumption relationships when omitting lots of meaningful ones. Our method CLHQS outperforms the Substring method in recall and F_1 measures. Moreover, 48.80% of the subsumption relationships found by our method CLHQS are not the substring cases.

Table 1. CLHQS vs. Substring model

	Precision	Recall	F1
Substring	0.927	0.575	0.710
CLHQS	0.795	0.871	0.831

Table 2. Query distribution by the *HSTEP/LSTEP* value

	[0, 0.25)	[0.25,0.5)	[0.5, 0.75)
Behavior1	11	9	10
Behavior2	23	8	12
	[0.75,1]	**1**	**>1**
Behavior1	48	6	16
Behavior2	30	23	4

3 Hierarchy vs. Ranked List

There are two main kinds of user behavior when served by query suggestion.

User behavior 1. The user would like to read through the queries first and then make the decision. For the ranked list, users read through the queries to the end of the list and then choose the favorite. For the hierarchy, users read all the queries in the peer level and then decide which query to continue their exploration (drill down, level up or stop).

User behavior 2. The user would like to stop once an interesting query is located. For the ranked list, the user stops whenever he or she meets interesting query. For the hierarchy, the user reads the queries in the same level one by one and stops whenever encountering the interesting query. Then the user drills down/level up from this query, or stops the exploration.

In the candidate set, we build the hierarchy structure with n queries $\{q_1, q_2, ..., q_n\}$, whose score is ranked in descending order. For the ranked list, behavior 1: $LSTEP1 = n * n / n = n$, behavior 2: $LSTEP2 = (1 + 2 + ... + n)/n = (n+1)/2$. For the hierarchy, $HSTEP1$ and $HSTEP2$ represent the average length of the paths from the start query to destination queries.

Again, we randomly select 100 queries from the top 10% queries and set $\sigma = 0.1$, $\theta = \theta_m = 0.8$, $\alpha = \alpha_m = 0.2$. For each query q_i, $LSTEP1$, $LSTEP2$, $HSTEP1$ and $HSTEP2$ are calculated. The distribution of $HSTEP1/LSTEP1$ and $HSTEP2/LSTEP2$ are reported in Table 2. We find that, for most queries, users need fewer steps in the hierarchy than in the ranked list. T-test [8] shows that the improvements are statistically significant (p-value < 0.05). Taking the 100 queries as a whole, we also calculate the following values.

$$Percentage_{global} = \frac{\sum HSTEP_i}{\sum LSTEP_i} \quad Percentage_{local} = \frac{1}{n}\sum\left(\frac{HSTEP_i}{LSTEP_i}\right) \tag{6}$$

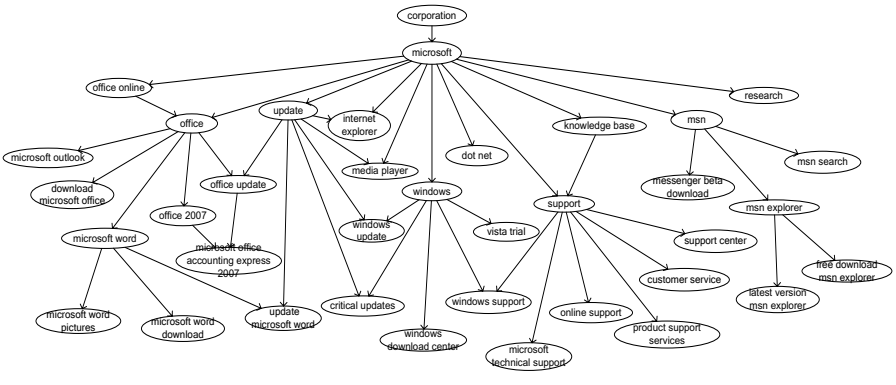


Fig. 6. The hierarchy structure of query *"microsoft"*

For user behavior 1, $Percentage_{local} = 79.60\%$ and $Percentage_{global} = 72.62\%$. For user behavior 2, $Percentage_{local} = 67.47\%$ and $Percentage_{global} = 52.60\%$. The user effort needed to find interesting queries is reduced significantly with the suggested query hierarchy.

4.4 Case Study

In order to show the practical use of our method, we take query *"microsoft"* as case study. We generate the top 100 related queries for *"microsoft"* and build the hierarchy based on them. Here, we choose $\theta = \theta_m = 0.8$, $\alpha = \alpha_m = 0.2$. Due to space restriction, we only demonstrate the main part of the hierarchy in Fig. 6.

5 Conclusion

In this paper, we propose a novel method CLHQS, to generate the hierarchy structure of query suggestion through mining clickthrough log data. Hierarchical query suggestion is more expressive than the ranked list. Besides, the use of clickthrough log data assures that the suggested queries are extracted from real user behaviors, which is more reasonable than the queries suggested based on term match.

References

1. Google Search, `http://www.google.com`
2. Live Search, `http://search.live.com`
3. Yahoo Search, `http://search.yahoo.com`
4. Brian, D.D., David, G.D., David, B.L.: Finding Relevant Website Queries. In: Proceedings of the 12th International World Wide Web Conference (WWW 2003), Budapest, Hungary, pp. 162–168 (2003)
5. Chien-Kang, H., Lee-Feng, C., Yen-Jen, O.: Relevant Term Suggestion in Interactive Web Search Based on Contextual Information in Query Session Logs. Journal of the American Society for Information Science and Technology 54(7), 638–649 (2003)
6. Eric, S.R., Peter, N.Y.: Learning String-edit Distance. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(5), 522–532 (1998)
7. Huanhuan, C., Daxin, J., Jian, P.: Context-aware Query Suggestion by Mining Click-through and Session Data. In: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, Las Vegas, Nevada, USA, pp. 875–883 (2008)
8. James, T.M., Terry, S.: Statistics, 10th edn. Prentice Hall, Englewood Cliffs (2006)
9. Silviu, C., Ryen, W.W.: Query Suggestion Based on User Landing Pages. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, Amsterdam, The Netherlands, pp. 875–876 (2007)
10. Yiming, Y.: An Evaluation of Statistical Approaches to Text Categorization. Journal of Information Retrieval 1(1-2), 67–88 (1999)
11. Zhiyong, Z., Olfa, N.: Mining Search Engine Query Logs for Query Recommendation. In: Proceedings of the 15th International World Wide Web Conference (WWW 2006), Edinburgh, Scotland, pp. 1039–1040 (2006)

X-Tracking the Changes of Web Navigation Patterns

Long Wang and Christoph Meinel

Hasso Plattner Institut, University Potsdam
14482, Potsdam, Germany
long.wang@hpi.uni-potsdam.de

Abstract. In this paper, we firstly investigate the possible changes on web usage behaviors and then propose an x-tracking method to detect these changes. The changes on web navigation patterns are depicted from microscopic and macroscopic levels: the former is for the "internal" and "external" changes which show the variations on the semantics and external physical features respectively, while the latter is modeled for the changes of the popularity on "local" and "global" time line. The x-tracking method we propose is to detect the newly emerged patterns (EP) based on "internal" feature, which is the premise to compute the changes on other features by tracking their internal unchanged patterns (IUP). Experiments show that the x-tracked changes are condensed, efficient and informative.

1 Introduction

Web searching and navigation activities have been widely discussed, and various techniques are used to mine types of navigation patterns. However, discovering the evolutions and representatives hidden in patterns from different time spans is always challenging. Some efforts are paid on detecting structural changes of web usage behaviors [1,3,8]. On the other hand, conceptual similarity computing has been deeply discussed in text mining, duplication deletion in XML and extracting representatives from frequent patterns[2,9,6].

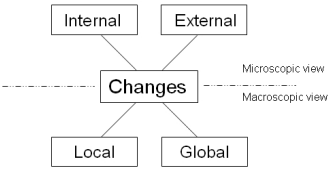


Fig. 1. X-tracking Changes

The goal of detecting changes in web navigation patterns is to discover the differences in the compositions of content and structure, and the variations of

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 772–779, 2009.
© Springer-Verlag Berlin Heidelberg 2009

the underlying populations. We propose an x-tracking method to detect the changes of web navigation patterns. The changes are described from two layers in an x-shape shown in figure 1. On microcosmic layer, we define two kinds of changes:

- ***Internal Change***: refers to the internal difference synthesizing the content and structure of a pattern in different time spans.
- ***External Change***: considers the varying of the external form of a single pattern.

On macroscopic layer, the changes of underlying populations are depicted from two aspects:

- ***Local Change***: concerns the changing of popularity of a single pattern between two time spans.
- ***Global Change***: computes the changes of popularity over the whole time line.

The rest of the paper is organized as: section 2 gives the formalization of changes and x-tracking method; we show the experiment results and performance discussion in section 3; the paper ends with the conclusion and future work in section 4.

2 X-Tracking the Changes and Its Decompositions

In this section, we will give our x-tracking strategy to detect the changes of navigation patterns. We firstly define the basic terms used in the following sections:

- t_i: the i^{th} time span along the entire time line T, and $T = \{t_0, ..., t_n\}$;
- U_i: the usage data collected at t_i;
- ζ_i: the set of navigation patterns discovered at t_i based on a defined model;
- X: one navigation pattern, and X_i is the pattern in ζ_i mined at t_i.

2.1 Measuring Internal Change

Edit distance is widely used in computing structural similarity ranging from string-to-string difference [7] to tree structure comparison [9]. Originally, *edit distance* between two structures is computed based on the smallest sum of cost of the basic edit operations that change one structure to the other. The basic edit operations are often defined as *insertion*, *deletion* and *updating*.

In order to amend precision and accuracy of pattern changes, we integrate the semantic distance in the structure-based edit distance. Semantic distance that one node changes to the other, or the distance between two versions of the same labeled node, is used in our algorithm as the cost for the basic changing operations. As explained in section 2.4, each web document d is represented by a term vector composed by the extracted terms with their weights. Computing similarity recursively has been explored in the specific context of database schema-matching. The *semantic distance* between two web documents d_1 and d_2 from

two patterns in different spans is computed as: $sDist(d_1, d_2) = 1 - sSim(d_1, d_2)$, where $sSim(d_1, d_2)$ is the semantic similarity between d_1 and d_2. We use the cosine to compute the semantic similarity between two vectors of d_1 and d_2: $sSim(d_1, d_2) = \frac{\sum_{i=1}^{K} w_{d_1,i} \cdot w_{d_2,i}}{\sqrt{\sum_{i=1}^{K} w_{d_1,i}^2} \cdot \sqrt{\sum_{i=1}^{K} w_{d_2,i}^2}}$.

Definition 1 (Internal Distance). *Let X_1 and X_2 be two navigation patterns, the* internal distance *$iDist(X_1, X_2)$ between X_1 and X_2 is computed as* edit distance *integrated with the "*semantic distance*" between the corresponding web pages from these two patterns, in which the* cost *of every basic edit operation is computed as the* semantic distance.

We overlook the effect of domain-specific taxonomy on computing importance similarity between two keywords, however, this taxonomy can be defined with the help of domain-experts [5].

Definition 2 (Internal Differential). *The* internal differential *between X_1 and X_2 is computed as: $Diff_I(X_1, X_2) = \frac{iDis(X_1, X_2)}{max(iDis(\emptyset, X_1), iDis(\emptyset, X_2))}$, where $iDis(\emptyset, X_i)$ $(i \in \{1, 2\})$ is the internal distance of building the entire X_i from empty based on the basic edit operations.*

Definition 3 (Similar Pattern). *Given X_i and X_j, we call X_j a "Similar Pattern" for X_i in ζ_j, if $Diff_I(X_i, X_j) < \theta_s$, where θ_s is a defined threshold for internal differential.*

Definition 4 (Most Similar Pattern). *Given X_i and X_j, we call X_j the "most similar pattern" for X_i in ζ_j, if X_j is one similar pattern for X_i, and $\neg\exists X_j^{'} \in \zeta_j$ that $Diff_I(X_i, X_j^{'}) < Diff_I(X_i, X_j)$, and we use $M_{X_i \dashv \zeta_j}$ to represent the most similar pattern for X_i in ζ_j.*

Definition 5 (Internal Unchanged Pattern). *Given X_i and its most similar pattern $M_{X_i \dashv \zeta_j}$, we call X_i "internal unchanged" pattern iff $Diff_I(X_i, M_{X_i \dashv \zeta_j}) < \theta_u$, where θ_u is the unchanged threshold. $M_{X_i \dashv \zeta_j}$ is called j^{th} "version" of X_i, and X_i is called the i^{th} "version" of $M_{X_i \dashv \zeta_j}$.*

Definition 6 (Emerged Pattern). *We call X_i "emerged pattern" if no version of X_i is found in the pattern sets from ζ_0 to ζ_{i-1}.*

Mining emerging patterns was discussed in [3,8], in which a pattern is considered as "emerging" if its support is over a threshold, while not considered as its conceptual and structure. From the definitions above, it is drawn that an "internal unchanged" pattern is discovered by being compared with patterns in its posterior time spans, while an "emerged" pattern is concluded based on the patterns in its prior time spans. Given two pattern sets ζ_i and ζ_j $(i < j)$, detecting internal unchanged patterns in ζ_i is based on locating most similar patterns in ζ_j for the patterns in ζ_i. All the detected internal unchanged patterns from ζ_i form a subset of ζ_i, and we use $\zeta_i \triangleright \zeta_j$ to denote this subset and $M_{\zeta_i \triangleright \zeta_j}$ to represent the subset of ζ_j which is composed by their corresponding versions in

ζ_j. $\zeta_i \triangleright \zeta_j$ and $M_{\zeta_i \triangleright \zeta_j}$ give the internally stable elements from ζ_i to ζ_j. On the other hand, $\zeta_i - \zeta_i \triangleright \zeta_j$ and $\zeta_j - M_{\zeta_i \triangleright \zeta_j}$ are the internal un-relatedness between ζ_i and ζ_j. Based on our definitions, $\zeta_i - \zeta_i \triangleright \zeta_j$ is composed by all the "**emerged**" patterns in ζ_j, while $\zeta_i - \zeta_i \triangleright \zeta_j$ are "**perished patterns**" from ζ_i to ζ_j.

2.2 Measuring External Change

The form size is the basic symbol showing the information quantity. Form size of pattern X, named as $|X|$, is the happening number of pages in X, including requests that involved revisits. The external differential between X_i and its j^{th} version is computed by: $Diff_E(X_i, M_{X_i \dashv \zeta_j}) = \frac{|X_i| - |M_{X_i \dashv \zeta_j}|}{max(|X_i|, |M_{X_i \dashv \zeta_j}|)}$. From X_i to $M_{X_i \dashv \zeta_j}$, there are two possible variations of external feature:

- the pattern expands: if $Diff_E(X_i, M_{X_i \dashv \zeta_j}) > 0$.
- the pattern shrinks: if $Diff_E(X_i, M_{X_i \dashv \zeta_j}) < 0$.

2.3 Measuring Local Popularity Change

The local change concerns the popularity difference between two time spans, while the global is for that over the whole history. The popularity support Sup for X_i has two forms: *support number* and *support ratio*. The local popularity change from X_i to its j^{th} version $M_{X_i \dashv \zeta_j}$ is computed by: $Diff_L(X_i, M_{X_i \dashv \zeta_j}) = \frac{Sup(M_{X_i \dashv \zeta_j}) - Sup(X_i)}{Sup(X_i)}$. Given a pre-defined positive threshold θ_l for local popularity change, a pattern suffers local popularity change in two directions:

- the pattern floats: $Diff_L(X_i, M_{X_i \dashv \zeta_j}) > \theta_l$.
- the pattern sinks: $Diff_L(X_i, M_{X_i \dashv \zeta_j}) < -\theta_l$.

2.4 Measuring Global Popularity Change

Based on local popularity change, we model the *life cycle* of a pattern to gain the insights on the evolution of the population against the entire time spans.

Definition 7 (Degree of Floats). *For an emerged pattern X_i, the degree of floats along the whole time line is defined as: $DoF(X_i, \theta_l) = \frac{\sum_{j=i+1}^{n} d_j}{n-1}$, where*

$$d_j = \begin{cases} 1, if Diff_L(X_i, M_{X_i \dashv \zeta_j}) > \theta_l \\ 0, if Diff_L(X_i, M_{X_i \dashv \zeta_j}) \leq \theta_l. \end{cases}$$

Definition 8 (Degree of Sinks). *For an emerged pattern X_i, the degree of sinks along the whole time line is defined as: $DoS(X_i, \theta_l) = \frac{\sum_{j=i+1}^{n} d_j}{n-1}$, where*

$$d_j = \begin{cases} 1, if Diff_L(X_i, M_{X_i \dashv \zeta_j}) < -\theta_l \\ 0, if Diff_L(X_i, M_{X_i \dashv \zeta_j}) \geq -\theta_l. \end{cases}$$

Degree of floats and sinks for a pattern reflect the changes in two directions over the history. So the global popularity change for a pattern is measured by the pair of its DoF and Dos under threshold θ_l: $Diff_G(X_i, \theta_l) = (DoF(X_i, \theta_l), DoS(X_i, \theta_l))$.

2.5 Algorithm for X-Tracking Changes

Given an emerged pattern X_i, the pseudo-code algorithm for x-tracking the changes of X_i in the rest of time spans is shown in algorithm 1.

Before tracking the changes of a pattern, the time span in which it was firstly emerged has to be detected. The changes of a pattern is tracked against the whole history, and a pattern needs to compute its internal differentials with all the patterns before finding its most similar pattern from another time span. Scanning the patterns in a posteriori time span for a priori pattern distinguishes the newly formed patterns in a posteriori time span. On the other hand, the final formation of the subset E_i of newly emerged patterns at t_i ($E_i \subseteq \zeta_i$) is purified and decided by all the sets of newly emerged patterns before t_i:

$$E_i = \begin{cases} \zeta_1, & i = 1 \\ F(E_1, ..., E_{i-1}), & i > 1. \end{cases}$$

The accumulation of scanning reduces the expense on tracking the changes for the patterns firstly emerged in a posteriori time span. The following algorithm gives the formation of the subset of newly emerged patterns at n time spans. $M_{E_j \triangleright E_i}$ is the subset of E_i, in which each element is the version for one pattern in E_j. Based on the definition on locating most similar pattern, it can not guarantee that every element in E_j could find its version in E_i.

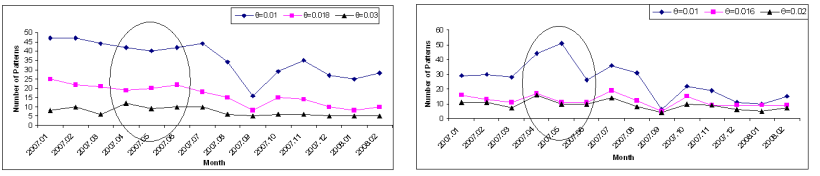
3 Experiment Results

We have experimented our x-tracking method on HPI (www.hpi.uni-potsdam.de) web site, and the usage data were taken from 2007.1 to 2008.2, from which we filtered 165535 valid usage sessions made by 79216 users. These valid usage sessions are monthly split into 14 sub sets from t_0 to t_{13}, and we mined three kinds of navigation patterns for each sub set: frequent page sets, frequent navigation paths and frequent tree navigation structures.

Figure 2 shows the number of patterns in different time spans. The set of frequent navigation paths, in 2007.05, compared with other time spans, has the

Algorithm 1. Tracking the Changes for a newly Emerged Pattern

Given X_i firstly emerged at t_i and n pattern sets $< \zeta_1, \zeta_2, \ldots, \zeta_n >$,
$DoF(X_i, \theta_l) = DoS(X_i, \theta_l) = 0$
for $j = i + 1$ to n **do**
  locate its j^{th} version $M_{X_i \dashv \zeta_j}$ for X_i in ζ_j
  if $M_{X_i \dashv \zeta_j} \neq$ NULL **then**
    compute $Diff_I(X_i, M_{X_i \dashv \zeta_j})$, $Diff_E(X_i, M_{X_i \dashv \zeta_j})$ and $Diff_L(X_i, M_{X_i \dashv \zeta_j})$
    if $Diff_L(X_i, M_{X_i \dashv \zeta_j}) > \theta_l$ **then**
      $DoF(X_i, \theta_l) = DoF(X_i, \theta_l) + 1/n$
    else if $Diff_L(X_i, M_{X_i \dashv \zeta_j}) < -\theta_l$ **then**
      $DoS(X_i, \theta_l) = DoS(X_i, \theta_l) + 1/n$
    end if
  end if
end for



(a) Frequent Page Sets (b) Frequent Navigation Paths

Fig. 2. Distribution of the number of different kinds Patterns

biggest number of paths under threshold $\theta = 0.01$, but has a relative small number under $\theta = 0.02$. On the other hand, we can discover the same results in figure 2 (a). The numbers of navigation patterns are decided by the thresholds and the types of patterns, however, the variations or differences of these numbers are manipulated by the compositions of individual behaviors and such differences are multi-facets. The variations of numbers of patterns could not show the difference of the compositions of patterns.

The methods we used to extract semantics from web pages is refereed from *Lucene* [10], but adjusted to our applications. We selected 2851 web pages from our target site, and *Lucene* generated 43126 searchable terms, from which 20654 terms are removed as n-grams, 2640 numeric terms are out filtered. After removing 6392 non-significant words including pronouns, articles, adjectives, adverbs and prepositions, 13440 meaningful term candidates are extracted and the average number of extracted terms for each page is 120.

The changes an emerged pattern suffered are computed by comparing with its versions in posterior time spans, two thresholds, δ_s and δ_u, are defined. For example, under $\delta_u = 0.3$ and $\delta_s = 0.5$, the distribution of emerged patterns at different time spans is given in figure 3. In this figure, the gap between the "num_raw" and "num_unchanged" presents the number of emerged patterns at different time spans. The high percentage of emerged patterns compared with unchanged patterns reminds the big internal change of usage patterns, which reflects the semantical drifting of usage interest. Shown from figure 3, all patterns at $t0$ are newly emerged, and $t8$ has a highest percentage of emerged patterns. Pattern sets with the remarkable internal change could be selected, and the most unimportant patterns are removed. This is one significance of detecting internal changes among navigation patterns. However, only concerning the occurrence of

Algorithm 2. Formation of newly Emerged Patterns at i_{th} Time Span

1: Given n pattern sets $< \zeta_1, \zeta_2, ..., \zeta_n >$
2: Initialize $E_1 = \zeta_1$, $E_2 = \zeta_2$, ..., $E_n = \zeta_n$
3: **for** $i = 2$ to n **do**
4: **for** $j = 1$ to $i - 1$ **do**
5: $E_i = E_i - M_{E_j \triangleright E_i}$
6: **end for**
7: **end for**

newly emerged patterns could not reveal their evolution in the posterior time spans and the relationship among the unchanged patterns in different time spans. We track the life cycle of one emerged pattern by computing its external change, local and global population change with its unchanged pattern in posterior time spans.

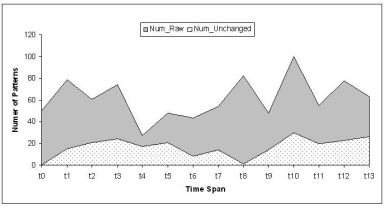


Fig. 3. EP v.s. UP under $\Theta = 0.01$, $\delta_u = 0.3$ and $\delta_s = 0.5$

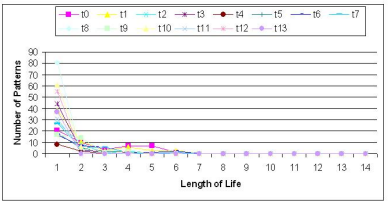


Fig. 4. Life length for emerged patterns at different time spans

The life time of a pattern X_i is the number of time spans, in which X_i has its versions. Shown in figure 4, a pattern emerged in priori time span has a higher possibility to have a long life time than that emerged in posterior time span, this is usually related to the periodic updates of the web site. However, a pattern with only 1 life length indicates a temporal variation of usage interest and such variation could be unexpected interesting, noise from robots or due to special events. Though $t8$ has the highest percentage of newly emerged patterns due to the big updates on the site, these patterns did not survive after $t8$. In practice, the manager possibly has interest on the changing history of a runtime pattern, especially the changes on its external form and support population suffered in its life cycle. This can be drawn by querying the DB table about the relations between every emerged pattern and its versions with their external and local popularity changes.

Here we only discuss the performance of x-tracking algorithm. The sizes of patterns and pattern sets have great impact on the effectiveness and efficiency, and the internal differential computation between two pages relies on the dimension of the extracted terms. The biggest part of time consuming is semantic extraction and data cleaning before mining and x-tracking, and depends highly on manually terms and data purifying. Two thresholds θ_s and θ_u decide the number of emerged patterns from every time spans and further affect the times on tracking the versions for internal unchanged patterns. Figure 5 collects the time consuming on x-tracking changes among patterns in two different scenarios. In scenario 1, there are altogether 585 patterns in 14 time spans, and the average size of pattern is 4.4; while 960 patterns in scenario 2 with an average size 5.8 for a pattern. From this figure, we see that x-tracking algorithm is effective in a reasonable application, though the time consuming in theory is $O(n^3)$ with the increasing of pattern sets and their lengths.

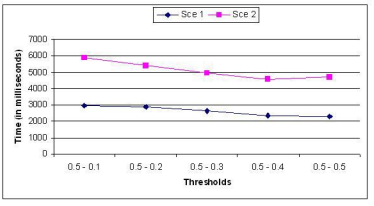


Fig. 5. Time results on x-tracking in two scenarios

4 Conclusion and Future Work

In this paper, we propose an x-tracking method to detect the changes of web navigation patterns from four aspects. The x-tracking method we propose is to detect the newly emerged patterns based on "internal" feature integrating semantic distance with structural distance. We are working on two directions: one is the asynchrony between the changes of site semantics and navigation patterns; the other is the presentation on the multi-facet changes.

References

1. Baron, S., Spiliopoulou, M.: Monitoring the Evolution of Web Usage Patterns. In: Berendt, B., Hotho, A., Mladenič, D., van Someren, M., Spiliopoulou, M., Stumme, G. (eds.) EWMF 2003. LNCS, vol. 3209, pp. 181–200. Springer, Heidelberg (2004)
2. Chawathe, S.S.: Comparing Hierarchical Data in Extended Memory. In: Proc. of the 25th VLDB (1999)
3. Dong, G., Li, J.: Efficient mining of emerging patterns: Discovering trends and differences. In: Proc. of SIGKDD 1999 (1999)
4. Nasraoui, O., Cardona, C., Rojas, C.: Mining Evolving Web Clickstreams with Explicit Retrieval Similarity Measures. In: Proc. of WWW 2004 (2004)
5. Eirinaki, M., Vazirgiannis, M., Varlamis, I.: SEWeP: Using Site Semantics and a Taxonomy to Enhance the Web Personalization Process. In: Proc. of SIGKDD 2003 (2003)
6. Yan, X., Cheng, H., Han, J., Xin, D.: Summarizing Itemset Patterns: A Profile-Based Approach. In: Proc. of SIGKDD 2005 (2005)
7. Wagner, R.A., Fischer, M.J.: The String-to-String correction problem. Journal of the ACM (1974)
8. Zhao, Q., Bhowmick, S.S.: Mining History of Changes to Web Access Patterns. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS, vol. 3202, pp. 521–523. Springer, Heidelberg (2004)
9. Zhao, Q., Bhowmick, S.S., Mohania, M., Kambayash, Y.: Discovering Frequently Changing Structures from Historical Structural Deltas of Unordered XML. In: Proc. of CIKM 2004 (2004)
10. `http://lucene.apache.org/`

Tree-Based Method for Classifying Websites Using Extended Hidden Markov Models

Majid Yazdani, Milad Eftekhar, and Hassan Abolhassani

Web Intelligence Laboratory, Computer Engineering Department,
Sharif University of Technology, Tehran, Iran
{yazdani,eftekhar}@ce.sharif.edu,
abolhassani@sharif.edu

Abstract. One important problem proposed recently in the field of web mining is website classification problem. The complexity together with the necessity to have accurate and fast algorithms yield to many attempts in this field, but there is a long way to solve these problems efficiently, yet. The importance of the problem encouraged us to work on a new approach as a solution. We use the content of web pages together with the link structure between them to improve the accuracy of results. In this work we use Naïve-bayes models for each predefined webpage class and an extended version of Hidden Markov Model is used as website class models. A few sample websites are adopted as seeds to calculate models' parameters. For classifying the websites we represent them with tree structures and we modify the Viterbi algorithm to evaluate the probability of generating these tree structures by every website model. Because of the large amount of pages in a website, we use a sampling technique that not only reduces the running time of the algorithm but also improves the accuracy of the classification process. At the end of this paper, we provide some experimental results which show the performance of our algorithm compared to the previous ones.

Keywords: Website classification, Extended Hidden Markov Model, Extended Viterbi algorithm, Naïve-Bayes approach, Class models.

1 Introduction

With the dramatically increasing number of sites and the huge size of the web which is in the order of hundreds of terabytes [5] and also with the wide variety of user groups with their different interests, probing for sites of specific interests in order to solve users' problems is really difficult. On the other hand, almost predominant section of the information which exists in the web is not practical for many users and this portion might interfere the results which are retrieved by users' queries. It is apparent that searching in the tiny relevant portion can provide us better information and lead us to more interesting sites and places on a specific topic.

At this time, there are a few directory services like DMOZ [3] and Yahoo [11] which provide us by useful information in several topics. However, as they

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 780–787, 2009.
© Springer-Verlag Berlin Heidelberg 2009

are constructed, managed and updated manually, most of the time they have incomplete old information. Not only webpages changes fast, but also linkage information and access records are updated day by day. These quick changes together with the extensive amount of information in the web necessitate automated subject-specific website classification.

This paper proposes an effective new method for website classification. Here, we use the content of pages together with the link structure of them to obtain more accuracy and better results in classification. Also among different models for representing websites, we choose tree structure for its efficiency. Tree model is useful because it displays the link structure of a given website clearly.

Before we begin to talk about ways of website classification, it is better to explain the problem more formally. Given a set of site classes C and a new website S consisting of a set of pages P, the task of website classification is to determine the element of C which best categorizes the site S based on a set of examples of preclassified websites. In other words, the task is to find a class C that website S is more likely to be its member.

The remaining of this paper is organized as follows. Related works are discussed in Section 2. Section 3 describes the models which we use for representing websites and website classes. In Section 4, we explain our method for classifying websites together with the extended version of Viterbi algorithm. Section 5 is about learning and sampling techniques. A performance study is reported in Section 6. Finally, Section 7 concludes the paper and discusses future works. A completer version of this paper is available through authors' homepages.

2 Related Works

Text classification has been an active area of research for many years. A significant number of these methods have been applied to classification of web pages but there was no special attention to hyperlinks. Apparently, a collection of web pages with a specific hyperlink structure conveys more information than a collection of text documents. A robust statistical model and a relaxation labeling technique are presented in [2] to use the class label and text of neighboring in addition to text of the page for hypertext and web page categorization. Categorization by context is proposed in [1], which instead of depending on a document alone, extracts useful information for classifying the document from the context of the page in which the hyperlink to the document exists. Empirical evidence is provided in [9] which shows that good web-page summaries generated by human editors can indeed improve the performance of web-page classification algorithms.

On the other hand, website classification has not been researched widely; the basic approach is superpage-based system that is proposed in [8]. Pierre discussed several issues related to automated text classification of Web sites, and described a superpage-based system for automatically classifying Web sites into industrial categories. In this work, we just generate a single feature vector counting the frequency of terms over all HTML-pages of the whole site, i.e. we represent a

web site as a single "superpage". Two new more natural and more expressive representation of web sites has been introduced in [6] in which, every webpage is assigned a topic out of a predefined set of topics. In the first one, Feature Vector of Topic Frequencies, each considered topic defines a dimension of the feature space. For each topic, the feature values represent the number of pages within the site having that particular topic. In the second one, Website Tree, the website is represented with a labeled tree and the k-order Markov tree classifier is employed for site categorization. The main difference between our work and this work is that in the latter, the topic (class) of each page is independently identified with a classifier and without considering other pages in the website tree, and then the topic of pages tree is used to compute the website class, but this independent topic identification will lower the accuracy of responses. Whereas in our work we calculate the website class without explicitly assigning a topic to each page and the topics of pages will be hidden to us. In [7] a website is represented with a set of feature vectors of terms. By choosing this representation, effort spent on deriving topic frequency vectors will be avoided. kNN-classification is employed to classify a given website according to training database of websites with known classes. In [10] the website structure is represented by a two layered tree: a DOM tree for each page and a page tree for the website. In [4] a new approach is presented for classifying multipage documents by naïve bayes HMM. However, to the best of our knowledge, there is no work on extending HMM for classifying data represented as tree.

3 Modeling Websites and Classes

Before modeling websites and website classes, defining some terminology is necessary.

Definition 1. *(The set of webSite class Label : SL) SL is the set of labels which can be assigned to websites as class labels, in other words, members of SL are category domains.*

Definition 2. *(The set of webPage class Labels : PL) For each label sl in SL, there is a set of known labels which can be assigned to individual pages of a website in that specified category. This labels can be seen as pages' topics.*

There are many choices to use as page class models like Naïve-Bayesian approach and Hidden Markov models. We prefer the former due to its simplicity. In this paper Naïve-Bayes model is adopted for modeling webpage classes. For modeling website classes, we extend Hidden Markov Model in order to satisfy the criteria of content and link structure within pages.

Definition 3. *(webSite Class Model : SCM) For each category domain $sl \in SL$, the model of sl is a directed graph $G(V, E)$ in which V,G's vertex set, is a set of webpage class models and for every two states $v_i, v_j \in V$ there is two directed edges $\{(i.j), (j, i)\} \in E$ that show the probability of moving from one to another. Also there is a loop for every state which shows the probability of transition between two pages of that class.*

In this model, the state-transition probability matrix A is an $n * n$ matrix in which n is the cardinality of SL and $a_{ij} = P(c_{t+1} = j|c_t = i),\ 0 \leq i, j \leq n$, which shows the probability of transition from state i to state j. Also the emission probability e is: $e_j(p) = P(p_t = p|c_t = j)$. $e_j(p)$ represents the probability of generating page p by webpage class model j.

As a result, we have a hybrid of Naïve-Bayes HMM model for all of websites classes. As we mentioned before, it was possible to model website classes by a hierarchial HMM, if HMM was used instead of Naïve-Bayes model for describing webpage classes. It is important to note that the input of website class models are websites which are modeled by trees. We describe website models in a more formal way, as follows.

Definition 4. *(webSite Model : SM) For each website ws, the ws's model is a page tree $T_{ws} = (V, E)$ in which V is a subset of ws's pages. The root of T_{ws} is the homepage of the website and there is a directed edge between two vertices $v_i, v_j \in V$ if and only if the corresponding page p_j is a child of p_i. We crawl the website by Breath-First-Search technique and ignore the links to previously visited pages in the time of model construction.*

4 Website Classification

In previous sections, we described a model for each website class. We assume SCM_i is the model of website class C_i. Also we consider that we want to classify website ws. To find the category of ws, we calculate $P(SCM_i|ws)$ for $1 \leq i \leq n$. By Considering Bayes rule, we can determine the class of ws as

$$C_{map} = argmax_i P(SCM_i|ws) = argmax_i P(SCM_i)P(ws|SCM_i)$$

$P(ws)$ is constant for all classes and we neglect $P(SCM_i)$. It can be considered later or even it can be equal for all classes if we use a fair distribution of websites over different classes. Therefore $C_{map} = argmax_i P(ws|SCM_i)$.

To find the probability of $P(ws|SCM_i)$ and also to solve the webpage classification problem, we should extend the Viterbi algorithm for our models.

4.1 Extended Viterbi Algorithm

Using dynamic programming techniques and therefore Viterbi algorithm ideas seems reasonable for solving these probabilities. However as we mentioned before, the inputs of our models are trees of pages, while Viterbi algorithm is provided to calculate the maximum probability of generating sequences. Therefore, we should modify the traditional Viterbi algorithm. Before introducing the new approach, it is necessary to present theorem 1 which is discussed in [10].

Theorem 1. *The probability of each node is only dependent to its parent and children. More formally, for every node n with p as its parent and $q_1, ...q_n$ as its children, if PL represents the webpage label set then*

$$P(PL_n|\{PL_k|k \in V\}) = P(PL_n|\{PL_p, PL_{q_1}, ..., PL_{q_n}\}) \quad (1)$$

Here, we propose a novel method in which the probability of a node and its siblings is computed simultaneously to avoid any possible inconsistency. Possible inconsistencies together with an illustrating example is provided in the completer version of the paper. Thus, in the nth level of the tree, we calculate the probability of the children of an $n-1$th level's node by equation 2.

Algorithm 1. *(Extended Viterbi Algorithm) For Classifying an indicated website ws which is modeled by a tree structure T_{ws} against n different classes $C_1, \ldots, C_n$ which are modeled by $SCM_1, \ldots, SCM_n$, if p is a node in level $n-1$ of T_{ws} and it has children $q_1, \ldots, q_n$ then*

$$P[(q_1, \ldots, q_n) \leftarrow (pl_{s_1}, \ldots, pls_n)] = \prod_{i=1}^{n} e_{pl_{s_i}}(q_i) * \max_{j=1}^{m}(P(p = pl_j) * \prod_{i=1}^{n} A_{jpl_{s_i}}) \tag{2}$$

where $pl_{s_i} \in PL$. In equation 2, $P[(q_1, \ldots, q_n) \leftarrow (pl_{s_1}, \ldots, pls_n)]$ is the maximum probability of generating the nodes of upper levels by the model as well as assigning pl_{s_1} to page q_1, pl_{s_2} to page q_2, $\cdots$, and pl_{s_n} to page q_n. To calculate equation 2 for lower levels, we should have the probability of each page individually. Therefore we calculate these probabilities by equation 3.

$$P(q_i = pl_j) = \sum P[(q_1, \ldots, q_n) \leftarrow (pl_{s_1}, \ldots, pl_{s_n})] \tag{3}$$

where $pl_{s_i} = pl_j$ and $\forall k \neq i : pl_{s_k} \in PL$. The probability of generating the tree model of the website ws from the model SCM_i when q represents a leaf of the tree is:

$$P(T_{ws}|SCM_i) = \prod_{q} max_{j=1}^{|PL|} P(q = pl_j) \tag{4}$$

Here we want to compute the complexity of our algorithm. We should compute $p(q_i = pl_j)$ for every node in each level. Suppose we have branching factor of maximum b in this tree, so for each set of siblings the computation of $P[(q_1, ..., q_b) \leftarrow (pl_{s_1}, ...pl_{s_b})]$ for every possible order of $< s_1, \ldots, s_b >$ has $O(n^{b+1})$ time complexity, in which n is the number of page labels. For each node in this set of siblings we can calculate $p(q_i = pl_j)$ from $P[(q_1, ..., q_b) \leftarrow (pl_{s_1}, ...pl_{s_b})]$ probabilities in $O(n^{b-1})$ time. So we compute $p(q_i = pl_j)$ for each node in $O(n^{b+1})$. Whereas we should compute this probability for every node in the tree, the total complexity will be $O(n^{b+1}b^{L-2})$, where L is the number of tree's levels.

5 Learning and Sampling

As mentioned above, first, we determine website class and webpage class labels. Then a set of sample websites for learning phase are assigned by an expert or from sites like DMOZ as seeds for constructing class models. One of the fundamental steps in this phase is assigning a label to each web page. There are two types of pages: One that has predetermined labels (i.e. form a constant part in their URL) and the other which has no specified label. We have to assign labels to

the second type. We assign labels to about %2 of pages in the Training set manually. The remaining %98 of the pages will be labeled by Naive Bayes based upon this 2 percent. To construct website class models, we have to compute the state-transition matrix A and emission probability e. A can be easily computed as follows.

$$a_{ij} = \frac{N(c_i, c_j) + 1}{\sum_{k=1}^{n} N(c_i, c_k) + n}$$

in which $N(c_i, c_j)$ is the number of times that a page of type c_i links to a page of type c_j in the training set. As we use naïve-bayes, we can also easily compute $e_{c_j}(p)$. Therefore, for each page model c_j and word w_l

$$P(w_l|c_j) = \frac{N(w_l, c_j) + 1}{\sum_{i=1}^{|V|} N(w_i, c_j) + |V|}$$

$N(w_l, c_j)$ is the number of occurrence of word w_l in webpages of class c_j and V represents the set of selected keywords. Therefore, if p is formed from $w_1 \dots w_n$ then $e_{c_j}(p) = P(w_1|c_j) \dots P(w_n|c_j)$.

There are two general reasons for our motivation to use page pruning algorithm. First, downloading web pages in contrast with operations that take place in memory is very expensive and time consuming. In a typical website there are too many pages that cannot convey useful information for website classification, if we can prune these pages, website classification performance improves significantly. The second reason that leads us to use pruning algorithm is that in a typical website, there are pages that affect classification in an undesirable direction, so pruning these unrelated or unspecific pages can improve our accuracy in addition to performance.

We use the pruning measures used in [6] for its efficiency and we modify the pruning algorithm for our method. To compute measures for a partial tree which we have downloaded up to now, we should be able to compute membership of this partial tree website for each website class. Our model is suitable for this computation because we compute the probability of each partial tree incrementally and when we add new level to previous partial tree we just compute the $P[(q_1, ..., q_b) \leftarrow (pl_{s_1}, ...pl_{s_b})]$ probabilities for every possible set of $\{s_1, \dots, s_b\}$ for all the new siblings by using previous probabilities and according to our algorithm. Then by using these probabilities we calculate $p(q_i = pl_j)$ for each node in the new level. For each node q, we define $P(q|sl_i) = \max_{pl_j \in PL} p(q = pl_j)$ where $sl_i \in SL$.

For each node q of partial website tree t: $weight(q) = \sigma^2_{sl_i \in SL}(P(q|sl_i)^{\frac{1}{depth(q)}})$. By adding a new node q to a partial tree t we obtain a new partial tree t_2. We stop descending the tree at q if and only if $weight(q) < weight(parent(q)) * \frac{depth(q)}{\omega}$. By means of this pruning procedure and saving probabilities, we perform classification in the same time we use pruning algorithm, and then we can determine the most similar class of the given website. We examine different ω to find appropriate one for our data set. Choosing proper ω can help us to achieve even higher accuracy than complete website download.

6 Experimental Results

In this section we demonstrate the results of some experimental evaluation on our approach and compare it with other existing algorithms, mainly extracted from [6]. We use these methods to classify scientific websites into ten following classes: Agriculture, Astronomy, Biology, Chemistry, Computer Science, Earth Sciences, Environment, Math, Physics, Other. We use DMOZ[3] directory to obtain websites for the first 9 classes. We downloaded 25 website for each class and a total of 86842 web pages. At this point we downloaded a website almost completely (Limited number of pages at most 400). For the "other" class we randomly chose 50 websites from Yahoo! Directory classes that were not in the first nine classes which had 18658 web pages. We downloaded all websites to local computer and saved them locally. To use our algorithm first we should prepare our initial data seed and then we build a model for each class and compute its parameter as stated in the learning phase. To classify the pages of a website class, we labeled about %2 of them in each web site class manually then The remaining %98 of the pages were labeled by Naive Bayes based upon this labeled pages. At the end of this process we have a naïve-bayes model for each page class of each website category. By means of these naïve-base models we classified web pages for other methods. For testing our methods we randomly downloaded 15 new website almost complete (limiting to 400 pages)for each class.

We compare our algorithm to 4 other methods: 0-order Markov tree, C4.5, Naïve-bayes, classification of superpage. In superpage, classifying a web site is to extend the methods used for page classification to our definition of web sites. We just generate a single feature vector counting the frequency of terms over all HTML-pages of the whole site, i.e. we represent a web site as a single "super-page". For C4.5 and naïve-bayes, first we build Feature vector of topic frequencies and then apply naïve-bayes and C4.5 algorithms on them. For the 0-order Markov tree we used the method described in [6]. You can find the accuracy of tested methods on testing dataset in table 1.

Table 1. Comparison of accuracy between different classification methods

Classifier	Accuracy
Super Page	57.7
Naïve-Bayes	71.8
C4.5	78.5
0-Order Markov Tree	81.4
Our algorithm	85.9

As it can be seen the accuracy of our method is better than other methods. It is more accurate compared to 0-order Markov tree because page classes are hidden here and we calculate probability of the whole website that is generated from a model.

At last we examine the impact of different ω values on the sampling algorithm in our training set. With an appropriate ω, the accuracy increases in comparison to complete website. To find appropriate ω, we increased ω gradually and when the overall accuracy stopped to increase, we choose ω. In our data set the appropriate ω was 6 , but this can change in respect to data set.

7 Conclusions and Future Works

In the growing world of web, taking advantage of different methods to classify websites seems to be very necessary. Website classification algorithms for discovery of interesting information leads many users to retrieve their desirable data more accurately and more quickly. This paper proposes a novel method for solving this problem. With extending Hidden Markov Model, we described models for website classes and looked for the most similar class for any website. Experimental Results show the efficiency of this new method for classification.

In the ongoing work, we are seeking for new methods to improve the efficiency and accuracy of our website classification method. Demonstrating websites with stronger models like website graphs can bring us more accuracy.

References

1. Attardi, G., Gullí, A., Sebastiani, F.: Automatic Web page categorization by link and context analysis. In: Proc. of THAI-99, European Symposium on Telematics, Hypermedia and Artificial Intelligence, Varese, IT, pp. 105–119 (1999)
2. Chakrabarti, S., Dom, B.E., Indyk, P.: Enhanced hypertext categorization using hyperlinks. In: Proc. ACM SIGMOD, Seattle, US, pp. 307–318 (1998)
3. DMOZ. open directory project
4. Frasconi, P., Soda, G., Vullo, A.: Text categorization for multi-page documents: A hybrid naïve bayes hmm approach. In: Proc. 1st ACM-IEEE Joint Conference on Digital Libraries (2001)
5. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publisher, San Francisco (2006)
6. Ester, M., Kriegel, H., Schubert, M.: Web site mining: A new way to spot competitors, customers and suppliers in the world wide web. In: Proc. of SIGKDD 2002, Edmonton, Alberta, Canada, pp. 249–258 (2002)
7. Kriegel, H.P., Schubert, M.: Classification of Websites as Sets of Feature Vectors. In: Proc. IASTED DBA (2004)
8. Pierre, J.M.: On the automated classification of web sites. In: Linköping Electronic Article in Computer and Information Science, Sweden 6(001) (2001)
9. Shen, D., Chen, Z., Zeng, H.-J., Zhang, B., Yang, Q., Ma, W.-Y., Lu, Y.: Web-page classification through summarization. In: Proc. of 27th Annual International ACM SIGIR Conference (2004)
10. Tian, Y.-H., Huang, T.-J., Gao, W.: Two-phase web site classification based on hidden markov tree models. Web Intelli. and Agent Sys. 2(4), 249–264 (2004)
11. Yahoo! Directory service

Emotion Recognition of Pop Music Based on Maximum Entropy with Priors

Hui He, Bo Chen, and Jun Guo

School of Information and Communication Engineering,
Beijing University of Posts and Telecommunications
100876 Beijing P.R. China
{hh1012,chb615}@gmail.com, guojun@bupt.edu.cn

Abstract. Efficient and intelligent music retrieval has become a very important topic nowadays. Analysis of lyrics must be a complement of acoustic methods for music retrieval. One basic aspect of music retrieval is music emotion recognition by learning from lyrics. This problem is different from traditional text classification in that more linguistic or semantic information is required for better emotion analysis. Thereby, we focus on how to extract meaningful features and how to modeling them for music emotion recognition. First, we investigate the lyrics corpus based on Zipf's Law using word as a unit, and results roughly obey Zipf's Law. Then, we study three kinds of preprocessing methods and a series of language grams under the well-known n-gram language model framework to extract more semantic features. At last, we employ three supervised learning methods, Naïve Bayes, maximum entropy classification, and support vector machine, to examine the classification performance. Besides that, we also improve ME with Gaussian and Laplace priors to model features for music emotion recognition. Experiment al results show that feature extraction methods improved music emotion recognition accuracy. ME with priors obtained the best.

Keywords: Music emotion, Maximum Entropy, Zipf's Law, Gaussian prior, Laplace prior.

1 Introduction

The rapid growth of the Internet and the advancements of Internet technologies have made it possible for music listeners to have access to a large amount of online music data, including music sound signals, lyrics, biographies, and so on. This raises the question of whether computer programs can enrich the experience of music listeners by enabling the listeners to have access to such a large volume of online music data.

Traditionally musical information has been retrieved and/or classified based on standard reference information, such as the name of the composer and the title of the work etc. But these are far from satisfactory. Huron pointed out that since the preeminent functions of music are social and psychological, the most useful characterization would be based on four types of information: the style, emotion, genre, and similarity [1].

The emotional component of music has attracted interest of the Music Information Retrieval (MIR) community, and experiments have been conducted to classify music by mood [2]. In this paper, we choose to learn from lyrics for two main reasons. First, it is easy to get lyrics from Internet. Second, it is much easier to process lyrics than

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 788–795, 2009.
© Springer-Verlag Berlin Heidelberg 2009

audio signals since the preprocessing of audio signals is more complicated and time-consuming. Thus it is more suitable for computing in mobile or MP3, MP4, etc.

However, previous work [3] suggests that lyrics are difficult for natural language processing. First, unlike other text data such as news, emotion words in lyrics are less occurring. Instead, a large portion will be devoted to the background. This makes the term absolute frequency information not so useful, or even misleading. Second, because lyrics are free-style, approaches based on word positions also face difficulty. Third, lyrics downloaded from Internet may contain spelling mistakes, because most of lyrics are added by listeners. Therefore, in this paper, we adopt the popular n-gram language models and part of speech (POS) to mine kinds of language features. And then we employ three supervised learning methods and two improved ME models to examine the classification performance.

2 Related Work

2.1 Learning from Audio Signals

Relations between musical sounds and their impact on the emotion of the listeners have been studied for decades. Experiments taken by Hevner [4] confirmed a hypothesis that music inherently carries emotional meaning. Li [2] combined Daubechies Wavelet Coefficient Histograms (DWCH) with timbral features to extract music emotion features from audio.

2.2 Learning from Lyrics

Most previous work learning from lyrics is on stylometric analysis. Li [2] studied the problem of identifying groups of artists by combining acoustic-based features and lyrics-based features and using a semi-supervised classification algorithm. Zheng [5] studied Chinese song lyrics corpus based on Zipf's Law and discussed how to use the time annotation for further analysis. Wei [6] proposed a scheme for content based keyword generation of song lyrics.

Work of music emotion classification via lyrics is rare. Lyrics is a special kind of text, so learning from lyrics can be taken as sentiment classification via texts. Similar work has been reported in [7] and improved work in [8]. In these two papers, the language feature extraction methods and machine learning techniques they used are of great help of our work.

3 Language Feature Extraction

In normal documents, there are always enough items to form term-vectors, while in lyrics, fewer items are there, especially for emotion words. Thus, we need to apply language model to capture more language features to express these lyrics.

3.1 Language Features

3.1.1 Basic Unigram

The standard word-vector or so called bag-of-words is the basic and mostly used representation of document. For example, in a text snippet T consist of n words can

be represented as a word set $\{w_1, w_2, \cdots, w_n\}$. While being put into the n-gram framework, it can be viewed as a basic unigram form, which is a context free document representation model.

3.1.2 N-Gram

It's obvious that a high order n-gram model can capture some short-distance dependencies by combining n sequential individual terms into a compounded feature.

3.1.3 Part of Speech

It is found in previous work that only a few words, whose part-of-speech are adjective, verb, noun, and adverb, can present text sentiment. So, one way of extracting features is to filter lyrics only retain words mentioned above.

3.2 Feature Selection and Weighting

The original amount of these features will be too huge for an efficient document analysis system. And, a great percent of these language-grams are nothing but sequential words, which have neither linguistic structure nor semantic meaning and should be cleared out of our language model. Therefore, there are two procedures need to be perform on the original language-grams, one is feature selection, and the other is feature weighting. We mainly applied a simple feature elimination method that is to abandon those feature appearing fewer times than a threshold. For feature weighting, we try three feature weighting methods: Boolean value, absolute term frequency and term frequency-inverse document frequency (TFIDF) weighting method.

4 Language Feature Modeling

Let $\{f_1, f_2, \cdots, f_m\}$ be a predefined set of m features that can appear in a lyrics. Let $n_i(l)$ be the number of times f_i occurs in lyrics l. Then, each lyrics l is represented by the lyric vector $\vec{l} = weighting(n_1(l), n_2(l), \cdots, n_m(l))$, $weighting()$ is weighting methods mentioned in section 3.2. For each lyrics $l \in L$, L is a lyrics set. $c \in C$ denotes emotion category and C is emotion label set.

4.1 Maximum Entropy Model

Maximum entropy classification (ME, for short) [9] estimates of $P(c \mid l)$ takes the following exponential form:

$$P_{ME}(c \mid l) = \frac{1}{Z(l)} \exp\left(\sum_i \lambda_i F_{i,c}(l, c)\right) \tag{1}$$

where $Z(l)$ is a normalization function. $F_{i,c}$ is a feature/class function for feature f_i and class c, defined as follow:

$$F_{i,c}(l,c') = \begin{cases} 1, & n_i(l) > 0 \quad and \quad c' = c \\ 0, & otherwise \end{cases} \tag{2}$$

The λ_i's are feature-weight parameters; inspection of the definition of P_{ME} shows that a large λ_i means that f_i is considered a strong indicator for class c. For each indication function $F_{i,c}(l,c)$, let $\tilde{p}(l,c)$ be the empirical distribution of sample (c,l), then the empirical expectation of $F_{i,c}(l,c)$ is defined as:

$$\tilde{E}(F_{i,c}) = \sum_{l \in L, c \in C} \tilde{p}(l,c) F_{i,c}(l,c) \tag{3}$$

And the expected value of $F_{i,c}(l,c)$ estimated by the ME model as:

$$E(F_{i,c}) = \sum_{l \in L, c \in C} \tilde{p}(l) p_\Lambda(c \mid l) F_{i,c}(l,c) \tag{4}$$

wherein, $\tilde{p}(l)$ is the empirical distribution of sample l, and $p_\Lambda(c \mid l)$ is the conditional distribution of (c,l) estimated by the ME model.

ME models require constrains on $F_{i,c}(l,c)$ that

$$E(F_{i,c}) = \tilde{E}(F_{i,c}) \text{, for all } F_{i,c}(l,c) \tag{5}$$

Another important constrain is that ME models try to remain as similar to the uniform distribution as possible. With these constrains, a ME model is trained to find the most suitable parameter vector $\Lambda = \{\lambda_i\}$ that maximize the likelihood over the dataset, and the optimization object function will be:

$$\arg\max_{\Lambda} \Pr(\tilde{p} \mid p) = \arg\max_{\Lambda} \prod_{l \in L, c \in C} p_\Lambda(c \mid l)^{\tilde{p}(l,c)} = \arg\max_{\Lambda} \sum_{l \in L, c \in C} \tilde{p}(l,c) \log\left(p_\Lambda(c \mid l)\right) \tag{6}$$

4.2 Maximum Entropy Model with Priors

Like most models trained under the maximum likelihood principle, ME models also involve in the over fitting problem [10]. This could be illustrated as follows:

Finding the model that matches the empirical distribution the best can be formulated as solving $\arg\max_{p_\Lambda} \Pr(p_\Lambda \mid \tilde{p})$. By using Bayes' law, we get

$$\arg\max_{\Lambda} \Pr(p_\Lambda \mid \tilde{p}) = \arg\max_{p_\Lambda} \frac{\Pr(\tilde{p} \mid p_\Lambda) \Pr(p_\Lambda)}{\Pr(\tilde{p})} = \arg\max_{p_\Lambda} \Pr(\tilde{p} \mid p_\Lambda) \Pr(p_\Lambda) \tag{7}$$

Assuming $\Pr(p)$ is uniform, then

$$\arg\max_{p_\Lambda} \Pr(p_\Lambda \mid \tilde{p}) = \arg\max_{p_\Lambda} \Pr(\tilde{p} \mid p_\Lambda) \tag{8}$$

from the above two formulations, it's can be seen that in the maximum likelihood estimation method, the prior $\Pr(p_\Lambda)$ of each possible model $p_\Lambda(c \mid l)$ is lost.

Chen and Rosenfeld [11] have implemented a Gaussian prior with 0 mean for ME model on language modeling task, and concluded that it was consistently the best compared to previous n-gram smoothing methods. With a Gaussian prior on the parameter λ_i as $p(\lambda_i) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp(-\frac{\lambda_i^2}{2\sigma_i^2})$, the object function (6) becomes

$$\arg\max_{\Lambda} \Pr(\tilde{p} \mid p) \Pr(p) = \arg\max_{\Lambda} \sum_{l \in L, c \in C} \tilde{p}(l,c) \log\left(p_\Lambda(c \mid l)\right) - \sum_i \frac{\lambda_i^2}{2\sigma_i^2} \tag{9}$$

Such a model is aimed at maximize posteriori instead of maximum likelihood on parameter values. Gaussian priors will also lead to changes on constrains (5) as

$$E(F_{i,c}) = \tilde{E}(F_{i,c}) - \frac{\lambda_i}{\sigma_i^2}, \text{ for all } F_{i,c}(l,c) \tag{10}$$

it can be found that the Gaussian priors add some discounts on the original constrain of $F_{i,c}(l,c)$, which result in a typical smoothing approach. Kazama [12] explored a ME model with box-type inequality constraints defined by (5) as

$$A_i \geq E(F_{i,c}) - \tilde{E}(F_{i,c}) \geq -B_i, A_i, B_i > 0 \text{ for all } F_{i,c}(l,c) \tag{11}$$

In fact, the inequality ME model is just like a ME model with Laplace priors, which is similar to models with exponential priors (i.e. single-side Laplace priors), as introduced by Goodman in [13]. Assuming an exponential prior on parameter λ_i as $p(\lambda_i) = \mu_i \exp(-\mu_i \lambda_i)$, the object function of a ME model becomes

$$\arg\max_{\Lambda} \Pr(\tilde{p} \mid p) \Pr(p) = \arg\max_{\Lambda} \sum_{l \in L, c \in C} \tilde{p}(l,c) \log\left(p_\Lambda(c \mid l)\right) - \sum_i \mu_i \lambda_i \tag{12}$$

$$\begin{cases} E(F_{i,c}) = \tilde{E}(F_{i,c}) - \mu_i & \lambda_i > 0 \\ E(F_{i,c}) \geq \tilde{E}(F_{i,c}) - \mu_i & \lambda_i = 0 \end{cases} \text{ for all } F_{i,c}(l,c) \tag{13}$$

Such constrains lead to bounded absolute discounting by constants. Goodman showed that exponential priors could also lead to a simpler learning algorithm.

5 Experiments

5.1 Datasets and A Closer Look

In this paper, the music emotion recognition is based on the lyrics of Chinese pop music. This data set consists of 1,903 songs. 803 songs are labeled with emotion of

love, and 1100 songs are labeled as lovelorn songs. For comparison, we use 863 plan text classification evaluation data as plain text. Results are shown in figure 1, which roughly obey Zipf's Law except special high and low frequency words. Term distribution of plain text and lyrics are similar.

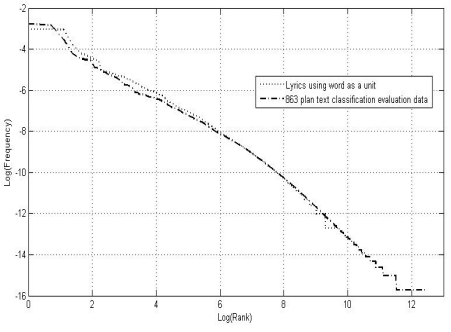


Fig. 1. Zipf's Law curves

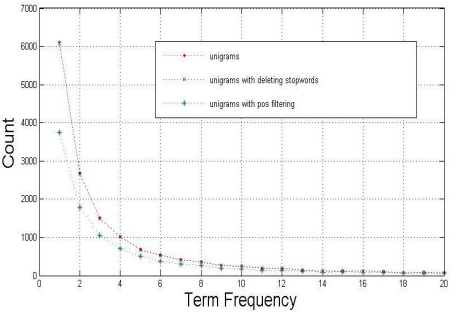


Fig. 2. Distribution of TF in lyrics corpus

We made a simple statistics on the distribution of term frequencies in the lyrics corpus, as displayed in Figure 2. The importance of language features are reflected by the parameter λ_i s assigned by a ME model. Therefore, we may assume that the prior distribution of the parameters is also Gaussian or exponential. In our previous work [14], it has been proven that ME with Laplace prior is significantly more suitable for text subjectivity analysis task. The corpus used is different from lyrics in Chinese.

5.2 Experiment Scheme

We tried our methods on three kinds of segmented data sets. First data set was just segmented lyrics. The second was deleted stop words. The third was filtered and only retained words with POS of adjective, verb, noun, and adverb. We focused on features based on unigrams, bigrams and trigrams. We ignored features that occurred 5 or fewer times in dataset and measured the performance using 5-fold cross validations. We tried three supervised learning methods to examine the performance. Besides that, we also applied the improved ME model with priors to music emotion recognition.

5.3 Experiment Results and Analysis

In these tables, the mark "ABS" stands for the absolute term frequency weighting, "BOOL" for Boolean weighting, and "TFIDF" for term frequency-inverse document frequency weighting method. Results are measured by accuracy.

5.3.1 Different Supervised Learning Algorithms

In three different preprocessing datasets, accuracies of ME and SVM with same features and weighting methods are nearly the same. ME with priors get best results.

5.3.2 Different Preprocessing

A little improvement has been made by deleting stop words with same conditions. Seen from Table 3, accuracies of lyrics with POS filtered processing decline a little with cases of Naïve Bayes and ME. However, SVM with POS filtered increases in Table 3. We suppose that the retained words with POS of adjective, verb, noun, and adverb represent main emotion expressed by songs.

Table 1. Accuracies (Lyrics segmented without deleting stop words)

Features	Weighting	NB	ME	SVM	ME+ GAU	ME+ EXP
Unigrams	ABS	64.37%	86.71%	86.76%	86.76%	**87.18%**
Uni+bigrams	ABS	66.74%	87.86%	**88.07%**	87.86%	87.13%
Uni+bi+trigrams	ABS	68.94%	87.39%	**87.91%**	87.39%	86.91%
Unigrams	BOOL	64.37%	86.71%	**87.49%**	86.76%	87.18%
Uni+bigrams	BOOL	66.74%	87.86%	**88.07%**	87.86%	87.13%
Uni+bi+trigrams	BOOL	68.94%	87.39%	**88.49%**	87.39%	86.91%
Unigrams	TFIDF	58.85%	89.28%	88.56%	90.28%	**90.65%***
Uni+bigrams	TFIDF	54.70%	89.28%	88.81%	90.28%	**90.49%**
Uni+bi+trigrams	TFIDF	53.86%	89.65%	89.28%	**90.65%***	90.12%

Table 2. Accuracies (Lyrics segmented and deleted stop words)

Features	Weighting	NB	ME	SVM	ME+ GAU	ME+ EXP
Unigrams	ABS	68.94%	**87.97%**	86.60%	**87.97%**	87.23%
Uni+bigrams	ABS	70.84%	**88.65%**	87.70%	**88.65%**	88.23%
Uni+bi+trigrams	ABS	70.94%	**88.49%**	88.28%	**88.49%**	87.23%
Unigrams	BOOL	68.94%	**87.97%**	87.81%	**87.97%**	87.23%
Uni+bigrams	BOOL	70.84%	88.65%	**88.75%**	88.65%	88.23%
Uni+bi+trigrams	BOOL	70.94%	88.86%	**89.23%**	88.49%	87.23%
Unigrams	TFIDF	61.32%	88.48%	89.02%	**89.49%**	90.17%
Uni+bigrams	TFIDF	55.33%	89.18%	89.34%	**90.18%**	89.65%
Uni+bi+trigrams	TFIDF	54.91%	88.70%	89.21%	**89.76%**	89.18%

Table 3. Accuracies (Lyrics segmented with POS filtered)

Features	Weighting	NB	ME	SVM	ME+ GAU	ME+ EXP
Unigrams	ABS	64.30%	**86.49%**	85.76%	**86.49%**	85.91%
Uni+bigrams	ABS	65.72%	**87.38%**	86.71%	87.28%	86.75%
Uni+bi+trigrams	ABS	66.14%	**87.65%**	86.23%	87.64%	86.70%
Unigrams	BOOL	64.30%	**86.49%**	86.39%	**86.49%**	85.91%
Uni+bigrams	BOOL	65.72%	87.38%	**88.39%**	87.28%	86.75%
Uni+bi+trigrams	BOOL	66.14%	87.65%	**87.81%**	87.64%	86.70%
Unigrams	TFIDF	59.99%	88.70%	89.20%	89.70%	**89.85%**
Uni+bigrams	TFIDF	55.42%	89.12%	89.43%	**90.49%**	90.43%
Uni+bi+trigrams	TFIDF	54.84%	89.17%	89.33%	**90.17%**	89.86%

5.3.3 Different Weighting Methods

In general, the performances of TFIDF are obviously better. Interestingly, accuracies of Naïve Bayes with TFIDF weighting drop 10%. We speculate that this perhaps due

to topic being conveyed mostly by particular content words that tend to be repeated, but this remains to be verified in music emotion recognition with Naïve Bayes.

5.3.4 Different n-Gram Features

It can be seen that all accuracies with same algorithm and weighting methods are improved by adding high order grams except Naïve Bayes with TFIDF weighting.

6 Conclusions and Future Work

In this paper, we investigate three kinds of preprocessing methods and a series of language grams Then we employ three prevail supervised learning. Experimental results show that feature extraction methods improve music emotion classification accuracies. ME with priors are suitable for music emotion recognition. Some supervised feature selection and extraction methods could be adopted to improve accuracy and efficiency.

References

1. Huron, D.: Perceptual and Cognitive Applications in Music Information Retrieval. In: ISMIR 2000 (2000)
2. Li, T., Ogihara, M.: Toward Intelligent Music Information Retrieval. IEEE Transactions on Multimedia 8(3), 564–574 (2006)
3. Scott, S., Matwin, S.: Text Classification Using WordNet Hypernyms. In: COLING-ACL 1998 Workshop, pp. 38–44 (1998)
4. Hevner, K.: Experimental Studies of the Elements of Expression in Music. Amer. J. Psychol. 48, 246–268 (1936)
5. Zheng, Y.B., Liu, Z.Y., Sun, M.S.: Statistical Features of Chinese Song Lyrics and Its Application to Retrieval. Journal of Chinese Information Processing (05), 61–67 (2007)
6. Wei, B., Zhang, C., Ogihara, M.: Keyword Generation for Lyrics. In: ISMIR 2007 (2007)
7. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment Classification Using Machine Learning Techniques. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Philadelphia, US, pp. 79–86 (2002)
8. Pang, B., Lee, L.: A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cus. In: Proceedings of 42nd Meeting of the Association for Computational Linguistics, Barcelona, ES, pp. 271–278 (2004)
9. Berger, A.L., Della Pietra, S.A., Della Pietra, V.J.: A Maximum Entropy Approach to Natural Language Processing. Computational Linguistics 22(1), 39–71 (1996)
10. Bahl, L., Jelinek, F., Mercer, R.: A Maximum Likelihood Approach to Continuous Speech Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 5(2), 179–190 (1983)
11. Chen, S.F., Rosenfeld, R.: A Gaussian prior for smoothing maximum entropy models. Tech. Rep. CMUCS-99-108, Carnegie Mellon University (1999)
12. Kazama, J., Tsujii, J.: Evaluation and Extension of Maximum Entropy Models with Inequality Constraints. In: Proc. EMNLP 2003, pp. 137–144 (2003)
13. Goodman, J.: Exponential Priors for Maximum Entropy Models, Microsoft Research Tech. Rep. (2003)
14. Chen, B., He, H., Guo, J.: Constructing Maximum Entropy Language Models for Movie Review Subjectivity Analysis. Journal of Computer Science and Technology (JCST) 23(2), 231–239 (2008)

Simultaneously Finding Fundamental Articles and New Topics Using a Community Tracking Method

Tieyun Qian[1], Jaideep Srivastava[2], Zhiyong Peng[3], and Phillip C.Y. Sheu[1,4]

[1] State Key Lab of Software Engineering, Wuhan Uviersity,
16 Luojiashan Road, Wuhan, Hubei, 430072, China
[2] Dept of Computer Science & Engineering, University of Minnesota, Twin Cities
[3] Computer School, Wuhan University
[4] Department of Electrical and Computer Engineering, University of California, Irvine
qty@whu.edu.cn, srivasta@cs.umn.edu, peng@whu.edu.cn,
psheu@uci.edu

Abstract. In this paper, we study the relationship between fundamental articles and new topics and present a new method to detect recently formed topics and its typical articles simultaneously. Based on community partition, the proposed method first identifies the emergence of a new theme by tracking the change of the community where the top cited nodes lie. Next, the paper with a high citation number belonging to this new topic is recognized as a fundamental article. Experimental results on real dataset show that our method can detect new topics with only a subset of data in a timely manner, and the identified papers for these topics are found to have a long lifespan and keep receiving citations in the future.

Keywords: Community tracking, Fundamental article finding, New topic identification.

1 Introduction

With the development of the World Wide Web, the number of on line publications has grown exponentially in recent years. For example, in the field of computer science, DBLP, a computer science bibliography site, has indexed more than one million articles and is still growing at a rapid pace. Due to the huge number of publications, it is clear that there are increasing costs for researchers to conduct the task of paper survey. A basic question every researcher would ask is: Which papers are worth reading?

Citation number is a bibliometric indicator to evaluate the importance of a contribution. It is widely adopted in both the traditional Institute of Scientific Information's Impact Factor (ISI IF) [1] and modern search engines such as Google Scholar, Microsoft Libra, and CiteSeer, etc. Citation number is calculated as follows: if there is another paper B citing an existing paper A, the citation count of A will increase by 1. It is clear that citation-based systems are unable to differentiate between research fields. Further more, the intention of researchers in practice is usually focused on looking for something new or interesting. This contradicts the principle of citation-based criterion, as citations accumulate over time.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 796–803, 2009.
© Springer-Verlag Berlin Heidelberg 2009

In this paper, we are particularly interested in the problem of finding significant articles even in the very early stages of a specific research topic. These articles should open up new scientific areas and have the potential to be continuously cited for quite a long period in the future. As a result, the problem of fundamental article finding is closely related to that of topic emergence detection.

There has been great interest in analyzing general research trend from large amount of history documents [2-4][13][14]. Recent research has found that changes in network structure could be used to trace topic development [5][6]. In this study, we first model communities in the citation graph as documents related by a specific topic. The task of topic emergence detection is then formulated as identification of new communities, which can be performed by comparing communities at two time periods. However, communities are always varying over time: some members will join while some members may leave; one community will divide, and the other may fuse. To address the problem of community tracking, we employ a "node-guided community" approach, which can identify a new topic community by tracking the communities where the top cited nodes lie. After a new theme is detected in such a way, the papers mostly cited at this stage are then recognized as fundamental articles. We evaluate the proposed approach on a real-world dataset. Results demonstrate that our method can detect new topics in a timely manner, and the found articles associated with such topics have a much longer lifespan than ordinary top-cited papers.

2 Related Work

Identifying influential articles in any field of research is important, and this task has thus attracted significant attention. As a result, many metrics have been proposed to evaluate the importance of publications. Besides the simple but widely adopted citation count [1] [7], a set of social network centrality metrics have been proposed for this purpose [8-11]. However, none of these metrics take topic issue into consideration. Consequently, the deficiencies we point out in the previous section exist in current evaluation systems. The work in [12] associated scientific theme with fundamental articles. However, their analysis is conducted on co-word frequency, which can reveal overall topic trends but fails to identify the exact time of topic emergence.

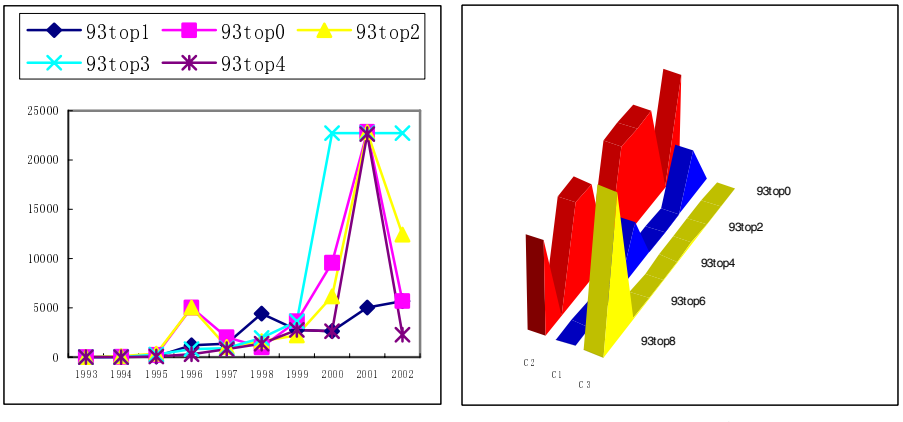
Topic trend analysis is another research tread in recent years. A variety of useful topic models have been proposed to discover summaries of documents. Notable is Latent Dirichlet Allocation (LDA) style topic model [2] [13]. The authors of [4] also presented an LDA-style topic model, associating each topic with a continuous distribution over timestamps. Unlike other work based on LDA, their model does not discretize time and does not make Markov assumptions for state transitions over time. In [3] latent themes are extracted from text using a probabilistic mixture model. The parameters in the model are estimated using an EM algorithm to obtain the theme word distributions. Generally speaking, current topic trend analysis can identify the overall tendency at a coarse granularity.

3 A Case Study on the Top K Highest Citation Articles

To illustrate the relationship between citation and research trend, we first trace the pattern of change in citations. We wish to ask a question for the *K* most cited paper: *How long can a highly cited paper keep holding researchers' interest?*

Fig.1 (a) reports the yearly citation ranking for the top 5 articles in 1993. One can observe from this figure that the citation ranking for these articles decreases dramatically over time. Four of them drop out of the top-10 list in 1994, and none of them keeps staying in top 300 by 1996. From 1998, all of the top 5 papers even leave the top-1000 sequence.

Next, we consider the problem of topic distribution of the top *K* articles. As the increasing breadth of research field has made it difficult for any scientist to have the energy to read papers from all sub-areas, it is important to classify papers into different topics. Fig.1 (b) shows that the top 10 articles in 1993 belong to 3 topic communities.



(a) (b)

Fig..1. (a) The yearly citation ranking for the top 5 articles in 1993 (b) The topic community to which the top 10 articles in 1993 belong

From Figure 1, following statements can be made: (1) A top cited paper may not have a high continuous citation over time. (2) Citation number does not reflect topic distribution, thus it can not help judge the importance of an article in a research theme.

4 Finding Fundamental Articles and New Topics

4.1 Fundamental Articles Identification

Let P be a scientific paper set $\{p_1 \mid t_1, p_2 \mid t_2, \ldots, p_n \mid t_n\}$, where p_i is the paper id and t_i is the time when p_i was published. Let C_P be a citation graph (V, E) with vertex set V and edge set E. Each vertex in V is an element from P. Each edge in E is of the form:

($<p_i, p_j>$, t_i), where $<p_i, p_j>$ is an edge if p_j has a citation from p_i and t_i is the timestamp of edge $<p_i, p_j>$ which is exactly same as when p_i is published.

As citation reflects the admission from other authors, we intend to select fundamental articles for researchers to read from this citation graph.

Given the above defined citation graph, citation number of a paper is the degree of a node. It is easy to choose the nodes with the highest degree for users. However, as we have illustrated in the previous section, (a) citation number (degree) does not differentiate topics, thus can not well meet information needs today, (b) a current highly cited paper may not hold researchers' interest for a long time period, (c) the most cited paper may have been out of date already. Thus the nodes with highest degree may not be papers we want to recommend.

According to the paradigms of Kuhn [17], the life of a scientific theme follows the logistic curve. It can be divided into several phases: birth, flourish, stagnation and death. Only those important papers in the first or second stage have the chance to survive to further periods. In contrast, the follow-up research causes an existing theme to saturate and mature. From this, we can reach the conclusion that fundamental articles lie in the newly formed topic, and topic emergence detection should be done before a fundamental article can be recommended as the one worth reading.

4.2 Article, Community and Topic

Citation is a sign of connection between articles having relevant content. All references among documents taken together comprise a citation network. Community structure is defined as groups of vertices within which connections are dense but between which they are sparse [15].

Observation 1: Each paper must be part of an existing community when it first comes into the network. The reason is that the paper needs to cite previously published bibliography.

Hypothesis 1: Generally speaking, the transformation of a community corresponds to the transformation of topic in the citation network. This hypothesis has been demonstrated by experimental results on Supreme Court dataset [5]. Experiments on KDD-Cup03 dataset also show very similar characteristics, which we discuss later.

Modeling topic community: Given a scientific paper set P and a citation graph C_P, we represent the community $T(\alpha)$ of a topicα as a subset of articles $\{ p_{a1}, p_{a2}, \ldots p_{am} \}$ in which the citation edges are more dense than those linked to any other community.

Community, cluster or group in other terminology, can be found by a variety of methods, such as community discovery, clustering and graph partitioning. However, current research on community discovery from documents works with only a static snapshot of network data [6]. These methods are unable to discover the temporal trends in community evolution. To the best of our knowledge, we are the first ones to focus on detecting the exact time a new community emerges. For this particular task, it is necessary to trace the evolution of community in the sequence of citation graph snapshots, and detect whether there is qualitative change in a topic community.

From Observation 1, all new papers join an old topic community before the best of them attract more followers and are powerful enough to form a new theme. This

indicates that a new topic community can not appear suddenly, but rather must evolve gradually. Thus we propose the following definition.

Definition 1. Indication of Topic emergence: When a sub-community (subset of a community) takes on the characteristics (over time) of a full-fledged community, a new topic is said to have emerged, represented by the sub-community.

Observation 2: A new research topic must include recent most cited papers (top papers), otherwise it could not draw enough attention and have no chance to develop into a full-fledged community.

Based on observation 2, we now present a "*node-guided community*" strategy to solve the problem of community tracking. That is, we first find the nodes that are recent most cited papers, and then we locate which communities these nodes are in. Two communities where the top nodes are located in successive time steps should be one community in different phases. We further have:

Definition 2. Topic emergence time: time step at which the top cited paper separates from the old community and forms a new community.

Definition 3. Fundamental article: the most cited articles that play key roles in the formation of new topics.

4.3 Topic Emergence and Fundamental Article Detection (TE-FAD) Algorithm

From definition 2 and definition 3, a new scientific theme emerges when an article node with the highest precedence separates from its old community. Hence, the new topic can be detected by tracking the communities where the nodes lie in. If the component parts of its communities at different time periods have changed significantly, then a new theme forms and the top cited papers in this community are just fundamental articles. Definition 2 involves how to distinguish two communities. There are many possible similarity (dissimilarity) criteria such as entropy, mutual information etc. For computational efficiency, in this paper we adopt Jaccard coefficient to measure the similarity between community C_i and C_j. The algorithm for topic emergence and fundamental article detection is shown in Figure 3.

In Figure 3, line 1 initiates some variables, lines 2-3 find topic communities of citation graphs at the all time period, and then assign each node to the community it belongs to. Lines 4-10 find the new topics and fundamental articles. Line 5 finds nodes with the top K largest degree in Gi. Lines 6-10 search the communities where a highly cited node is located at the previous time periods and then compute the similarity value of these communities. If the change of the similarity value is greater than the predefined threshold δ and this community has not entered the topic list, then lines 9-10 treat the current community as a newly formed topic and recognize the node as fundamental articles.

For step 3, i.e., the topic community finding algorithm, there exists a lots of algorithms to do this task. For a particularly implementation, we choose the fast modularity algorithm [15] because this algorithm is fast and parameter free.

```
Input:  G1, G2, …, Gn: Snapshots of a series of citation
graphs at time T1, T2, …, Tn.
Parameter: K: top cited paper set size, δ: similarity
control factor
Output: TList-Topic List, Alist-Fundamental Article List
Method:
1.  Tlist = ∅, Alist = ∅
2.  for i = 1 to n
3.     Find topic communities of G_i, denotes the topic
       community where node n_j lies as C_i(n_j).
4.  for i = 3 to n
5.     Find nodes with the top K largest degree in G_i
6.     for each node n_j with the K largest degree in G_i
7.        if newTopicEmerge(C_{i-2}(n_j),C_{i-1}(n_j),C_i(n_j),δ) then
8.           if (C_i(n_j) ∉ TList) then
9.                 Tlist  =  TList ∪ {C_i(n_j)}
10.                AList  =  AList ∪ {n_j}
```

Fig. 3. Algorithm for topic emergence and fundamental article detection (TE-FAD)

Line 7 in Fig.3 calls a procedure for detecting new topic, which is listed below.

```
//Procedure for new topic community emergence detection
newTopicEmerge(C_{i-1}, C_i, C_{i+1}, δ)
Input:  C_{i-1}, C_i, C_{i+1}: topic community at time T_{i-1},T_i,T_{i+1};
Parameter: δ: similarity control factor
Output:  true if a new topic emerges and C_{i+1} is a new
      topic community, false otherwise.
Method:
1.    s_1 = sim(C_{i-1}, C_i);
2.    s_2 = sim(C_i, C_{i+1});
3.    if   s_2-s_1 > δ
4.    then return true;
5.    else return false;
```

5 Experimental Results

We have conducted experiments on hep-th (high-energy physics theory) dataset from KDD Cup 2003 [16], which is based on arXiv, an e-print service in the fields of physics and some related disciplines. The citation structure includes *29555* papers and *352807* citations from January 1993 through May 2003. For each month *m* we create a sub-graph using all vertices whose timestamps are earlier than month *m*. Thus, we have a sequence of time periods T_1=*Jan. 1993*, T_2=*Feb.1993*, … , T_{125}=*May 2003* and the relevant time sub-graph G_1, G_2…. G_{125}.

In order to find the exact time when topic community appears, we ran the proposed TE-FAD algorithm on G_1, $G_2 \ldots G_{124}$. The parameter values we used were $K = 5$, $\delta = 0.35$ and a total number of *22* topics was discovered. In Table 1, we illustrate the emergence time, initial size and final size of the three largest topic communities, as well as the turnover value of similarity. As can be seen in Table 1, topic C_1, C_2 and C_3 appeared at Aug. 1996, Dec. 1993 and Dec. 1998, respectively. Please note that these time points are obtained using time sub-graph G_{41}, G_{24} and G_{72}. It is a predictive analysis rather than a reviewing analysis. In addition, if we compare the 4th column with the 5th column in Table 1, it is clear that the initial sizes of these three communities are much smaller than their final sizes. That is to say, these topic communities are still in their very early stage when they are detected by our algorithm.

Table 1. Topic communities

T-Id	Start Time T_0	Turnover Value	Initial Size	Final Size
C1	1996.08	44.64 %	1446	7331
C2	1993.12	87.44 %	128	10938
C3	1998.12	56.18 %	2753	6368

Table 2. Fundamental articles in each topic

T-Id	Paper-Id	Time	Total Citation	Citation After T_0
C1	# 9410167	1994-10-25	748	273
	# 9503124	1995-03-21	1114	739
	# 9602022	1996-02-6	520	370
	# 9407087	1994-07-19	1299	878
C2	# 9210010	1992-09-29	156	95
C3	# 9711200	1997-11-28	2414	1966
	# 9802109	1998-02-17	1641	1329
	# 9802150	1998-02-23	1775	1431

Table 2 presents the fundamental articles according to *TE-FAD* algorithm. For each paper, we show the *topic-id, paper-id, the year of publication,* and *the number of citations*. We can see that articles identified by our algorithm all have a high citation number, even after the emergence time of its relevant topic. This reveals that *TE-FAD* can help find articles that ought to be read shortly after their publication.

6 Conclusion

In this paper we proposed a topic emergence and fundamental article detection algorithm *TE-FAD* which is based on community division and citation number. It can identify new theme emergence in a timely manner, using data up to a specific moment, while identifying fundamental articles in the new topics simultaneously. Experimental results demonstrate that the topics found by our algorithm have semantic characteristic and the articles have a much longer life span than other top cited

papers. We believe this work is a credible attempt to bridge citation graph analysis with scientific theme detection, as well as the fundamental article identification and recommendation. The framework introduced in this paper can apply to many other applications, such as literature visualization, knowledge organization, and decision support.

Acknowledgments. This research was supported in part by the 111 Project (B07037), the NSFC Project (60873007), the NSF of Hubei Province (2008CDB340), the 973 Project (2007CB310801) and the Key Project of Chinese Ministry of Education (107072).

References

[1] Garfield, E.: Citation analysis as a tool in journal evaluation. Science 178(60), 471–479 (1972)
[2] Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proceedings of the National Academy of Sciences 101(suppl. 1), 5228–5235 (2004)
[3] Mei, Q., Zhai, C.: Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In: Proc. of the 11th SIGKDD (2005)
[4] Wang, X., McCallum, A.: Topics over Time: A Non-Markov Continuous Time Model of Topical Trends. In: Proc. of the 12th ACM SIGKDD (2006)
[5] Leicht, E.A., Clarkson, G., Shedden, K., Newman, M.E.J.: Large-scale structure of time evolving citation networks. Eur. Phys. J. B 59, 75–83 (2007)
[6] Zhou, D., Councill, I., Zha, H., Lee Giles, C.: Discovering Temporal Communities from Social Network Documents. In: Proc. of the 7th ICDM (2007)
[7] Lehmann, S., Jackson, A.D., Lautrup, B.E.: Measures and mismeasures of scientific quality (2005)
[8] Wasserman, S., Faust, K.: Social Network Analysis: Methods and Applications. Cambridge University Press, Cambridge (1994)
[9] Brin, S., Page, L.: The Anatomy of a large-scale hypertextual web search engine. WWW7/Computer Networks 30(1-7), 107–117 (1998)
[10] Kleinberg, J.: Authoritative Sources in a Hyperlinked Environment. Journal of the ACM 46(5), 604–632 (1999)
[11] Bollen, J., Van de Sompel, H., Smith, J., Luce, R.: Toward alternative metrics of journal impact: A comparison of download and citation data. Information Processing & Management 41(6) (December 2005)
[12] Cahlik, T.: Search for fundamental articles in economics. Scientometrics 49(3) (2000)
[13] Blei, D., Ng, A., Jordan, M.: Latent Dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
[14] Steyvers, M., Smyth, P., Rosen-Zvi, M., Griffiths, T.: Probabilistic Author-Topic Models for Information Discovery. In: Proc. of the tenth ACM SIGKDD (2004)
[15] Newman, M.E.J.: Fast algorithm for detecting community structure in networks. Phys. Rev. E 69 (2004)
[16] http://www.cs.cornell.edu/projects/kddcup/
[17] Kuhn, T.S.: The structure of scientific revolutions. Chicago University Press (1970)

Towards a Novel Association Measure via Web Search Results Mining

Xiaojun Wan and Jianguo Xiao

Institute of Computer Science and Technology,
Peking University, Beijing 100871, China
{wanxiaojun,xiaojianguo}@icst.pku.edu.cn

Abstract. Web-based association measure aims to evaluate the semantic similarity between two queries (i.e. words or entities) by leveraging the search results returned by search engines. Existing web-relevance similarity measure usually considers all search results for a query as a coarse-grained single topic and measures the similarity between the term vectors constructed by concatenating all search results into a single document for each query. This paper proposes a novel association measure named WSRCM based on web search results clustering and matching to evaluate the semantic similarity between two queries at a fine-grained level. WSRCM first discovers the subtopics in the search results for each query and then measures the consistency between the sets of subtopics for two queries. Each subtopic for a query is expected to describe a unique facet of the query, and two queries sharing more subtopics are deemed more semantically related. Experimental results demonstrate the encouraging performance of the proposed measure.

Keywords: Association Measure, Query Similarity, WSRCM.

1 Introduction

The study of measuring semantic similarity between words or short text segments (e.g. entities) has become very important for many web-related tasks, including word clustering [4], name disambiguation [2] and community mining [1], etc. In recent years, web-based association measures have been well studied to evaluate the semantic similarity between two words or entities [1, 2, 9, 11]. In contrast with knowledge-based measures relying on existing knowledge databases or taxonomies (e.g. WordNet) [5], web-based measures make use of the up-to-date web search results returned by web search engines (e.g. Yahoo Search) and they can reflect the up-to-date semantic similarity between two words or entities. Moreover, web-based measures can be successfully applied to compute the semantic similarity between new words or entities, which are usually not defined in any existing knowledge database.

Web-based similarity kernel and web-relevance similarity measure [9, 11] are typical association measures to leverage web search results for evaluating semantic

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 804–812, 2009.
© Springer-Verlag Berlin Heidelberg 2009

similarity between two queries (i.e. words or entities), and they take all search results for a query as a coarse-grained single topic and use a single term vector to represent all the search results, and then the similarity value between two queries is obtained by computing the inner-product between the term vectors for the queries. In this study, the web-relevance similarity measure is implemented by first concatenating all the titles and snippets in the search results into a single document $D(q_i)$ for each query q_i and then computing the cosine similarity between documents as the semantic similarity between two queries q_1 and q_2.

$$WebRelevance(q_1, q_2) = sim_{cosine}(D(q_1), D(q_1)) = \frac{\vec{D}(q_1) \cdot \vec{D}(q_2)}{\left|\vec{D}(q_1)\right| \times \left|\vec{D}(q_2)\right|} \quad (1)$$

where $\vec{D}(q_i)$ is the corresponding term vector of document $D(q_i)$.

However, for each short query, the number of the returned search results is usually very large and the results usually contain diverse information about the query: a cluster of results reflects one facet of the query, and another cluster of results reflects another facet of the query. For example, some results for query "apple" are about the apple fruit, while other results are about the Apple computer. Even for the Apple computer, the results can be also grouped into a few clusters, each cluster referring to a specific aspect of the Apple computer, such as its price, its hardware, its software, and so on. It is not appropriate to simply take all the diverse results into one single vector for each query. Therefore, we assume that the search results for a query can be organized into a few subtopics about the query, each subtopic representing a specific facet of the query. If the subtopic sets of two queries (e.g. "car" and "automobile") are highly consistent with each other, the queries are semantically similar from the whole perspective; if only some subtopics of the two queries (e.g. "apple" and "fruit") are consistent with each other, the queries are semantically similar on some facets denoted by the common subtopics; and if no subtopics of the two queries (e.g. "sun" and "school") are consistent with each other, the queries are not semantically similar at all.

In this study, we propose a novel association measure named WSRCM (Web Search Results Clustering & Mining) to evaluate semantic similarity between words or entities by first discovering the subtopics in the search results for each query and then measuring the consistency between the subtopic sets, rather than taking all search results for each query as a single topic represented by a single vector. The k-means clustering algorithm is used for grouping the search results and discovering the subtopics. And then the optimal matching method is employed to measure the consistency between the subtopic sets by formalizing the problem as the optimal matching problem in the graph theory. The normalized optimal matching value is used as the semantic similarity value between the queries. Experiments have been performed on the benchmark Miller Charles' dataset, and the results demonstrate the good effectiveness of the proposed WSRCM measure. Moreover, the performance is improved by using the ensemble techniques to combine a few similarity measures.

2 The Proposed WSRCM Measure

2.1 Overview

The basic idea of the proposed WSRCM measure is to measure the semantic similarity between two queries by measuring the consistency between the discovered subtopics related to the queries, under the assumption that the corresponding subtopics in the search results for a query can reflect the query at a fine-grained level, while the single topic representation of the search results for the query in previous work is coarse-grained.

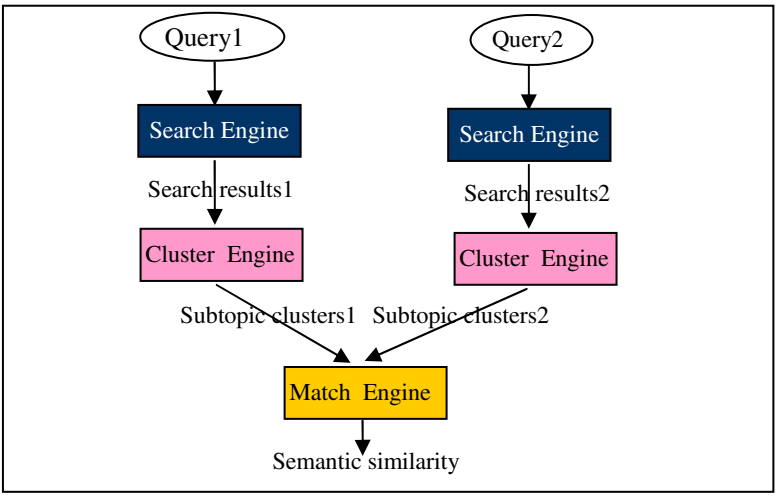


Fig. 1. Framework of WSRCM

The framework of WSRCM is given in Figure 1 and it consists of the following three steps:

1) **Search result retrieval:** Each word or entity is issued as a query into a commercial search engine and the search results are retrieved for the query.

2) **Search results clustering:** A clustering algorithm is applied on the search results for each query to obtain a set of clusters, each cluster representing a subtopic related to the query.

3) **Subtopic cluster matching:** Given the two subtopic sets for two queries, the optimal matching method is used for measuring their consistency and the semantic similarity value is returned.

The details of the above steps will be described as follows, respectively.

2.2 Search Results Retrieval

In this study, we base our experiments on *Yahoo Search* because *Yahoo Search* outperforms *Google Search* and *Live Search* for most web-based measures in our pilot study. *Yahoo Search* is one of the most popular search engines used today and it

returns the estimated result number and at most 1000 results for each query. We extract each result record consisting of the title, snippet and url from the returned result page. Instead of downloading the full web pages, we use only the titles and snippets in the search results for efficient computation. The title and snippet are concatenated into a single text for each search result. All the 1000 (or fewer) returned search results are used in this study. We let n denote the number of the top search results used in this study and n is set to 1000 by default.

2.3 Search Results Clustering

It has been a challenging task to group search results into meaningful clusters. Various methods have been proposed to address this problem, and they aim to facilitate users to browse the search results and efficiently reach required web pages [12, 13]. However, the clustering results are far from satisfactory and most users still prefer to browsing traditional ranked list of search results. Note that it is not the focus of this study to group search results into accurate and meaningful clusters for user browsing, so we simply adopt a popular document clustering algorithm to cluster the search results into a few clusters and we expect the clusters can represent the real subtopics for the query to some extent. We choose the efficient k-means algorithm for search results clustering.

The k-means algorithm is a partition based clustering algorithm. After the cluster number k is given, the algorithm randomly selects k documents as the initial centroids of the k clusters and then iteratively assigns all documents to the closest cluster, and recomputes the centroid of each cluster, until the centroids do not change. The similarity value between a document and a cluster centroid is computed using the standard cosine measure. It is not a trivial task to predict the cluster number k, and in this study we heuristically set $k=n/10$, which means that the average document number in each cluster is 10. The time complexity of the k-means algorithm is $O(nkl)$, where l is the number of iterations taken by the algorithm to converge.

We then filter out the trivial clusters with less than two documents from the k clusters. We believe that such clusters are outliers in the clustering results and they cannot represent the real subtopics for the given query. Lastly, only the remaining non-trivial clusters are used for cluster matching.

To our knowledge, there exist advanced clustering algorithms which can find more accurate and meaningful subtopics from the search results, which will be exploited in our future work.

2.4 Subtopic Cluster Matching

Given two sets of subtopic clusters X and Y for two queries q_1 and q_2, this step aims to measure the consistency between the sets of topic clusters from a global perspective. We formalize this problem as the optimal matching problem and allow only one-to-one matching between the topic clusters. A globally optimal solution can be achieved by solving the optimal matching problem.

Optimal matching (OM) is a classical problem in graph theory. Let $G=\{X, Y, E\}$ be a weighted bipartite graph, where $X=\{x_1,x_2,\ldots,x_p\}$ and $Y=\{y_1,y_2,\ldots y_q\}$ are the partitions representing the two sets of subtopic clusters for queries q_1 and q_2. x_i and y_j also refer to the cluster texts by concatenating all the result texts in the clusters respectively. $V=X \cup Y$ is the vertex set, and $E=\{e_{ij}\}$ is the edge set with each edge e_{ij} connects a

subtopic cluster x_i in X and a subtopic cluster y_j in Y. A weight w_{ij} is assigned to every edge e_{ij} in G. The weight w_{ij} is set to the cosine similarity value between the two cluster texts as follows:

$$w_{ij} = sim_{\cos ine}(x_i, y_j) \tag{2}$$

A matching M of G is a subset of the edges with the property that no two edges of M share the same node. Given the weighted bipartite graph G, OM is to find the matching $\tilde{M}$ that has the largest total weight.

The Kuhn-Munkres algorithm [3, 7] is employed to solve the OM problem. The computational complexity of Kuhn-Munkres algorithm is $O((p+q)^3)$, where p, q are the numbers of vertices in two partitions in the bipartite graph respectively. Faster version of optimal matching algorithms exists [10] and can run in $O((p+q)(m+(p+q)\log(p+q))$, where m is the number of matching edges.

Lastly the optimal matching $\tilde{M}$ in graph G is acquired and we use the normalized total weight in $\tilde{M}$ as the semantic similarity value between queries q_1 and q_2:

$$WSRCM(q_1, q_2) = \frac{\sum_{e_{ij} \in \tilde{M}} w_{ij}}{\min\{|X|, |Y|\}} \tag{3}$$

where $\min\{|X|,|Y|\}$ returns the minimum subtopic number in X and Y.

3 Empirical Evaluation

We compare various unsupervised web-based association measures based on the popular Miller-Charles dataset [6], which contains 30 word-pairs[1] rated by a group of 38 human subjects. The word pairs are rated on a scale from 0 (no similarity) to 4 (perfect synonymy). The dataset is actually a subset of Rubenstein-Goodenough's original dataset [8]. The Miller-Charles dataset has been widely used in previous work for evaluating word similarity measures.

We use the Pearson product-moment correlation coefficient to measure the correlation between automatic computed values and human-labeled values. Given two ratings A and B, the Pearson's correlation coefficient is computed as follows:

$$r = \frac{\sum_{i=1}^{n} (a_i - \bar{a})(b_i - \bar{b})}{(n-1)s_a s_b} \tag{4}$$

where $\bar{a}$ and $\bar{b}$ are respectively the sample means of a_i in A and b_i in B, and s_a and s_b are sample standard deviations of a_i and b_i and n is the total number of samples. The correlation coefficient r belongs to the interval [-1,1]. r=0 indicates no correlation, and r=1 indicates perfect positive correlation, and r=-1 indicates perfect negative correlation. In this study, we let r=0 if r<0 in order to make the following geometric mean ensemble meaningful.

[1] In this study, we follow most previous research work [1, 2] and use only 28 pairs for evaluations, because of the omission of two word pairs in earlier versions of WordNet.

Our proposed measure (WSRCM) is compared with several popular web-based measures, including WebJaccard, WebDice, WebOverlap, WebPMI [2] and WebRelvence. Furthermore, we investigate the following two ensemble measures to fuse the well-performing measures in the experiments:

Fusion1 (Average Mean): The similarity value is the average of the similarity values computed by WebPMI, WebRelevance and WSRCM, respectively.

Fusion2 (Geometric Mean): The similarity value is the geometric mean of the similarity values computed by WebPMI, WebRelevance and WSRCM, respectively.

Table 1 shows the results and the similarity values for each measure is normalized by dividing by the maximum value.

Seen from Table 1, the proposed WSRCM measure outperforms all the popular web-based measures. In particular, WSRCM much outperforms the WebRelevance measure, which validates the assumption of WSRCM that it can evaluate word semantic similarity at a fine-grained level based on the discovered subtopics. We can also see that the two ensemble measures outperform all individual measures, which demonstrates the good effectiveness of the ensembles. Moreover, we can see that the Fusion2 measure (geometric mean) outperforms the Fusion1 measure (average mean).

Table 1. Performance comparison on Miller Charles' dataset

Word Pair	Miller-Charles	Web Jaccard	Web Dice	Web Overlap	Web PMI	Web Relevance	WSRCM	Fusion1	Fusion2
cord-smile	0.13	0.066	0.076	0.051	0.200	0.049	0.518	0.330	0.250
rooster-voyage	0.08	0.019	0.023	0.035	0.174	0.067	0.576	0.352	0.276
noon-string	0.08	0.097	0.112	0.058	0.223	0.056	0.519	0.344	0.272
glass-magician	0.11	0.047	0.054	0.184	0.321	0.045	0.484	0.366	0.279
monk-slave	0.55	0.147	0.168	0.088	0.554	0.061	0.631	0.537	0.405
coast-forest	0.42	0.340	0.376	0.184	0.315	0.101	0.675	0.470	0.405
monk-oracle	1.1	0.035	0.041	0.041	0.049	0.037	0.489	0.248	0.140
lad-wizard	0.42	0.043	0.050	0.045	0.364	0.092	0.652	0.477	0.408
forest-graveyard	0.84	0.051	0.059	0.183	0.458	0.049	0.503	0.435	0.328
food-rooster	0.89	0.017	0.019	0.336	0.166	0.051	0.524	0.319	0.239
coast-hill	0.87	0.477	0.515	0.207	0.357	0.115	0.738	0.522	0.456
car-journey	1.16	0.222	0.249	0.330	0.114	0.073	0.452	0.275	0.227
crane-implement	1.68	0.071	0.082	0.071	0.146	0.079	0.645	0.375	0.285
brother-lad	1.66	0.088	0.101	0.231	0.580	0.075	0.605	0.542	0.434
bird-crane	2.97	0.169	0.191	0.229	0.498	0.111	0.571	0.508	0.461
bird-cock	3.05	0.118	0.135	0.121	0.304	0.114	0.602	0.440	0.402
food-fruit	3.08	0.441	0.479	0.822	0.403	0.317	0.794	0.652	0.680
brother-monk	2.82	0.198	0.223	0.379	0.539	0.082	0.733	0.583	0.465
asylum-adhouse	3.61	0.030	0.035	0.084	0.876	0.095	0.761	0.746	0.582
furnace-stove	3.11	0.342	0.377	0.186	1.000	0.181	0.921	0.905	0.803
magician-wizard	3.5	0.155	0.176	0.176	0.718	0.126	0.796	0.706	0.608
journey-voyage	3.84	0.176	0.199	0.120	0.370	0.187	0.698	0.540	0.531
coast-shore	3.7	0.371	0.407	0.339	0.554	0.203	0.931	0.727	0.688
implement-tool	2.95	1.000	1.000	1.000	0.531	0.171	0.748	0.625	0.596
boy-lad	3.76	0.092	0.105	0.405	0.579	0.212	0.664	0.627	0.632
automobile-car	3.92	0.330	0.365	0.747	0.322	1.000	1.000	1.000	1.000
midday-noon	3.42	0.066	0.076	0.098	0.563	0.189	0.611	0.587	0.586
gem-jewel	3.84	0.328	0.362	0.165	0.697	0.152	0.767	0.696	0.632
Correlation	1	0.324	0.336	0.413	0.618	0.510	0.703	0.780	**0.820**

[2] The WebJaccard, WebDice, WebOverlap, WebPMI measures rely on the number of hits returned by the web search engine, and they are defined the same as in [1].

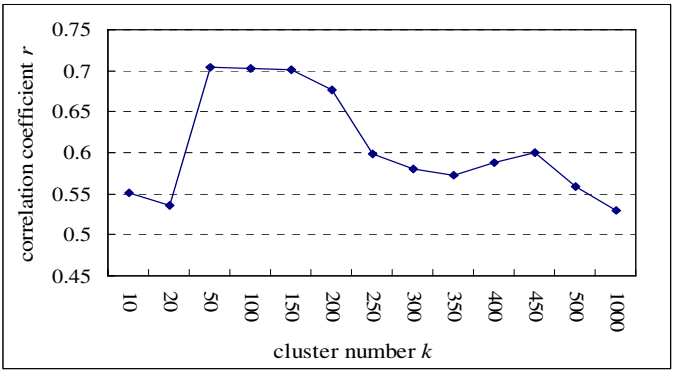


Fig. 2. Correlation vs. cluster number for WSRCM

In the above experiments, the expected cluster number k is set to $n/10=100$, where $n=1000$ is the number of returned search results for a query. We now vary k from 10 to 1000 when n is fixed to 1000. Figure 2 shows the curve of correlation results. Seen from the figure, WSRCM performs well when k is set in the range of [50, 200], though the performance is affected by the cluster number: very small or very large clusters will deteriorate the performance because the clusters cannot reflect the true subtopics in the search results for the query.

We now compare WSRCM, WebRelevance and the two ensembles with respect to different numbers of top search results. The number n of top results used in the above measures varies from 100 to 1000. For WSRCM, the corresponding cluster number k then varies from 10 to 100. Figure 3 gives the comparison results. We can see that when few top results ($n\leq300$) are used for computation, WSRCM does not perform well, even worse than WebRelevance. While when more top results ($n\geq400$) are used, WSRCM can always outperform WebRelevance. Moreover, the best performance for WSRCM (when $n=1000$) is better than the best performance for WebRelevance (when $n=100$). We explain the results by that the clustering of few top results might

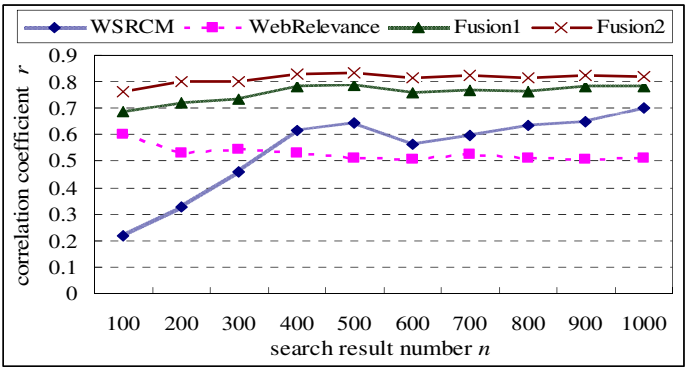


Fig. 3. Correlation vs. result number

not produce good subtopic clusters because the few top results are usually very coherent. We can also see that the two ensemble measures can always outperform the individual measures and the geometric mean ensemble always outperforms the average mean ensemble.

All the above experiments were conducted based on the search results obtained on Nov. 14, 2007. To our knowledge, the search results for a given query can vary frequently due to the dynamic indexing of web pages for the search engine. In order to investigate how the change of search engine affects the performance of the association measures, we conducted the experiments once a week from Nov. 14, 2007 and Jan. 2, 2008, and there were totally eight dates when we repeated the experiments. Figure 4 shows the correlation results of the association measures with respect to different dates.

We can see from Figure 4 that almost all measures are affected by the time dimension, which demonstrates that the search engine does return different results for a given query at different time. Fortunately, we observe that the proposed WSRCM measure is effective for most dates and the two ensemble measures are always performing very well. Moreover, the geometric mean ensemble always outperforms the average mean ensemble.

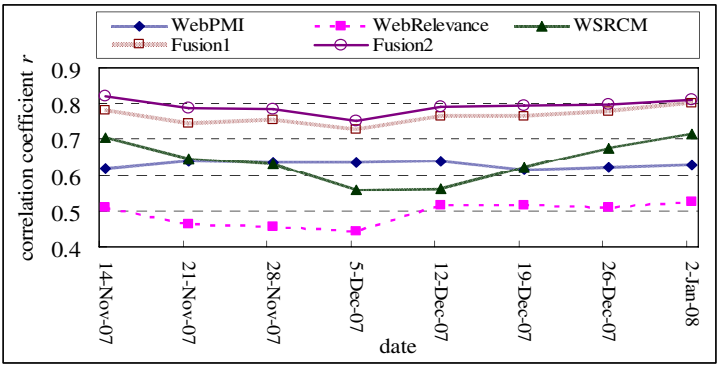


Fig. 4. Performance vs. date

In future work, we will explore advanced clustering algorithm to better discover meaningful subtopic clusters from search results. We will apply the propose measure to other web-related tasks (e.g. query suggestion) to validate its robustness.

Acknowledgments. This work was supported by NSFC (60703064,60873155), RFDP (20070001059), National High-tech R&D Program (2008AA01Z421) and NCET.

References

1. Bollegala, D., Matsuo, Y., Ishizuka, M.: Measuring semantic similarity between words using web search engines. In: Proceedings of WWW 2007 (2007)
2. Chen, H.-H., Lin, M.-S., Wei, Y.-C.: Novel association measures using web search with double checking. In: Proceedings of COLING-ACL 2006 (2006)

3. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Res. Logist. Quart.~2, 83--97 (1955)
4. Matsuo, Y., Sakaki, T., Uchiyama, K., Ishizuka, M.: Graph-based word clustering using web search engine. In: Proc. of EMNLP 2006 (2006)
5. Mihalcea, R., Corley, C., Strapparava, C.: Corpus-based and knowledge-based measures of text semantic similarity. In: Proceedings of AAAI 2006 (2006)
6. Miller, G., Charles, W.: Contextual correlates of semantic similarity. Language and Cognitive Processes~6(1), 1--28 (1998)
7. Munkres, J.: Algorithms for the assignment and transportation problems. J. Soc. Indust. Appl. Math.~5, 32--38 (1957)
8. Rubenstein, H., Goodenough, J.B.: Contextual Correlates of Synonymy. Communications of the ACM~8(10), 627--633 (1965)
9. Sahami, M., Heilman, T.: A web-based kernel function for measuring the similarity of short text snippets. In: Proc. of WWW 2006 (2006)
10. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency, vol.~A. Springer, Berlin (2003)
11. Yih, W.-T., Meek, C.: Improving similarity measures for short segments of text. In: Proceedings of AAAI 2007 (2007)
12. Zamir, O., Etzioni, O.: Grouper: A dynamic clustering interface to web search results. In: Proceedings of WWW 1999 (1999)
13. Zeng, H.-J., He, Q.-C., Chen, Z., Ma, W.-Y.: Learning to cluster web search results. In: Proceedings of SIGIR 2004 (2004)

A New Local Distance-Based Outlier Detection Approach for Scattered Real-World Data

Ke Zhang[1], Marcus Hutter[1,2], and Huidong Jin[1,2,3]

[1] RSISE, Australian National University
[2] National ICT Australia (NICTA), Canberra Lab, ACT, Australia
[3] CSIRO Mathematical and Information Sciences, Acton ACT 2601, Australia
{ke.zhang, marcus.hutter}@rsise.anu.edu.au, Warren.Jin@csiro.au

Abstract. Detecting outliers which are grossly different from or inconsistent with the remaining dataset is a major challenge in real-world KDD applications. Existing outlier detection methods are ineffective on scattered real-world datasets due to implicit data patterns and parameter setting issues. We define a novel *Local Distance-based Outlier Factor* (LDOF) to measure the outlier-ness of objects in scattered datasets which addresses these issues. LDOF uses the relative location of an object to its neighbours to determine the degree to which the object deviates from its neighbourhood. We present theoretical bounds on LDOF's false-detection probability. Experimentally, LDOF compares favorably to classical KNN and LOF based outlier detection. In particular it is less sensitive to parameter values.

Keywords: local outlier; scattered data; k-distance; KNN; LOF; LDOF.

1 Introduction

Of all the data mining techniques that are in vogue, outlier detection comes closest to the metaphor of mining for nuggets of information in real-world data. It is concerned with discovering the exceptional behavior of certain objects [TCFC02]. Outlier detection techniques have widely been applied in medicine (e.g. adverse reactions analysis), finance (e.g. financial fraud detection), security (e.g. counter-terrorism), information security (e.g. intrusions detection) and so on. In the recent decades, many outlier detection approaches have been proposed, which can be broadly classified into several categories: distribution-based [Bar94], depth-based [Tuk77], distance-based (e.g. KNN) [KN98], cluster-based (e.g. DBSCAN) [EKSX96] and density-based (e.g. LOF) [BKNS00] methods.

However, these methods are often unsuitable in real-world applications due to a number of reasons. Firstly, real-world data usually has a scattered distribution, where objects are loosely distributed in the domain feature space, e.g. like stars in the universe, forming many mini-clusters rather than a few main clusters. Only the objects which do not belong to any mini-cluster are genuine outliers. Secondly, most outlier detection approaches require fine-tuning of their parameters through trial-and-error [FZFW06], which is impractical, because real-world data

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 813–822, 2009.
© Springer-Verlag Berlin Heidelberg 2009

usually do not contain labels for anomalous objects. Top-n style outlier detection methods alleviate the parameter setting problem somewhat. They provide a ranked list of objects that represent the degree of 'outlier-ness' of each object.

Top-n K^{th}-Nearest Neighbour distance [RRS00] is a typical top-n style outlier detection approach. In top-n KNN outlier, the distance from an object to its k^{th} nearest neighbour (denoted as k-distance for short) indicates outlier-ness of the object. Intuitively, the larger the k-distance is, the higher outlier-ness the object has. Top-n KNN outlier regards the n objects with the highest values of k-distance as outliers [RRS00].

A density-based outlier, Local Outlier Factor (LOF) [BKNS00], was proposed in the same year as top-n KNN. In LOF, an outlier factor is assigned for each object w.r.t its surrounding neighbourhood. The outlier factor depends on how the data object is closely packed in its locally reachable neighbourhood [FZFW06]. In recent real-world applications, researchers have found it more reliable to use LOF in a top-n manner [TCFC02], i.e. only objects with the highest LOF values will be considered outliers. Hereafter, we call it top-n LOF.

In this paper, we propose a new outlier detection definition which is sensitive to outliers in scattered datasets, named Local Distance-based Outlier Factor (LDOF). LDOF uses the relative distance from an object to its neighbours to measure how much objects deviate from their scattered neighbourhood. The higher the violation degree an object has, the more likely the object is an outlier.

In Section 2, we illustrate and discuss the problems of top-n KNN and top-n LOF on a real-world dataset. In Section 3, we formally introduce the outlier definition of our approach, and mathematically analyse properties of our outlier-ness factor in Section 4. In Section 5, the top-n LDOF outlier detection algorithm is described, together with an analysis of its complexity. Experiments are reported in Section 6, which show the superiority of our method to previous approaches, at least on the considered datasets. Finally, conclusions are presented in Section 7.

2 Problem Formulation

In real-world datasets, high dimensionality (e.g. 30 features) and sparse feature value range usually cause objects to be scattered in the feature space. The scattered data is similar to the distribution of stars in the universe. Locally, they seem to be randomly allocated in the night sky (i.e. stars observed from the Earth), whereas globally the stars constitute innumerable galaxies. Figure 1(a) illustrates a 2-D projection of a real-world dataset, *Wisconsin Diagnostic Breast Cancer* (WDBC)[1], which is typically 30-D. The green points are the benign diagnosis records (regarded as normal objects), and the red triangles are malignant diagnosis records (i.e. outliers we want to capture). Obviously, we cannot detect these outliers in 2-D space, whereas in high dimension (e.g. 30-D), these scattered normal objects constitute a certain number of loosely bounded mini-clusters, and we are able to isolate genuine outliers. Unlike galaxies, which always contain billions of stars, these mini-clusters in scattered datasets usually have

[1] WDBC dataset is from UCI ML Repository: http://archive.ics.uci.edu/ml

a relatively small number of objects. Figure 1(b) is a simple demonstration of this situation, where C_1 is a well-shaped cluster as we usually define in other outlier detection methods. C_2 and C_3 are comprised of scattered objects with loose boundary, called mini-clusters. These small clusters should be recognised as 'normal', even if they contain a small number of objects. The objects of our interest are the points lying far away from other mini-clusters. Intuitively, o_1, o_2, o_3, o_4 are outliers in this sample. We recall a well accepted informal outlier definition proposed by Hawkins [Haw80]: *"An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism"*. In scattered datasets, an outlier should be an object deviating from any other group of objects.

The only way in which our outlier definition differs from others (e.g. in [KN98] and [BKNS00]) is that the normal pattern of data is represented by scattered objects, rather than crowded main clusters. The neighbourhood in scattered real-world datasets has two characteristics: (1) objects in mini-clusters are loosely distributed; (2) when neighbourhood size k is large, two or more mini-clusters are taken into consideration. The neighbourhood becomes sparse as more and more objects which belong to different mini-clusters should be taken into account.

As discussed above, top-n KNN and top-n LOF are ineffective for scattered datasets. Take a typical example, in Figure 1(b), when k is greater than the cardinality of C_3 (10 in this case), some objects in C_1 become neighbours of the objects in C_3. Hence, for top-n KNN, the k-distance of the object can be larger than genuine outliers. For top-n LOF, since the density of C_3 is smaller than that of C_1, it also fails for ranking o_1, o_2, o_3 and o_4 in the highest outlier-ness positions. In Section 6, we will demonstrate that the two methods fail to detect genuine outliers when k grows greater than 10.

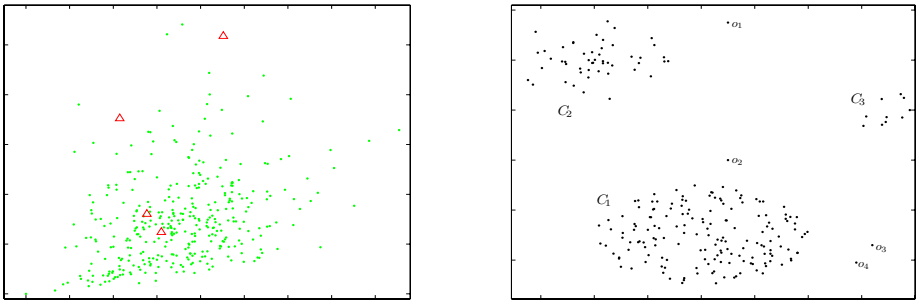
Intuitively, it is more reasonable to measure how an object deviates from its neighbourhood system as an outlier-ness factor rather than global distance (top-n KNN) or local density (top-n LOF). Thereby, we propose LDOF to measure the degree of neighbourhood violation. The formal definition of LDOF is introduced in the following section.

3 Formal Definition of Local Distance-Based Outliers

In this section, we develop a formal definition of the Local Distance-based Outlier Factor, which avoids the shortcomings presented above.

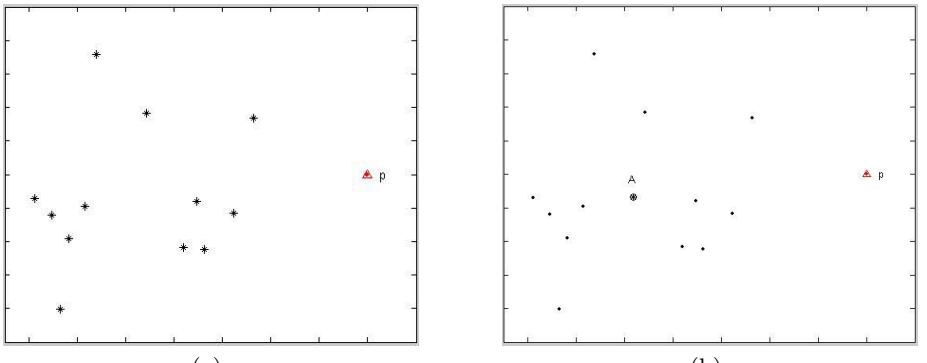
Definition 1 (KNN distance of x_p). *Let $\mathcal{N}_p$ be the set of the k-nearest neighbours of object x_p (excluding x_p). The k-nearest neighbours distance of x_p equals the average distance from x_p to all objects in $\mathcal{N}_p$. More formally, let $dist(x, x') \geq 0$ be a distance measure between objects x and x'. The k-nearest neighbours distance of object x_p is defined as*

$$\bar{d}_{x_p} := \frac{1}{k} \sum_{x_i \in \mathcal{N}_p} dist(x_i, x_p).$$



(a) 2-D projection of WDBC. (b) Synthetic 2-D data.

Fig. 1. (a) The 2-D projection of a real-world dataset (b) Simple 2-D illustration



(a) (b)

Fig. 2. (a) An anomalous object x_p with scattered neighbours. (b) The explicit outlierness of object x_p with the help of LDOF definition. A is the center of neighbourhood system of x_p. The dashed circle includes all neighbours of x_p. The solid circle is x_p's "reformed" neighbourhood region.

Definition 2 (KNN inner distance of x_p). *Given the k-nearest neighbours set $\mathcal{N}_p$ of object x_p, the k-nearest neighbours inner distance of x_p is defined as the average distance among objects in $\mathcal{N}_p$:*

$$\bar{D}_{x_p} := \frac{1}{k(k-1)} \sum_{x_i, x_{i'} \in \mathcal{N}_p, i \neq i'} dist(x_i, x_{i'}).$$

Definition 3 (LDOF of x_p). *The local distance-based outlier factor of x_p is defined as:*

$$LDOF_k(x_p) := \frac{\bar{d}_{x_p}}{\bar{D}_{x_p}}$$

If we regard the k-nearest neighbours as a neighbourhood system, $LDOF$ captures the degree to which object x_p deviates from its neighbourhood system. It

has the clear intuitive meaning that $LDOF$ is the distance ratio indicating how far the object x_p lies outside its neighbourhood system. When $LDOF \lesssim 1$, it means that x_p is surrounded by a data 'cloud'. On the contrary, when $LDOF \gg 1$, x_p is outside the whole 'cloud'. It is easy to see that the higher $LDOF$ is, the farther x_p is away from its neighbourhood system.

To further explain our definition, we exemplify it in Euclidian space. Hereinafter, let $x_i \in \mathcal{X} = I\!R^d$, and $\bar{x} := \frac{1}{k}\sum_{x_i \in \mathcal{N}_p} x_i$. For the squared Euclidian distance $||\cdot||^2$, the outlier definition can be written as:

$$\bar{d}_{x_p} = \frac{1}{k}\sum_{x_i \in \mathcal{N}_p} ||x_p - x_i||^2 \;=\; ||x_p - \bar{x}||^2 + \frac{1}{k}\sum_{x_i \in \mathcal{N}_p} ||x_i - \bar{x}||^2, \tag{1}$$

$$\bar{D}_{x_p} = \frac{1}{k(k-1)}\sum_{x_i, x_{i'} \in \mathcal{N}_p, i \neq i'} ||x_i - x_{i'}||^2 \;=\; \frac{2}{k-1}\sum_{x_i \in \mathcal{N}_p} ||x_i - \bar{x}||^2. \tag{2}$$

Thus, $LDOF_k(x_p) \gg 1$, i.e. x_p lies outside its neighbourhood system, iff

$$||x_p - \bar{x}||^2 \;\gg\; \frac{k+1}{k(k-1)}\sum_{x_i \in \mathcal{N}_p} ||x_i - \bar{x}||^2. \tag{3}$$

The same expression holds for the more general Mahalanobis distance [MKB79]. In Equation 3, the lefthand-side is the square distance of x_p to its neighbourhood centroid $\bar{x}$, and the righthand-side becomes the distance variance in $\mathcal{N}_p$ when $k \gg 1$. Therefore, Equation 6 can be understood as follows: The k-nearest neighbours of object x_p form a "reformed" neighbourhood region, represented as a hyperball with radius $\bar{D}_{x_p}$, centered at $\bar{x}$. As illustrated in Figure 2(a), since the neighbours of x_p are scattered, it is unclear whether x_p (indicated by $\triangle$) belongs to its neighbourhood system or not. Our $LDOF$ definition, as shown in Figure 2(b), it clearly regards x_p as lying outside its reformed neighbourhood region. The $LDOF$ of x_p is obviously greater than 1, which indicates that x_p is an outlier. Through this example we can see that $LDOF$ can effectively capture the outlier-ness of an object among a scattered neighbourhood. In addition, as k grows, $LDOF$ takes more objects into consideration, and the view of $LDOF$ becomes increasingly global. If an object is far from its large neighbourhood system (extremely the whole dataset) it is definitely a genuine outlier. Hence, the detection precision of our method might be stable over a large range of k. In the following section, we will theoretically analyse properties of $LDOF$, and propose a heuristic for selecting the neighbourhood size k.

4 Properties of LDOF

Lower bound of *LDOF*. Ideally, we prefer a universal threshold of $LDOF$ to unambiguously distinguish abnormal from normal objects (e.g. in any datasets, an object is outlier if $LDOF > 1$). However, the threshold is problem dependent due to the complex structure of real-world datasets. Under some continuity assumption, we can calculate an asymptotic lower bound on $LDOF$, denoted as

$LDOF_{lb}$. $LDOF_{lb}$ indicates that an object is an inlier (or normal) if its $LDOF$ is smaller than $LDOF_{lb}$.

Theorem 1 ($LDOF$ lower-bound of outliers). *Let data $\mathcal{D}$ be sampled from a density that is continuous at x_p. For $N \gg k \gg 1$ we have $LDOF_{lb} \approx \frac{1}{2}$ with high probability. More formally, for $k, N \to \infty$ such that the neighbourhood size $\bar{D}_{x_p} \to 0$ we have*

$$LDOF_{lb} = \frac{\bar{d}_{x_p}}{\bar{D}_{x_p}} \to \frac{1}{2} \quad \textit{with probability 1}$$

The theorem shows that when $LDOF \approx \frac{1}{2}$, the point is squarely lying in a uniform cloud of objects, i.e. it is not an outlier. The lower-bound of $LDOF$ provides a potential pruning rule of algorithm complexity. In practice, objects can be directly ignored if their $LDOF$s are smaller than $\frac{1}{2}$. Remarkably, $LDOF_{lb}$ does not depend on the dimension of $\mathcal{X}$. This is very convenient: data often lie on lower-dimensional manifolds. Since locally, a manifold is close to an Euclidian space (of lower dimension), the result still holds in this case. Therefore, we do not need to know the effective dimension of our data.

False-detection probability. As discussed in Section 1, in real-world datasets, it is hard to set parameters properly by trial-and-error. Instead of requiring prior knowledge from datasets (e.g. outlier labels), we theoretically determine the false-detection probability, given neighbourhood size k.

Theorem 2 (False-detection probability of LDOF). *Let data $\mathcal{D}$ be uniformly distributed in a neighbourhood of x_p containing k objects $\mathcal{N}_p$. For $LDOF$ threshold $c > \frac{1}{2}$, the probability of false detecting $x_p \in \mathbb{R}^d$ as an outlier is exponentially small in k. More precisely,*

$$\mathrm{P}[LDOF_k(x_p) > c] \;<\; \mathrm{e}^{-\alpha(k-2)}, \quad \textit{where} \quad \alpha := \tfrac{2}{25}(1-\tfrac{1}{2c})^2(\tfrac{d}{d+2})^2$$

The bound still holds for non-uniform densities continuous in x_p, provided $N \gg k$.

In particular, for $c = 1$ in high-dimensional spaces ($d \to \infty$) we get $\alpha \to \frac{1}{50}$. So for $k \gg 50$ the false-detection probability is very small. Note that because the bound is quite crude, we can expect good performance in practice for much smaller k. On the other hand, choosing $c \approx \frac{1}{2}$ degenerates the bound (i.e. $\alpha \to 0$), consistent with Theorem 1.

Due to space limitation, we omit the proofs of Theorems 1 and 2.

5 LDOF Outlier Detection Algorithm and Its Complexity

Top-n LDOF. Even with the theoretical analysis of the previous section, it is still hard to determine a threshold for $LDOF$ to identify outliers in an arbitrary dataset. Therefore we employ top-n style outlier detection, which ranks the n objects with the highest $LDOF$s. The algorithm that obtains the top-n $LDOF$ outliers for all the N objects in a given dataset $\mathcal{D}$ is outlined in Algorithm 1.

Algorithm 1. Top-n LDOF (Top-n Local Distance-based Outlier Factor)

Input: A given dataset $\mathcal{D}$, natural numbers n and k.

1. For each object p in $\mathcal{D}$, retrieve p's k-nearest neighbours;
2. Calculate the $LDOF$ for each object p.
   The objects with $LDOF < LDOF_{lb}$ are directly discarded;
3. Sort the objects according to their $LDOF$ values;
4. **Output:** the first n objects with the highest $LDOF$ values.

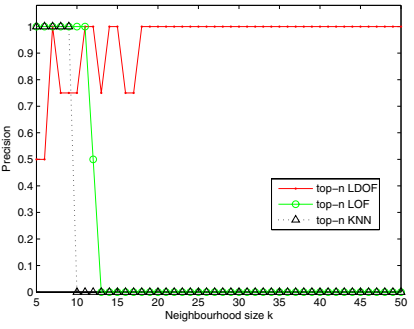
How to choose *k*. Based on Theorem 2, it is beneficial to use a large neighbourhood size k. However, too large k will lead to a global method with the same problems as top-n KNN outlier. For the best use of our algorithm, the lower bound of potentially suitable k is given as follows: If the effective dimension of the manifold on which $\mathcal{D}$ lies is m, then at least m points are needed to 'surround' another object. That is to say a $k > m$ is needed. In Section 6, we will see that, when k increases to the dimension of the dataset, the detection performance of our method rises, and remains stable for a wide range of k values. Therefore, the parameter k in LDOF is easier to choose than in other outlier detection approaches.

Algorithm complexity. In Step 1, querying the k-nearest neighbours, takes the majority of the computational load. Naively, the runtime of this step is $O(N^2)$. If a tree-based spatial index such as X-tree or R^*-tree is used [BKNS00, BKNS99], the complexity is reduced to $O(N \log N)$. Step 2 is straightforward and calculates $LDOF$ values according to Definition 3. As the k-nn query is materialised, this step is linear in N. Step 3 sorts the N objects according to their $LDOF$ values, which can be done in $O(N \log N)$. Since the objects with $LDOF < LDOF_{lb}$ are flushed (i.e. they are definitely non-outliers), the number of objects needed to sort in this step is smaller than N in practice. Finally, the overall computation complexity of Algorithm 1 is $O(N \log N)$ with appropriate index support.

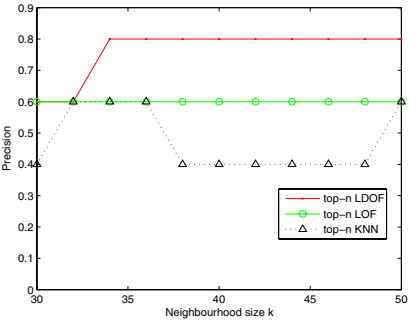
6 Experiments

In this section, we compare the outlier detection performance of top-n LDOF with two typical top-n outlier detection methods, top-n KNN and top-n LOF. Experiments start with a synthetic 2-D dataset which contains outliers that are meaningful but are difficult for top-n KNN and top-n LOF. In Experiments 2 and 3, we identify outliers in two real-world datasets to illustrate the effectiveness of our method in real-world situations. For consistency, we only use the parameter k to represent the neighbourhood size in the investigation of the three methods. In particular, in top-n LOF, the parameter *MinPts* is set to neighbourhood size k as chosen in the other two methods.

Synthetic Data. In Figure 1(b), there are 150 objects in cluster C_1, 50 objects in cluster C_2, 10 objects in cluster C_3, and 4 additional objects $\{o_1, o_2, o_3, o_4\}$



(a) Precisions in synthetical dataset.



(b) Precisions in WDBC dataset.

Fig. 3. Detecting precisions of top-n LDOF, top-n KNN and top-n LOF on (a) Synthetical dataset, (b) WDBC dataset

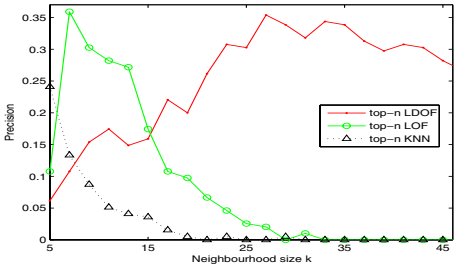
which are genuine outliers. We ran the three outlier detection methods over a large range of k. We use detection precision[2] to evaluate the performance of each method. In this experiment, we set $n = 4$ (the number of real outliers). The experimental result is shown in Figure 3(a). The precision of top-n KNN becomes 0 when the k is larger than 10 due to the effect of the mini-cluster C_3 as we discussed in Section 2. For the same reason, the precision of top-n LOF dramatically descends when k is larger than 11. When the k reaches 13, top-n LOF misses all genuine outliers in the top-4 ranking (they even drop out of top-10). On the contrary, our method is not suffering from the effect of the mini-cluster. As shown in the Figure 3(a), the precision of our approach keeps stable at 100% accuracy over a large neighbourhood size range (i.e. 20-50).

Medical Diagnosis Data. In real-world data repositories, it is hard to find a dataset for evaluating outlier detection algorithms, because only for very few real-world datasets it is exactly known which objects are really behaving differently [KSZ08]. In this experiment, we use a medical dataset, WDBC (Diagnosis)[1], which has been used for nuclear feature extraction for breast tumor diagnosis. The dataset contains 569 medical diagnosis records (objects), each with 32 attributes (ID, diagnosis, 30 real-valued input features). The diagnosis is binary: 'Benign' and 'Malignant'. We regard the objects labeled 'Benign' as normal data. In the experiment we use all 357 'Benign' diagnosis records as normal objects and add a certain number of 'Malignant' diagnosis records into normal objects as outliers. Figure 3(b) shows the experimental result for adding the first 10 'Malignant' records from the original dataset. Based on the rule for selecting neighbourhood size, k, suggested in Section 4, we set $k \geq 30$ in regards to the data dimension. We measure the percentage of real outliers detected in top-10 potential outliers as detection precision[2]. In the experiments, we progressively increase the value of k and calculate the detection precision for each

[2] Precision= $n_{\text{real-outliers in top-n}}/n$. We set n as the number of real outliers if possible.

Table 1. The detecting precision for each method based on 30 independent runs

Number of outliers	Precision (mean ± std.)		
	LDOF	LOF	KNN
1	**0.29**±0.077	0.12±0.061	0.05±0.042
2	**0.33**±0.040	0.13±0.028	0.11±0.037
3	**0.31**±0.033	0.22±0.051	0.22±0.040
4	0.35±0.022	0.27±0.040	0.26±0.035
5	**0.38**±0.026	0.28±0.032	0.28±0.027



Precision (mean ± std.)		
LDOF	LOF	KNN
0.25±0.081	0.03±0.057	0.08±0.114

Fig. 4. & Tab.4. Outlier detection precision over different neighbourhood size for Shuttle dataset based on 15 independent runs

method. As shown in Figure 3(b), the precision of our method begins to ascend at $k = 32$, and keeps stable when k is greater than 34 with detection accuracy of 80%. In comparison, the precision of the other two techniques are towed over the whole k value range.

To further validate our approach, we repeat the experiment 5 times with a different number of outliers (randomly extracted from 'Malignant' objects). Each time, we perform 30 independent runs, and calculate the average detection precision and standard deviation over the k range from 30 to 50. The experimental results are listed in Table 1. The bold numbers indicate that the detection precision vector over the range of k is statistically significantly improved compared to the other two methods (paired T-test at the 0.1 level).

Space Shuttle Data. In this experiment, we use a dataset originally used for classification, named Shuttle[3]. We use the testing dataset which contains 14500 objects, and each object has 9 real-valued features and an integer label (1-7). We regard the (only 13) objects with label 2 as outliers, and regard the rest of the six classes as normal data. We run the experiment 15 times and each time we randomly pick a sample of normal objects (i.e. 1,000 objects) to mix with the 13 outliers. The mean values of detection precision of the three methods are presented in Figure 4. As illustrated in Figure 4, top-n KNN has the worst performance (rapidly drops to 0). Top-n LOF is better, which has a narrow precision peak (k from 5 to 15), and then declines dramatically. Top-n LDOF has the best performance, as it ascends steadily and keeps a relative high precision over the k range from 25 to 45. Table 4 shows the average precisions for the three methods over 15 runs. The bold numbers indicate that the precision

[3] The Shuttle dataset can also be downloaded from UCI ML Repository.

vector is statistically significantly improved compared to the other two methods (paired T-test at the 0.1 level).

7 Conclusion

In this paper, we have proposed a new outlier detection definition, LDOF. Our definition uses a local distance-based outlier factor to measure the degree to which an object deviates from its scattered neighbourhood. We have analysed the properties of LDOF, including its lower bound and false-detection probability. Furthermore, a method for selecting k has been suggested. In order to ease the parameter setting in real-world applications, the top-n technique has been used in this approach. Experimental results have demonstrated the ability of our new approach to better discover outliers with high precision, and to remain stable over a large range of neighbourhood sizes, compared to top-n KNN and top-n LOF. As future work, we are looking to extend the proposed approach to further enhance the outlier detection accuracy for scattered real-world datasets.

References

[Bar94] Barnett, V.: Outliers in Statistical Data. John Wiley, Chichester (1994)

[BKNS99] Breunig, M.M., Kriegel, H.-P., Ng, R.T., Sander, J.: OPTICS-OF: Identifying local outliers. In: Żytkow, J.M., Rauch, J. (eds.) PKDD 1999. LNCS, vol. 1704, pp. 262–270. Springer, Heidelberg (1999)

[BKNS00] Breunig, M.M., Kriegel, H.-P., Ng, R.T., Sander, J.: LOF: Identifying density-based local outliers. In: SIGMOD Conference, pp. 93–104 (2000)

[EKSX96] Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A density- based algorithm for discovering clusters in large spatial databases with noise. In: KDD, pp. 226–231 (1996)

[FZFW06] Fan, H., Zaïane, O.R., Foss, A., Wu, J.: A non- parametric outlier detection for effectively discovering top-n outliers from engineering data. In: Ng, W.-K., Kitsuregawa, M., Li, J., Chang, K. (eds.) PAKDD 2006. LNCS, vol. 3918, pp. 557–566. Springer, Heidelberg (2006)

[Haw80] Hawkins, D.: Identification of Outliers. Chapman and Hall, London (1980)

[KN98] Knorr, E.M., Ng, R.T.: Algorithms for mining distance-based outliers in large datasets. In: VLDB, pp. 392–403 (1998)

[KSZ08] Kriegel, H.-P., Schubert, M., Zimek, A.: Angle-based outlier detection in high-dimensional data. In: KDD, pp. 444–452 (2008)

[MKB79] Mardia, K.V., Kent, J.T., Bibby, J.M.: Multivariate analysis. Academic Press, New York (1979)

[RRS00] Ramaswamy, S., Rastogi, R., Shim, K.: Efficient algo- rithms for mining outliers from large data sets. In: SIGMOD Conference, pp. 427–438 (2000)

[TCFC02] Tang, J., Chen, Z., Fu, A.W.-C., Cheung, D.W.-L.: Enhancing effectiveness of outlier detections for low density patterns. In: Chen, M.-S., Yu, P.S., Liu, B. (eds.) PAKDD 2002. LNCS, vol. 2336, pp. 535–548. Springer, Heidelberg (2002)

[Tuk77] Tukey, J.W.: Exploratory Data Analysis. Addison-Wiley, Chichester (1977)

Mining Outliers with Faster Cutoff Update and Space Utilization

Chi-Cheong Szeto and Edward Hung

Department of Computing, Hong Kong Polytechnic University
{csccszeto,csehung}@comp.polyu.edu.hk

Abstract. It is desirable to find unusual data objects by Ramaswamy et al's distance-based outlier definition because only a metric distance function between two objects is required. It does not need any neighborhood distance threshold required by many existing algorithms based on the definition of Knorr and Ng. Bay and Schwabacher proposed an efficient algorithm ORCA, which can give near linear time performance, for this task. To further reduce the running time, we propose in this paper two algorithms RC and RS using the following two techniques respectively: (i) faster cutoff update, and (ii) space utilization after pruning. We tested RC, RS and RCS (a hybrid approach combining both RC and RS) on several large and high-dimensional real data sets with millions of objects. The experiments show that the speed of RCS is as fast as 1.4 to 2.3 times that of ORCA, and the improvement of RCS is relatively insensitive to the increase in the data size.

1 Introduction

Hawkins [1] defines an outlier as "an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism". Outlier detection has many applications such as detecting malicious activities in computer systems and criminal activities in commercial organizations.

Knorr and Ng [2] first proposed a distance-based outlier definition that "an object O in a data set T is an outlier if at least fraction p of the objects in T lies greater than distance D from O." Distance-based outliers generalize certain statistical outlier tests. As it is not hard to define a metric distance function, it readily applies to numerous applications. Knorr and Ng proposed the simple Nested-Loop algorithm (NL) [2] to find outliers by calculating their distances to all other objects or until an object can be confirmed as a non-outlier. Its complexity is $O(kN^2)$ where k is the number of dimensions and N is the number of objects in the data set.

Even though efficient methods ([3] and [4]) have been developed, in practice, it is non-trivial to define an appropriate value for D. Knorr and Ng commented that it may take numerous iterations of trial and error to find D. In [3], an interesting range $[D_{min},D_{max}]$ of D was found by an expensive brute-force method.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 823–830, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Later, Ramaswamy et al. [5] proposed a closely related definition without the need to set D: "given integers k and n, an object O is an outlier if it is one of the n objects with the largest distances to their kth nearest neighbors." Users can simply specify how many outliers they want the algorithm to return without worrying about distance D. Moreover, the outliers returned are ranked and easier for users to examine. In fact, methods to detect such outliers can also be readily applied to the definition of Knorr and Ng by substituting $k = (1-p)|T|$, letting n be a very large value and initializing a cutoff distance, which is assumed to be unknown, to the value of given D. Therefore, our work focuses on the definition of Ramaswamy et al.

Efficient methods ([6] and [7]) were proposed to detect such outliers, where parameters are easier to specify without knowing the value of D. In [6], Bay and Schwabacher proposed ORCA, which extends NL by employing a pre-processing step of randomizing the data set. ORCA is shown to give near linear time performance on real, large and high-dimensional data sets. In [7], Ghoting et al. presented RBRP (Recursive Binning and Re-Projection), which extends NL by employing a pre-processing step of data clustering. In addition to the enhancement of [7], we observed the following two other areas of inefficiency of ORCA. We will discuss how to improve them to further increase the time saving.

1. ORCA prunes away an object if the object's distance to its (so far) kth nearest neighbor (which is decreasing during processing) is smaller than a cutoff value. The cutoff value is defined as the kth-nearest-neighbor-distance of the top nth outlier (found so far), which is initialized to zero at the start. As more objects are processed and better outliers are found, the cutoff value increases gradually to its final value and the pruning efficiency also increases. Thus, in the beginning, the computational cost is very high due to low pruning efficiency. For a real data set with 48,842 objects, the time spent to process the first thousand objects (2% of the data set) occupies 54% and 32% of total running time when n (the number of outliers to find) is 0.01% and 0.1% of the data set respectively.
2. The memory space is under-utilized leading to an increase in disk I/O cost. For a real data set with 1,000,000 objects, on average over 98% of memory space is available again by pruning confirmed non-outliers after scanning 10% of the data set. At the start, each object read from the disk is compared with b objects in the memory buffer. After scanning 10% of the data set, each object read from the disk is compared with $0.02b$ objects. The disk I/O cost per object increases by 4900%! Moreover, this under-utilization lasts until the end of the data scan. For the same data set, a data scan ends after scanning on average 45.8% and 93.5% of the data set when n is 0.01% and 0.1% of the data set respectively. Therefore, the under-utilization happens in more than 78% of the disk I/O time.

Therefore, to solve the above two problems, we propose two techniques: (i) faster cutoff update, and (ii) space utilization after pruning. Both techniques complement each other in reducing execution time. The resulting algorithms are

called RC and RS respectively. Details will be described in Sect. 3. In Sect. 4, we will discuss the experiments done on RC, RS and RCS (a hybrid approach combining both techniques). We will first describe the working principle of the ORCA algorithm [6] in Sect. 2.

2 The ORCA Algorithm

The ORCA algorithm calculates for each data object a score which can be defined as either (i) the average distance to the object's k nearest neighbors or (ii) the distance to the object's kth nearest neighbor. A higher score means that the object is further away from the other objects and hence is more dissimilar to the others. The top n outliers are defined as objects with the top n scores.

The algorithm (see Fig. 1) consists of three main components: (1) nested loops, (2) randomization and (3) a pruning rule. The nested loop computes the distances of an object to all other objects to find the object's k nearest neighbors. The randomization refers to randomizing the order of the objects in the data set so that an object drawn sequentially from the data set will have the same chance to be a nearby neighbor of any object. The nth top score from the objects processed so far is called the cutoff value. Any object with a score smaller than the cutoff value cannot be the top n outliers and we can stop computing its distance to other objects. This condition acts as a pruning rule to reduce the number of distance computations.

```
1   Shuffle the objects in the data set D; initialize the cutoff value to be zero.
2   While there are objects in D unprocessed do
3       Load a batch of unprocessed objects of a fixed regular size to the memory buffer B.
4       For each object d in D
5           For each object b in B
6               Compute the distance between b and d. If d is closer to b than the objects kept
                previously as the k nearest neighbors of b, update the nearest neighbors of b with d and
                compute a new score of b. If the score is lower than the cutoff value, prune b.
7           End for
8       End for
9       At the end of a data scan, remove the objects remaining in B. Rank them together with the top n
        outliers previously kept. Keep the new top n outliers and update the cutoff value.
10  End while
```

Fig. 1. Algorithm of ORCA

3 Our Proposed Algorithms RC, RS and RCS

3.1 Randomization with Faster Cutoff Update (RC)

Our first technique solves the first problem described in Sect. 1 and is applied in the early processing stage. We define the early processing stage to be from the start of processing to the time when the ratio of new to old cutoff values is smaller than a threshold, that is, when the cutoff value no longer increases

quickly. In our experiment, we set the threshold to be $1.1^{b/100}$ for a start batch size of b.

There are two characteristics in this stage. Firstly, there is more room to reduce the computational cost because the cutoff value is small. Secondly, the computational cost decreases quickly in consecutive batch because better top n outliers are frequently found and so a larger cutoff value is found frequently. To save the computational cost, we take advantage of the rapid increase in cutoff value and propose using a smaller batch size in this stage. Using a smaller batch size causes a larger cutoff value to be used earlier and so the computational cost can be lowered. However, using a smaller batch size will also increase the disk I/O cost because processing each batch requires a disk scan of the data set.

The smaller batch size used in this stage is referred as the start batch size and is set to be $4n$ or 100, whichever is larger. The reason is that after processing the first batch of objects, the temporary top n outliers will be a quarter of objects (or less) with the highest scores. The cutoff value will be at least at the upper quartile of the scores of this first batch, which would be a good initial guess. If the batch size is too small, the initial guess of the cutoff value would be subjected to severe random variations. Hence, the start batch size is set to be at least 100 objects.

We call the algorithm with this technique RC, which replaces Step 3 of the ORCA algorithm (in Fig. 1) by the following STEP:

– If it is in the early processing stage, load a batch of unprocessed objects of a specified size (e.g. max($4n$,100)) to the memory buffer **B**; Otherwise, load a batch of unprocessed objects of a regular size instead.

3.2 Randomization with Space Utilization After Pruning (RS)

Our second technique solves the second problem described in Sect. 1 and is applied in the main processing stage. We define the main processing stage to be from the time that a specified percentage of the data set is processed to the end of processing. In our experiment, we set the percentage to be 10%.

There are two characteristics in this stage. Firstly, there is little room to reduce the computational cost because the cutoff value starts getting close to its final value. Secondly, the computational cost decreases slowly in consecutive batch because better top n outliers are infrequent to find and so a larger cutoff value is found infrequently.

To reduce the total running time, we could increase the batch size so that the number of data scans and the disk I/O time can be reduced at the expense of small increase in computational time. However, this approach increases the memory requirement. Instead, we utilize the space of pruned objects and fill it with unprocessed objects. Then, more objects can be processed during one data scan. Thus, the number of data scans and the disk I/O time decrease. The frequency of filling the space controls the number of objects processed in one data scan. In our experiment, the frequency is set to be every 10% of a complete data scan.

```
1  Shuffle the objects in the data set D; initialize the cutoff value to be zero.
2  Load the first batch of unprocessed objects to the memory buffer B.
3  While there are objects in D unprocessed do
4      For each object d in D
5          For each object b in B
6              Compute the distance between objects b and d. If d is closer to b than the objects kept
               previously as the k nearest neighbors of b, update the nearest neighbors of b with d and
               compute a new score of b. If the score is lower than the cutoff value, prune b.
7          End for
8          If it is at the end of every fixed percentage f of a complete data scan (e.g., f = 25%).
9              Remove the objects in the memory buffer that have been compared with all objects in
               D. Rank them together with the top n outliers previously kept. Keep the new top n
               outliers and update the cutoff value.
10             Prune any object in the memory buffer with a score lower than the updated cutoff value.
11             Load unprocessed objects into the memory buffer to fill the available space up, if any.
12         End if
13     End for
14 End while
```

Fig. 2. Algorithm of RS

Figure 2 shows the algorithm of RS. In Step 6, objects with scores lower than the cutoff value are pruned. In Step 9, at the end of every fixed percentage of a complete data scan, objects that have been compared with all objects in the data set are removed and used to update the top n outliers and the cutoff value. In Step 10, objects with scores lower than the updated cutoff value are pruned. In Step 11, the space obtained from previous pruning steps is filled with new unprocessed objects.

3.3 Hybrid Approach

Randomization with faster Cutoff update and Space utilization after pruning (RCS), combining RC and RS, improves both the performance of the early and the main processing stages.

4 Experimental Study

In this section, we evaluate the time performance of our proposed algorithms (RC, RS and RCS) on several data sets (see the first column of Fig. 3) by comparing them with ORCA. We also study how the performance scales with the data size. The effects of parameters introduced with the proposed algorithms are studied at the end of this section. Our experiments are conducted on a lightly loaded Intel Core 2 Duo computer with 2 GHz processors and 2 GB memory.

4.1 Performance against Different Data Sets

Figure 3 compares the running time of ORCA, RC, RS and RCS on the four real data sets. The running time includes both CPU and disk I/O time but not the pre-processing and shuffling time. The continuous variables of the data sets are normalized to the range of [0,1] and the categorical variables are represented

Data set	Data size (number of objects)	Dimensions	Running time / s (percentage saved compared with ORCA)			
			ORCA	RC	RS	RCS
Outlier percentage (percentage of data as outliers) = 0.01%						
1a Census income [8]	48842	15	11.6	6.87(40.9%)	9.89(15.0%)	5.14(55.9%)
2a Corel image features [9]	68040	32	35.7	23.7(33.8%)	27.9(22.0%)	15.9(55.4%)
3a Forest cover type [9]	581012	55	2444	2378(2.68%)	1512(38.1%)	1455(40.5%)
4a KDD cup 1999[1] [9]	1000000	42	3397	3356(1.22%)	1750(48.5%)	1760(48.2%)
5a KDD cup 1999 [9]	4898431	42	71874	71874(0%)	39258(45.4%)	39258(45.4%)
Outlier percentage (percentage of data as outliers) = 0.1%						
1b Census income [8]	48842	15	19.4	16.1(17.0%)	14.8(23.8%)	11.4(41.0%)
2b Corel image features [9]	68040	32	62.1	54.5(12.2%)	47.9(22.9%)	40.1(35.4%)
3b Forest cover type [9]	581012	55	6494	6494(0%)	4718(27.3%)	4718(27.3%)
4b KDD cup 1999[1] [9]	1000000	42	9327	9327(0%)	4895(47.5%)	4895(47.5%)

[1]The size was reduced to one million objects due to long running time.

Fig. 3. Performance against different data sets

by integers. The parameter values used are the same as the previous study [6]: the number of nearest neighbors to consider, $k = 5$ and the regular batch size = 1000. Two outlier percentages, 0.01% and 0.1% are tested.

There are three points to note. Firstly, the speed of RCS, combining RC and RS, is as fast as 1.4 to 2.3 times that of ORCA. Secondly, the running time with the outlier percentage set to 0.1% is longer. The reason is that if the outlier percentage is higher, the cutoff value is smaller. More objects need to be compared with to find k neighbours within the smaller cutoff value. Hence, the running time is longer. Thirdly, in each of the outlier percentages, the percentages of time saved by RC shown in the fifth column have decreasing trends and the percentages saved by RS shown in the sixth column have an increasing trend. This is strongly related to the size of the data set and is studied in the next subsection. RC does not save any time in rows 5a, 3b and 4b because the start batch size exceeds the regular batch size. In these cases, the start batch size is set as the regular batch size and RC works as ORCA.

4.2 Varying the Data Size

In this subsection, we investigate how the performance of RC, RS and RCS is affected by the data set size. The data sets of different sizes were generated by truncating the shuffled data set "KDD cup 1999". Two outlier percentages, 0.01% and 0.1% are tested. Only the results with the outlier percentage set to 0.01% are included because of the page limit. However, the results of these two outlier percentages are similar.

Figure 4(a) shows the percentage of running time saved compared with ORCA. As the data size N increases, the time saving of RC decreases but the time saving of RS increases. The improvement of RCS, where RC and RS complement each other, is relatively insensitive to the increase in the data size.

The reason why the time saving of RC decreases with N is that the start batch size of RC equals $4n = 4\alpha N$ where n and α are the number and the percentage of outliers respectively. As N increases, the start batch size also increases. The

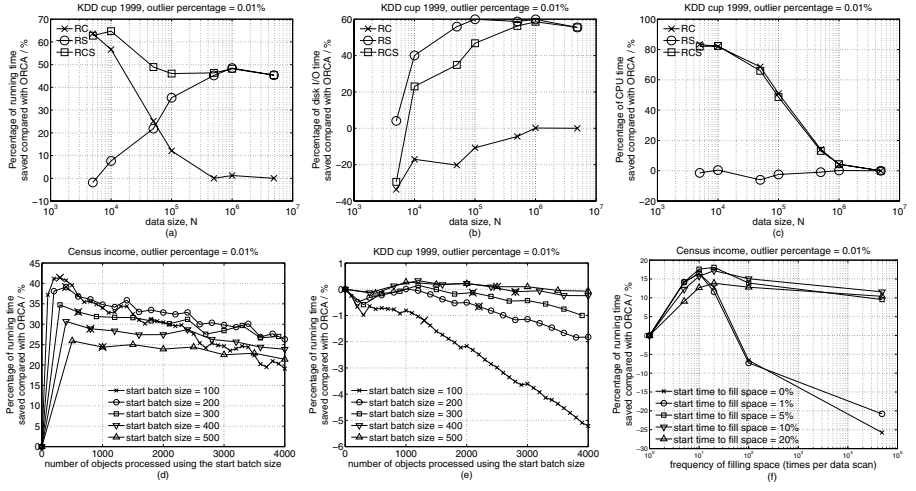


Fig. 4. Percentage of (a) running, (b) disk I/O and (c) CPU time saved against the data size; Percentage of running time saved (d), (e) by RC against the number of objects processed, and (f) by RS against the frequency of filling space.

difference between using a smaller start batch size and the original regular batch size decreases and so is the effect of RC.

To examine why RS improves more with more data, we decompose the total running time into CPU and disk I/O time. Fig. 4(b) shows that RS gives constant disk I/O time improvement over ORCA except the first two points of small data sizes. The disk I/O cost per object in the last data scan depends on the number of unprocessed objects remaining to process. The total numbers of data scans of the two small data sizes are 2 and 3, so the time improvement is seriously affected by the number of unprocessed objects remaining to process in the last data scan. RS cannot help reduce the disk I/O cost as it can refill no more unprocessed objects. Fig. 4(c) shows that RS consumes insignificantly small and constant extra CPU time over ORCA. Moreover, we observed that the ratio of the disk I/O cost to the computational cost increases with N. Hence, when N increases, the saving of the disk I/O cost becomes more significant to the overall time saving. As a result, the overall time saving of RS increases with N.

4.3 Effects of the Parameters of RC and RS

In this subsection, we study the effects of the start batch sizes and the ending time of RC (see Figs. 4(d) and (e)) as well as the start time and the frequency of space utilization of RS (see Fig. 4(f)) by varying their values. Here is the summary of the results. The effect of the start batch size shows no similar pattern between different data sets. The time saving does not change much with different ending time except that a very small start batch size like 100 is used. Using the ending time determined by RC (indicated by the big cross symbols in the figures),

RC saves at least 98% of the optimal time saving. The percentage of time saved may be lower if RS starts too early such as after processing 0% and 1% of the data set. It is also very sensitive to the frequency of filling the memory space. However, if RS starts later such as after processing 20% of the data set, the percentage of time saved is less sensitive to this frequency.

5 Conclusion

In this paper, we proposed two algorithms RC and RS using the following two techniques respectively: (i) faster cutoff update, and (ii) space utilization after pruning. Both techniques complement each other in reducing execution time. Our experimental results show that the speed of our RCS algorithm, which combines these two techniques, is as fast as 1.4 to 2.3 times that of ORCA. We investigated the saving in total running time, CPU time and disk I/O time of these two techniques against different data sets and data sizes. The experimental result indicates that the improvement of RCS is relatively insensitive to the increase in the data size.

Acknowledgement

This work has been partially supported by grant PolyU 5174/04E (B-Q05W) from Hong Kong RGC, A-PA5S and A-SA14 from PolyU.

References

1. Hawkins, D.M.: Identification of outliers. Chapman and Hall, Boca Raton (1980)
2. Knorr, E.M., Ng, R.T.: Algorithms for mining distance-based outliers in large datasets. In: VLDB 1998: Proceedings of the 24th International Conference on Very Large Data Bases, pp. 392–403. Morgan Kaumann Publishers Inc., San Francisco (1998)
3. Tao, Y., Xiao, X., Zhou, S.: Mining distance-based outliers from large databases in any metric space. In: KDD 2006: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 394–403. ACM, New York (2006)
4. Angiulli, F., Fassetti, F.: Very efficient mining of distance-based outliers. In: CIKM 2007: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, pp. 791–800. ACM, New York (2007)
5. Ramaswamy, S., Rastogi, R., Shim, K.: Efficient algorithms for mining outliers from large data sets. SIGMOD Rec. 29(2), 427–438 (2000)
6. Bay, S.D., Schwabacher, M.: Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In: KDD 2003: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 29–38. ACM, New York (2003)
7. Ghoting, A., Parthasarathy, S., Otey, M.E.: Fast mining of distance-based outliers in high-dimensional datasets. Data Min. Knowl. Discov. 16(3), 349–364 (2008)
8. Asuncion, A., Newman, D.: UCI machine learning repository (2007)
9. Hettich, S., Bay, S.D.: The UCI KDD archive (1999)

Outlier Detection in Axis-Parallel Subspaces of High Dimensional Data

Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek

Ludwig-Maximilians-Universität München
Oettingenstr. 67, 80538 München, Germany
{kriegel,kroegerp,schube,zimek}@dbs.ifi.lmu.de
http://www.dbs.ifi.lmu.de

Abstract. We propose an original outlier detection schema that detects outliers in varying subspaces of a high dimensional feature space. In particular, for each object in the data set, we explore the axis-parallel subspace spanned by its neighbors and determine how much the object deviates from the neighbors in this subspace. In our experiments, we show that our novel subspace outlier detection is superior to existing full-dimensional approaches and scales well to high dimensional databases.

1 Introduction

Outlier detection aims at finding the "different mechanism" [1], i.e., detecting outliers that do not fit well to the mechanisms that generate most of the data objects. All existing approaches somehow rely on the full-dimensional Euclidean data space in order to examine the properties of each data object to detect outliers. However, today's applications are characterized by producing high dimensional data. In general, mining these high dimensional data sets is imprecated with the *curse of dimensionality*. For outlier detection, two specific aspects are most important. First, in high dimensional spaces Euclidean distances (and other L_p-norms) can no longer be used to differentiate between points. All points are more or less equi-distant to each other (see e.g. [2]). As a consequence, no particular outlier can be detected that deviates considerably from the majority of points. Second, we may have still concrete mechanisms that have generated the data but, usually, for each of these generating mechanisms only a subset of features may be relevant (this problem is known as *local feature relevance* [3]). In addition, these subsets of relevant features may be different for different generating mechanisms. As a consequence, outlier detection makes sense only when considering the subsets of relevant features of these generating mechanisms, i.e. subspaces of the original feature space. Figure 1(a) illustrates the general idea of finding outliers in subspaces. Point o is not an outlier in the full (two) dimensional space because it does not deviate considerably from its neighboring points (indicated by crosses). Since the density among o and its neighbors in the two dimensional feature space is rather uniform, o will also not be recognized as an outlier by any existing full dimensional outlier detection method. However, when projected on the axis A_1, point o is an outlier because it deviates

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 831–838, 2009.
© Springer-Verlag Berlin Heidelberg 2009

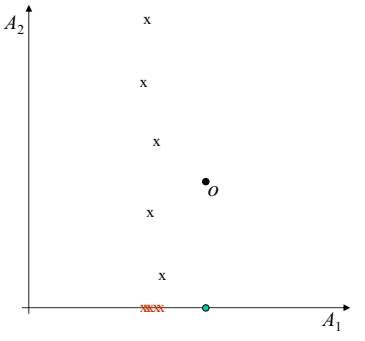
considerably from the neighboring points. Apparently, the points indicated by crosses have been generated by a mechanism where a low variance around a certain value in attribute A_1 is characteristic while the values of attribute A_2 are uniformly distributed and obviously not characteristic for the given mechanism. Finding outliers in subspaces is particularly interesting in high dimensional data where we can expect a rather uniform distribution in the full dimensional space but interesting distributions (including outliers) in subspaces. Since these subspaces of relevant features are usually not known beforehand (outlier detection is an unsupervised task), the search for outliers must be coupled with the search for the relevant subspaces. In this paper, we present a novel outlier detection schema that searches for outliers in subspaces of the original data. Our method is particularly useful for high dimensional data where outliers cannot be found in the entire feature space but in different subspaces of the original space. The remainder is organized as follows. We review related work in Section 2. Our novel subspace outlier model is described in Section 3. An experimental evaluation is presented in Section 4. Section 5 provides conclusions.

2 Related Work

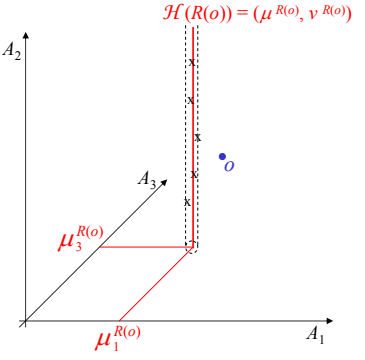
Existing approaches for outlier detection can be classified as global or local outlier models. A global outlier approach is based on differences of properties compared over the complete data set and usually models outlierness as a binary property: for each object it is decided whether it is an outlier or not. A local outlier approach rather considers a selection of the data set and usually computes a degree of outlierness: for each object a value is computed that specifies "how much" this object is an outlier w.r.t. the selected part of the data. Here, we focus on this second family of approaches. The first approach to overcome the limitations of a global view on outlierness has been the density-based local outlier factor (LOF) [4]. The LOF compares the density of each object o of a data set $\mathcal{D}$ with the density of the k-nearest neighbors of o. A LOF value of approximately 1 indicates that the corresponding object is located within a cluster, i.e. a region of homogeneous density. The higher the difference of the density around o is compared to the density around the k-nearest neighbors of o, the higher is the LOF value that is assigned to o. The outlier score ABOD [5] claims to be tailored to meet the difficulties in high dimensional data because it is not primarily based on conventional distance measures but assesses the variance in angles between an outlier candidate and all other pairs of points. Nevertheless, the special problem of irrelevant attributes in high dimensional data is not addressed by ABOD.

3 Outlier Detection in Axis-Parallel Subspaces

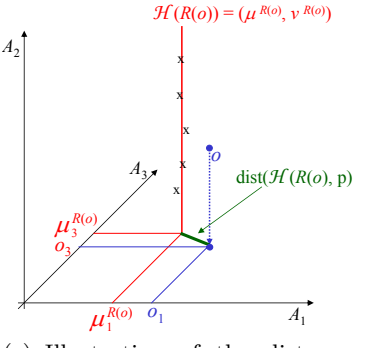
The general idea of our novel subspace outlier model is to analyze for each point, how well it fits to the subspace that is spanned by a set of reference points. The subspace spanned by a set of points is simply an axis-parallel hyperplane of any dimensionality $l < d$, where d is the dimensionality of the entire feature space,



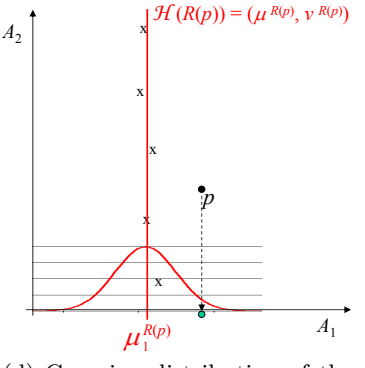
(a) The general idea of finding outliers in subspaces.



(b) Illustration of the subspace hyperplane of a reference set $R(o)$.



(c) Illustration of the distance between a point o and a subspace hyperplane $\mathcal{H}(R(o))$.



(d) Gaussian distribution of the distances of all points in $R(p)$ to $\mathcal{H}(R(p))$.

Fig. 1. Illustration of basic concepts.

such that all points of the reference set are close to this hyperplane. If a point deviates significantly from this reference hyperplane, it is considered to be an outlier in the subspace that is perpendicular to that hyperplane.

In the following, we assume that $\mathcal{D} \subseteq \mathbb{R}^d$ is a database of n points in a d-dimensional feature space and *dist* is a metric distance function on the points in $\mathcal{D}$, e.g. one of the L_p-norms or the cosine distance. For any point $p \in \mathbb{R}^d$, we denote the projection of p onto attribute i by p_i.

Intuitively, the subspace hyperplane of a set of points S (the reference set) captures the subspace in which the variance of the points in S is high, whereas in the perpendicular subspace, the variance of the points in S is low. The *variance* $VAR^S \in \mathbb{R}$ of S is the average squared distance of the points in S to the mean value μ^S, i.e., $VAR^S = \frac{\sum_{p \in S} dist(p,\mu^S)^2}{Card(S)}$, where $Card(S)$ denotes the cardinality

of the set S. Analogously, the *variance along an attribute* i, denoted by $var_i^S \in \mathbb{R}$ of S is defined as $var_i^S = \frac{\sum_{p \in S} \left(dist\left(p_i, \mu_i^S\right)\right)^2}{Card(S)}$.

Let $R(p) \subseteq \mathcal{D}$ be a set of reference points for $p \in \mathcal{D}$, called *reference set* w.r.t. which the outlierness of p should be evaluated. The *subspace defining vector* $v^{R(p)} \in \mathbb{R}^d$ of a reference set $R(p)$ specifies the relevant attributes of the subspace defined by the set $R(p)$, i.e. the attributes where the points in $R(p)$ exhibit a low variance. Thereby, we differentiate between high and low variance as follows. In all d attributes, the points have a total variance of $VAR^{R(p)}$. Thus, the expected variance along the i-th attribute is $1/d \cdot VAR^{R(p)}$. We evaluate the variance of the points along the i-th attribute as *low* if $var_i^{R(p)}$ is smaller than the expected variance by a predefined coefficient α. For each attribute in which $R(p)$ exhibits a low variance, the corresponding value of the subspace defining vector $v^{R(p)}$ is set to 1, for the remaining attributes to 0. Formally,

$$v_i^{R(p)} = \begin{cases} 1 & \text{if} \quad var_i^{R(p)} < \alpha \frac{VAR^{R(p)}}{d} \\ 0 & \text{else.} \end{cases} \tag{1}$$

The *subspace hyperplane* $\mathcal{H}(R(p))$ of $R(p)$ is defined by a tuple of the mean value $\mu^{R(p)}$ of $R(p)$ and the subspace defining vector $v^{R(p)}$ of $R(p)$, i.e. $\mathcal{H}(R(p)) = (\mu^{R(p)}, v^{R(p)})$. Figure 1(b) illustrates a subspace hyperplane for a sample reference set $R(o)$ (indicated by crosses) of a point o (indicated by a dot) in a three dimensional feature space. The points of $R(o)$ form a line in the three dimensional space. Thus, the subspace defining vector of $R(o)$ is defined as $v^{R(o)} = (1, 0, 1)^\intercal$, because attribute A_1 and A_3 are relevant and attribute A_2 is not, i.e. the variance along A_1 and A_3 is small whereas it is high along A_2. The subspace hyperplane of $R(o)$ is defined by the mean $\mu^{R(o)}$ of $R(o)$ and $v^{R(o)}$ and is visualized as the red solid line perpendicular to the plane spanned by A_1 and A_3. Now, we are able to measure how much p deviates from the subspace hyperplane $\mathcal{H}(R(p))$ spanned by its reference set $R(p)$. The deviation of any point o to a subspace hyperplane $\mathcal{H}(S)$ is thereby naturally defined as the Euclidean distance in the subspace which is perpendicular to the hyperplane. This can simply be computed using a weighted Euclidean distance between o and μ^S using the subspace defining vector v^S as weight vector, i.e.,

$$dist(o, \mathcal{H}(S)) = \sqrt{\sum_{i=1}^{d} v_i^S \cdot (o_i, \mu_i^S)^2}. \tag{2}$$

The idea of this distance between a sample 3D point o and the subspace hyperplane of its reference set $R(o)$ is illustrated in Figure 1(c). This distance value is a very intuitive measurement for the degree of outlierness of any $p \in \mathcal{D}$ w.r.t. the set of points in $R(p)$. A value near 0 indicates that the particular point p fits very well to the hyperplane $\mathcal{H}(R(p))$, i.e., is no outlier, whereas a considerably higher value indicates that p is an outlier. The final subspace outlier degree is defined as follows.

Definition 1 (subspace outlier degree). *Let $R(p)$ denote a set of reference objects for object $p \in \mathcal{D}$. The* subspace outlier degree *(SOD) of p w.r.t. $R(p)$, denoted by $SOD_{R(p)}(p)$, is defined as*

$$SOD_{R(p)}(p) := \frac{dist(o, \mathcal{H}(R(p)))}{\|v^{R(p)}\|_1},$$

i.e., the distance between point p and its reference set $R(p)$ according to Equation 2, normalized by the number of relevant dimensions as given e.g. by the number of entries $v_i^{R(p)} = 1$ in the weighting vector $v^{R(p)}$ as defined in Equation 1.

In contrast to most of the existing approaches, our model also gives an explanation *why* a point p is an outlier. Given an outlier p, we can obtain the subspace in which p is an outlier by simply inverting the subspace defining vector $v^{R(p)}$. This yields the subspace that is perpendicular to the subspace hyperplane of $R(p)$. In addition, we can derive the mean value of the points in $R(p)$ in that subspace. Thus, our model implicitly provides not only a quantitative outlier model but also a qualitative outlier model by specifying for each outlier the features that are relevant for the outlierness.

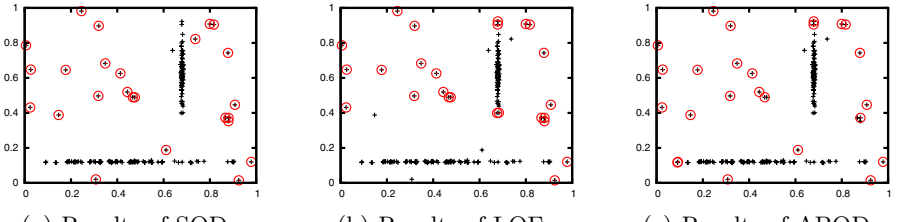
We now discuss how to choose a meaningful reference set for a given point $p \in \mathcal{D}$ to compute the outlierness of p. Existing local (full dimensional) outlier detection models usually examine the local neighborhood of p, e.g. the k-nearest neighbors or the ε-neighborhood based on Euclidean distance. However, due to the curse of dimensionality, distances cannot be used to differentiate points clearly in high dimensional feature spaces. As a consequence, the concept of "local neighborhood" is rather meaningless in high dimensional data (see e.g. [2]). An SNN approach usually measures the similarity of points based on the number of common nearest neighbors. An explanation for the robustness of SNN is that even though all points are almost equidistant to a given point p, a nearest neighbor *ranking* of the data objects is usually still meaningful. Two points p and q that have been generated by the same generating mechanism will most likely be neighbors or have similar neighbors in the subspace that is relevant for the common generating mechanism. Adding irrelevant attributes will blur these neighborhood relations by means of the absolute distances. However, most points of the common generating mechanisms will still be among the nearest neighbors of p and q. Thus, the number of shared neighbors of p and q will be large if both points originate from the same generating mechanism. Formally, let $N_k(p) \subseteq \mathcal{D}$ be the k-nearest neighbors of $p \in \mathcal{D}$ w.r.t. the distance function *dist*. The *shared nearest neighbor similarity* between two points $p, q \in \mathcal{D}$ is defined as $sim_{SNN}(p, q) = Card\,(N_k(p) \cap N_k(q))$. Now, the reference set $R(p)$ of p is the set of l-nearest neighbors of p using sim_{SNN}, i.e.,a subset of $\mathcal{D}$ that contains l points according to the following condition: $\forall o \in R(p), \forall \hat{o} \in \mathcal{D} \setminus R(p) : sim_{SNN}(\hat{o}, p) \leq sim_{SNN}(o, p)$.

The SOD algorithm relies on two input parameters. First, k specifies the number of nearest neighbors that are considered to compute the shared nearest neighbor similarity. This is not really a critical parameter as long as it is chosen high enough to grasp enough points from the same generating mechanism.

Second, l specifies the size of the reference sets. This parameter should also not be chosen too small for the same reason. Obviously, l should be chosen smaller or equal than k. Let us note that we have a third parameter α that specifies a threshold to decide about the significance of an attribute. If the variance of the reference set along an attribute is smaller than α times of the expected variance, then this attribute is considered relevant. In our experiments, setting $\alpha = 0.8$ yields consistently good results so we recommend to choose it accordingly. To compute the SOD, first the set of k-nearest neighbors of each of the n points of the database needs to be computed which requires in summary $O(d \cdot n^2)$ in the worst-case. This can be reduced to $O(d \cdot n \log n)$ if an index structure is applied to support the NN queries. Then, for each point p, the reference set of p consisting of the l nearest neighbors of p w.r.t. the SNN similarity needs to be computed which takes $O(k \cdot n)$, the mean and the variance of this reference set needs to be computed which takes $O(d \cdot l)$, and finally, the SOD can be computed. In summary, since $k \ll n$ and $l \ll n$, the runtime complexity of the latter steps and the overall complexity is in $O(d \cdot n^2)$ which is comparable to most existing outlier detection algorithms.

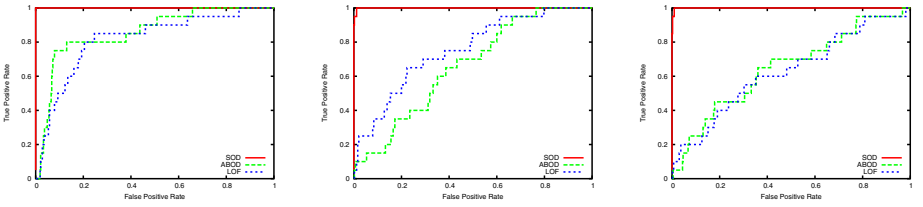
4 Experiments

We report the results of an experimental comparison of SOD with the full-dimensional distance-based LOF outlier model as one of the best-known outlier models and the full-dimensional angle-based ABOD outlier model as the most recent approach claiming to be specifically applicable to high dimensional data. All competitors are implemented within the ELKI-framework [6]. We first applied the competing outlier models to several synthetic data sets. Here, we focus on a toy 2D data set to illustrate the difference between a full dimensional approach like LOF or ABOD and the idea of a subspace outlier model followed by SOD. The results are visualized in Figure 2. Most points of the data set are produced by one of two generating mechanisms, for each mechanism only one attribute is relevant whereas the other is not. This results in one cluster of 80 points scattered along a line parallel to the y-axis and one cluster of 50 points scattered along a line parallel to the x-axis. In addition, 25 points have been generated randomly as outliers. The figures display the 25 top-ranked outliers by



(a) Results of SOD. (b) Results of LOF. (c) Results of ABOD.

Fig. 2. Comparison of the 25-top ranked outliers in a sample 2D data set



(a) 37 irr. attr. (b) 67 irr. attr. (c) 97 irr. attr.

Fig. 3. ROC curves for data sets of varying number of irrelevant attributes

each approach (marked by red circles). It can be observed that SOD has no problems in finding the outliers that deviate from the generating mechanisms. On the other hand, LOF and ABOD have two potential types of errors. First, points of the generating mechanisms are wrongly reported as outliers (false alarms) because the density is not high enough in the surrounding area. Second, points that are not generated by one of the generating mechanisms are missed (false drops) because their surrounding area is dense enough. We further conducted several experiments on higher dimensional synthetic data sets. Here, we defined a Gaussian distribution of 430 points in three dimensions with $\mu = 0.33$ and $\sigma = 0.08$. Additionally, 20 outliers are placed in a range of 0.455 to 1.077 as minimal and maximal distance from the cluster center, respectively, whereas the outmost cluster point has a distance of 0.347 from μ. These values are given w.r.t. the three relevant dimensions only. Then we added 7, 17, ..., 97 irrelevant attributes with values uniformly distributed in the range $[0, 1]$, resulting in 9 additional data sets of dimensionality 10, 20, ..., 100, respectively. In all experiments, SOD produced better results in terms of accuracy compared to LOF and ABOD. Figure 3 presents example ROC curves catching the performance of all three approaches for these data sets. While LOF and ABOD are very competitive in lower dimensional data sets, their performance considerably deteriorates with higher dimensionality while SOD remains very stable at optimal values. Only starting at 80 dimensions, 77 of which are irrelevant attributes, SOD starts to retrieve a false positive as the 19th outlier. Even at 100 dimensions, it only retrieves one false positive as 18th outlier.

We applied SOD and its competitors to a data set of career statistics of current and former NBA players[1] including 15 important parameters like points per game, rebounds per game, assists per game until the end of the 2007/2008 season. The data are normalized in order to avoid a bias due to different scaling of the attributes. The eight players with top SOD and ABOD values are displayed in Table 1. Both ABOD and SOD give some insightful results on this data set. They also agree on many of the top outliers. Eddy Curry — top outlier for both algorithms — for example is a significant outlier because of his 100% quote on three point field goals (2 of 2). We also ran LOF on this data set detecting mostly players with exceptional high values in particular parameters or particularly low

[1] Obtained from `http://www.nba.com`

Table 1. Results on NBA data set

(a) Top-8 outlier retrieved by SOD.

Rank	Name	SOD
1	Eddy Curry	0.0807
2	Dennis Rodman	0.0678
3	Amir Johnson	0.0560
4	Karl Malone	0.0473
5	Shawn Marion	0.0470
6	Michael Jordan	0.0457
7	Avery Johnson	0.0408
8	Andrei Kirilenko	0.0386

(b) Top-8 outlier retrieved by ABOD.

Rank	Name	ABOD
1	Eddy Curry	0.0021
2	Amir Johnson	0.0035
3	John Stockton	0.0043
4	Hakeem Olajuwon	0.0053
5	Dennis Rodman	0.0058
6	Karl Malone	0.0063
7	Shaquille O'Neal	0.0068
8	Andrei Kirilenko	0.0076

numbers.This experiment with real world data emphasizes that SOD can provide insightful information. It does not show a clear advantage over ABOD; they mostly agree on the outlier results. Depending on the use case, one or the other result can be seen as more useful. Outliers detected by ABOD can be seen as a more global kind of outlier, whereas SOD is stronger at detecting local outliers and additionally accounting for local feature correlation.

5 Conclusions

In this paper, we introduced SOD, a completely new approach to model outliers in high dimensional data. SOD explores outliers in subspaces of the original feature space by combining the task of outlier detection and relevant subspace finding. Our experimental evaluation showed that SOD can find more interesting and more meaningful outliers in high dimensional data with higher accuracy than full dimensional outlier models by no additional computational costs.

References

1. Hawkins, D.: Identification of Outliers. Chapman and Hall, London (1980)
2. Hinneburg, A., Aggarwal, C.C., Keim, D.A.: What is the nearest neighbor in high dimensional spaces? In: Proc. VLDB (2000)
3. Kriegel, H.P., Kröger, P., Zimek, A.: Clustering high dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. In: ACM Transactions on Knowledge Discovery from Data (TKDD) (to appear)
4. Breunig, M.M., Kriegel, H.P., Ng, R., Sander, J.: LOF: Identifying density-based local outliers. In: Proc. SIGMOD (2000)
5. Kriegel, H.P., Schubert, M., Zimek, A.: Angle-based outlier detection in high-dimensional data. In: Proc. KDD (2008)
6. Achtert, E., Kriegel, H.P., Zimek, A.: ELKI: a software system for evaluation of subspace clustering algorithms. In: Ludäscher, B., Mamoulis, N. (eds.) SSDBM 2008. LNCS, vol. 5069, pp. 580–585. Springer, Heidelberg (2008)

K-Dominant Skyline Computation by Using Sort-Filtering Method

Md. Anisuzzaman Siddique and Yasuhiko Morimoto

Hiroshima University,
1-7-1 Kagamiyama, Higashi-Hiroshima, 739-8521, Japan
Tel.: +81-82-424-5579
{d074370@,morimoto@mis.}hiroshima-u.ac.jp

Abstract. Skyline queries are useful in many applications such as multi-criteria decision making, data mining, and user preference queries. A skyline query returns a set of interesting data objects that are not dominated in all dimensions by any other objects. For a high-dimensional database, sometimes it returns too many data objects to analyze intensively. To reduce the number of returned objects and to find more important and meaningful objects, we consider a problem of **k**-dominant skyline queries. Given an **n**-dimensional database, an object **p** is said to **k**-dominates another object **q** if there are ($\mathbf{k} \leq \mathbf{n}$) dimensions in which **p** is better than or equal to **q**. A **k**-dominant skyline object is an object that is not **k**-dominated by any other objects. In contrast, conventional skyline objects are **n**-dominant objects. We propose an efficient method for computing **k**-dominant skyline queries. Intensive performance study using real and synthetic datasets demonstrated that our method is efficient and scalable.

Keywords: k-Dominant Skyline, Domination Power, Sort-Filtering.

1 Introduction

Skyline queries have attracted considerable attention due to its importance in many applications such as multi-criteria decision making, data mining, and user preference queries [1]. Given a database X, an object p is said to be in skyline of X if there is no other object q in X such that q is better than p in all dimensions. If there exist such a q, then we say that p is dominated by q, or q dominates p. A number of efficient algorithms for computing all skyline objects have been reported in the literature [1,2,3,4,5].

There are two problems in conventional skyline queries: (i) As the number of dimensions increases, the number of skyline objects increases substantially because it becomes difficult to dominate other objects. (ii) Usually, users have to select some noteworthy objects from skyline objects. Sometimes, the users may have to select many objects. Sometimes, they have to select a few objects. Conventional skyline query cannot control such selectivity.

In this paper, we consider k-dominant skyline queries [6], which are considered to overcome the above problems.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 839–848, 2009.
© Springer-Verlag Berlin Heidelberg 2009

1.1 Motivated Example

Assume a person want to purchase a notebook computer and is looking for suitable one. Assume there is a database containing eight notebooks as listed in Table 1. In the table, each notebook is represented as a tuple containing six attributes, CPU, RAM, HDD, HDD speed, Quality, and VRAM. Without loss of generality, we assume larger value is better in each attribute and all the attributes have equal importance.

Conventional skyline query for this database returns six notebooks: N_2, N_3, N_4, N_5, N_6 and N_7. N_1 and N_8 are not in skyline because both are dominated by N_2. If we look the six skyline notebooks, we can find that not all notebooks are significant in a sense. For example, N_3 is survived only by its value of "HDD Speed" and N_7 is survived only for "RAM" size. N_6 is skyline because no other notebook fails to dominate it in all dimensions, even though it does not have any maximal feature values. In such situation, the person naturally consider to eliminate the skyline notebooks by using stronger criterion.

Table 1. Database for Notebook PCs

NoteBook	CPU	RAM	HDD	HDD Speed	Q.	VRAM
N_1	3	3	5	6	6	8
N_2	9	4	9	7	7	9
N_3	8	4	7	9	2	7
N_4	5	6	8	9	5	9
N_5	9	7	9	6	2	4
N_6	6	6	6	5	3	5
N_7	5	7	3	8	4	6
N_8	4	4	8	6	6	4

Chan et al. considered k-dominant skyline query to handle the problem [6]. They relaxed the definition of "dominated" so that an object is likely to be dominated by another. Given a database X consists of n attributes, an object p is said to be in k-dominant skyline of X if there is no object q in X such that q is better than p in k ($\leq$n) dimensions. If there exist such a q, then we say that p is k-dominated by q or q k-dominates p.

In the example, if we consider 5-dominant skyline instead of conventional skyline, i.e., 6-dominant skyline, N_1, N_3, N_5, N_6, and N_8 are eliminated from 5-dominant skyline because they all are 5-dominated by N_2. N_7 fails to become 5-dominant skyline because it is 5-dominated by N_4.

If the person is satisfied with the selectivity of 5-dominant skyline, he/she can analyze the returned notebooks intensively. If he/she is still unsatisfied with the selectivity, he/she can compute further k-dominant skyline query with smaller k. Thus, k-dominant query solve the problems of conventional skyline query for high dimensional databases.

The contributions of this paper are as follows: 1) We have developped an efficient method for computing k-dominant skyline by using a *Sort-Filtering*

method that sorts objects by *domination power*. 2) We have performed intensive experiments on a variety of synthetic and real datasets to demonstrate that the proposed method is efficient and performs better than other existing methods.

2 *k*-Dominant Skyline

2.1 Preliminaries

Assume there is an n-dimensional database X containing m tuples. Let d_1,d_2,...,d_n be n attributes of X and let p_1,p_2,...,p_m be m tuples of X. We use $p_i.d_j$ to denote the j-th dimension value of p_i.

An object p_i is said to *dominate* another object q_j, denoted as $p_i \geq q_j$, if $p_i.d_k \geq q_j.d_k$ for all attributes d_k $(k = 1, \cdots, n)$ and $p_i.d_t > q_j.d_t$ for at least one dimension d_t $(1 \leq t \leq n)$. We call such p_i a *dominant object* and such q_j a *dominated object* between p_i and q_j. An object $p_i \in X$ is said to be a *skyline object* of X if p_i is not dominated by any other object in X.

An object p_i is said to *k-dominate* another object q_j, denoted as $p_i \geq_k q_j$, if $p_i.d_k \geq q_j.d_k$ in k attributes among n attributes and $p_i.d_t > q_j.d_t$ in an attribute d_t among the k attributes. We call such p_i as *k-dominant object* and such q_j as *k-dominated object* between p_i and q_j. An object p_i is said to be a *k-dominant skyline object* of X, if and only if there does not exist any object p_j $(j \neq i)$ in X that k-dominates p_i.

An object p_i is said to have *δ-domination power* if there are δ attributes in which p_i is better than or equal to all other objects of X.

2.2 A Priori Property

A k-dominant object has the following a priori property.

Theorem 1. *Any $(k-1)$-dominant object must be a k-dominant object for any k such that $1 < k \leq n$.*

Theorem 2. *Any k-dominated objects cannot be a $(k-1)$-dominant object for any k such that $1 < k \leq n$.*

Proof. Based on the definition, a $(k-1)$-dominant object p is not $(k-1)$-dominated by any other objects in X. It implies that p is not k-dominated by any other objects. Therefore, we can say p is k-dominant object. On the other hand, if an object q is k-dominated by another object, it must be $(k-1)$-dominated by the object. Therefore, q cannot be a $(k-1)$-dominant object.

The conventional skyline is the n-dominant skyline. If we decrease k of the k-dominant skyline, more objects are eliminated. For example, N_1 and N_8 of Table 1 are not in skyline because they are dominated (n-dominated) by N_2. So, they can't be a candidate of k-dominant skyline object for $k < n$. We can prune such non-skyline objects for further procedure of the k-dominant query. If we consider 5-dominant query, N_3, N_5, N_6, and N_7 are 5-dominated objects

in addition to the 6-dominated objects, N_1 and N_8. Therefore, we can prune those objects in 5-dominant query computation. Thus, by decreasing k, more dominated objects can be pruned away.

3 k-Dominant Skyline Algorithm

In this section, we present an efficient method for computing k-dominant skyline objects from X. We used a Sort-Filtering method that consists of two parts: one is "domination power" calculation and sorting, and the other is k-dominant skyline objects checking.

3.1 Domination Power Calculation

Objects whose sum of all their dimension values is large are likely to dominate other objects, while objects whose sum is small are likely to be dominated. Therefore, we sort the whole tuples in X in descending order of the sum of all their dimension values. This preprocess, sorting by sum, has been proposed by Chomicki et al. [4]. By this preprocessing, we can eliminate some of non-skyline objects easily. Chan et al. used the popular preprocessing in their OSA algorithm for k-dominant query [6]. But this preprocess is not effective for k-dominant query computation especially when values of each attribute is not normalized. For example, assume p(9,1,2) and q(3,2,3) are two objects in 3D space. Although the object p has greater sum than object q but p fails to become 2-dominant of q. Here, object p is 2-dominated by object q.

Therefore, in order to prune unnecessary objects efficiently in the k-dominant skyline computation, we compute *domination power* of each object, i.e., how many maximal values it has within all of dimensions. Then, we sort objects in descending order by domination power. If more than one objects have same domination power then we sort those objects in descending order of the sum value.

Without apply any sorting Table 2 represent the domination power and sum of the each Notebook PC's of Table 1. Table 3 is the example of sorted database (sort Notebook PC's in descending order by corresponding domination power

Table 2. Domination Power Calculation

Notebook	CPU	RAM	HDD	HDD S.	Q.	VRAM	Domination Power	Sum
N_1	3	3	5	6	6	8	0	31
N_2	9	4	9	7	7	9	4	45
N_3	8	4	7	9	2	7	1	37
N_4	5	6	8	9	5	9	2	42
N_5	9	7	9	6	2	4	3	37
N_6	6	6	6	5	3	5	0	31
N_7	5	7	3	8	4	6	1	33
N_8	4	4	8	6	6	4	0	32

Table 3. Sorted Dataset

Notebook	CPU	RAM	HDD	HDD S.	Q.	VRAM	Domination Power	Sum
N_2	9	4	9	7	7	9	4	45
N_5	9	7	9	6	2	4	3	37
N_4	5	6	8	9	5	9	2	42
N_3	8	4	7	9	2	7	1	37
N_7	5	7	3	8	4	6	1	33
N_8	4	4	8	6	6	4	0	32
N_1	3	3	5	6	6	8	0	31
N_6	6	6	6	5	3	5	0	31

and for same domination power sort in descending order of the sum value) of the Notebook PC's database of Table 1. In the sorted table, N_2 has the highest dominant power 4 and N_8, N_1 and N_6 have no dominant power. Note that N_2 dominates all notebooks lie below it in four attributes, CPU, HDD, Quality, and VRAM.

Let X' be the sorted database of X. X' has the following property.

Theorem 3. *Let p_i be the i-th object in the ordered object sequence of X'. If p_i has δ-domination power, it δ-dominates p_j for j such that $j > i$ in the sequence of X'.*

Proof. An object p_i with δ-domination power is δ-dominant object to any other object in X'. Since X' is sorted by domination power in descending order, an object with δ-domination power always comes before objects whose domination power is less than δ. Therefore, from the definition we can say p_i is a δ-dominant for any other object p_j since $j > i$.

By using this property, an object with k-domination power k-dominates all other following less domination power objects. Moreover, when more than one object has same domination power then our proposed method sort those objects in descending order of the sum values. Therefore higher objects in the sorted sequence are likely to dominate other objects. This sort filtering preprocessing helps to reduce the computational cost of k-dominant skyline.

The sorted sequence roughly reflects the importance of objects and our method can progressively output the k-dominant objects based on the sequence. It helps users' to make their decision more practical.

3.2 k-Dominant Checking

By using X', we progressively output k-dominant skyline objects as follows. We scan X' to compare each object $p \in X'$ against the first object q. In the scan procedure, objects that are k-dominated by the first object are removed from X'. During the procedure, if the first object q is k-dominated by any other objects $p \in X'$, we remove q from X' and stop the scanning procedure. If q is not

removed in the scanning procedure, then output q as a k-dominant object and remove q from X'. We repeat this scanning procedure until X' becomes empty.

Applying the k-dominant check with $k = 5$ for Table 3, we note that in the first scan N_5, N_3, N_8, N_1 and N_6 are 5-dominated by the first object, N_2. Therefore, those dominated objects are removed from X'. On the other hand, N_2 is not 5-dominated by any other notebooks. So, after the first scan, our method outputs N_2 as a 5-dominant skyline object and then remove N_2 from X'. In the second scan, N_4 becomes the first object and it 5-dominates N_7. So, we remove N_7 and outputs N_4 as a 5-dominant skyline object. Similarly, if we apply $k = 4$ for the same database X', then our Sort-Filtering method returns only N_2 as a 4-dominant skyline object.

4 Related Works

Chan et al. introduce k-dominant skyline query [6]. They proposed three algorithms to compute the k-dominant skyline query. The first algorithm, One-Scan Algorithm (OSA), uses the property that a k-dominant skyline objects cannot be worse than any skyline on more than k dimensions. This algorithm maintains the skyline objects in a buffer during a scan of the dataset and uses them to prune away objects that are k-dominated. As the whole set of skyline objects can be large, the authors proposed the Two-Scan Algorithm (TSA). In the first scan, a candidate set of dominant skyline objects is retrieved by comparing every object with a set of candidates. The second scan verifies whether these objects are truly dominant skyline objects. This method turns out to be much more efficient than the one-scan method. A theoretical analysis is provided to show the reason for its superiority. The third algorithm, Sorted Retrieval Algorithm (SRA), is motivated by the rank aggregation algorithm proposed by Fagin et al., which pre-sorts data objects separately according to each dimension and then merges these ranked lists [7].

As the authors mentioned in the OSA, skyline objects need to be maintained to compute the k-dominant skyline objects. Since the set of skyline objects could be much larger than the set of k-dominant skyline objects, maintaining skyline can incur large space and computation overhead.

Compared with their works, the proposed method can find k-dominant skyline objects without maintaining skyline. Therefore, there is no possibility for space and computational overhead. In addition, TSA algorithm scans whole data twice. In first scan, it generates candidate set of dominant skyline objects by comparing every object with a set of candidates and in second scan it verifies whether these objects are truly dominant skyline objects, while the proposed method can compute dominant skyline directly and does not suffer for false positive elimination procedure. As for SRA, the performance is uncertain because it depends crucially on the choice of proper dimension. Section 5 demonstrates that the performance of our algorithm is better than all of the three algorithms proposed in [6].

Algorithm called CoSMuQ also computes k-dominant skyline [8]. It divides the space in pairs of attributes and maintains a grid for each pair of dimensions. Each grid maintains its skyline tuples. Finally, the k-dominant skyline is obtained by the union of the skylines of these grids. This method has two severe problems, in high dimensional case. It needs to maintain huge number of grids. For example, if dimension size is equal to 15, then this algorithm needs to maintain 105 grids. In addition to the space complexity problem, CoSMuQ always needs to maintained 2-dominant skyline to compute k-dominant skyline. Compared with theirs, the proposed method does not suffer from such kinds grid as well as 2-dominant skyline maintaining problems.

5 Performance Evaluation

We have conducted a series of experiments to evaluate the performance of our Sort-Filtering method. We also compare the performance with all the algorithms proposed by Chan et al. which are One-Scan Algorithm (OSA), Two-Scan Algorithm (TSA) and Sorted Retrieval Algorithm (SRA) [6]. To make the comparison fair, we have include all the preprocessing cost, i.e., cost of domination power computation and sorting.

5.1 DataSets

We use both synthetic datasets and real datasets in the experiments. The generation of the synthetic datasets is controlled by three parameters, dimension number n, data size $Size$ and distribution $Dist$. There are three optional distributions in the synthetic data sets: Correlated, Independent and Anti-Correlated. Table 4 shows the number of the k-dominant skyline objects on 15-dimensional data set with 100k objects on different distributions and different constraint parameter k.

When k is close to the dimension size, the number of k-dominant skyline objects in the anti-correlated dataset becomes much larger than that of the independent and correlated datasets. However, when k is small, the correlated dataset can still have some dominant skyline objects, while no dominant skyline objects can be found on the other two distributions.

Table 4. Number of k-Dominant Skyline Objects

k	Correlated	Independent	Anti-Correlated
8	1	0	0
9	1	0	2
10	1	0	5
11	6	16	33
12	17	178	500
13	84	2180	5670
14	433	16143	29828

We also examine the performance for a real dataset. We used the NBA statistics. This dataset contains 17000 players season records on 17 attributes from the first season of NBA in 1945. Every records contain the statistical value of a player's performance in one season, such as game played(GP), field goal made(FGM), total assists(AST) and so on. One player may have several records if he played in NBA for more than one season.

5.2 Performance

We evaluated the computational cost of our Sort-Filtering method and compared the result with all three algorithms proposed by Chan et al. with similar parameter setting. The proposed method is implemented in Java. All experiments were conducted on a PC with an Intel Pentium 3GHz CPU and 2G main memory, which is running on Microsoft Windows XP operating systems. Figure 1, 2, and 3 are the time to compute k-dominant skyline for synthetic datasets.

Figure 1 examines the effect of the k value. We observe that our method is more efficient than other three methods on all distributions when k varies from 8 to 14. Because, maintaining the set of skyline objects for OSA incurs large space and computation overhead. To compute k-dominant skyline, TSA suffered for false candidates elimination. Again, the performance of SRA is not better than Sort-Filtering because this approach crucially depends on the choice of proper dimension.

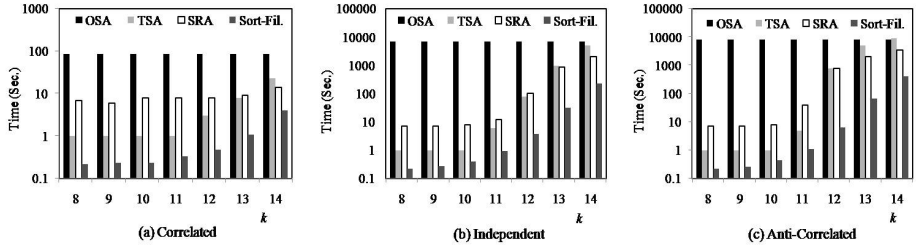


Fig. 1. k-Dominant Skyline Test on Varying k

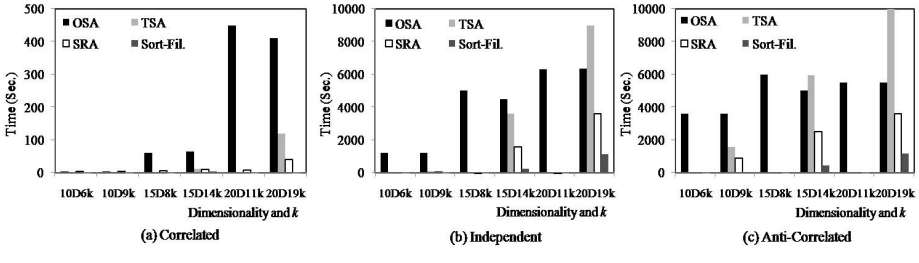


Fig. 2. k-Dominant Skyline Test on Varying *dimension*

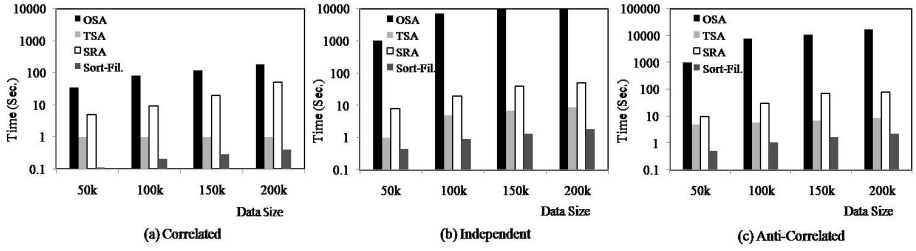


Fig. 3. k-Dominant Skyline Test on Varying *datasize*

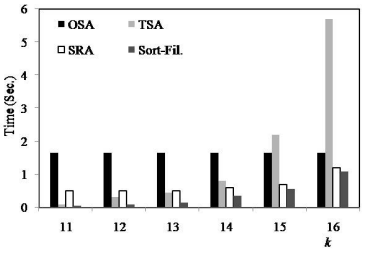


Fig. 4. k-Dominant Skyline Test on NBA Dataset Varying k

Figure 2 examines the effect of the dimension value. When k is small, TSA, RSA and Sort-Filtering are much faster than OSA on all three distributions. With increasing dimensionality TSA is several times slower than the other three algorithms. As shown in the figure, our Sort-Filtering is more scalable on high dimensional data sets.

In order to evaluate the effect of cardinality on the performance of the four algorithms, we use datasets with cardinality 50k, 100k, 150k and 200k. In this experiment, we fixed the size of n to 15 and k to 11. Figure 3 shows that when the size of the data set increases from 50k to 200k, the computation time of the four algorithms maintain a positive correlation. Notice that our Sort-Filtering performs best while OSA is the most worst.

In Figure 4, we show the experimental result on the NBA data set. When varying the constraint parameter k, TSA and our Sort-Filtering are the efficient algorithm when $k < 14$, but RSA is worst among the four algorithms when $k > 15$. Sort-Filtering and SRA are faster than other two when k is large.

6 Conclusion

We consider k-dominant skyline query problem and present a Sort-Filtering method. We demonstrate that our method is easy to compute and can be used for high dimensional large datasets. Performance evaluations show the superiority of the proposed method against the OSA, TSA and SRA algorithms.

k-Dominant skyline reduces the number of interesting objects returned by skyline. But sometimes the number of k-dominant skyline objects can still be large when k is not sufficiently small. If k is too small, no (or few) k-dominant skyline objects are returned. Though our efficient computation allows us to compute k-dominant objects for various k, proper guide for choosing the right value of k is an open problem.

Acknowledgements

This work was supported by KAKENHI (19500123) and Md. Anisuzzaman Siddique was supported by the scholarship of MEXT Japan.

References

1. Xia, T., Zhang, D., Tao, Y.: On Skylining with Flexible Dominance Relation. In: Proceedings of ICDE, Mexico, pp. 1397–1399 (2008)
2. Borzsonyi, S., Kossmann, D., Stocker, K.: The Skyline Operator. In: Proceedings of ICDE, Germany, pp. 421–430 (2001)
3. Kossmann, D., Ramsak, F., Rost, S.: Shooting Stars in the Sky: An Online Algorithm for Skyline Queries. In: Proceedings of VLDB, China, pp. 275–286 (2002)
4. Chomicki, J., Godfrey, P., Gryz, J., Liang, D.: Skyline with Presorting. In: Proceedings of ICDE, India, pp. 717–719 (2003)
5. Papadias, D., Tao, Y., Fu, G., Seeger, B.: Progressive Skyline Computation in Database Systems. ACM Transactions on Database Systems 30(1), 41–82 (2005)
6. Chan, C.Y., Jagadish, H.V., Tan, K.-L., Tung, A.K.H., Zhang, Z.: Finding k-Dominant Skyline in High Dimensional Space. In: Proceedings of ACM SIGMOD, USA, pp. 503–514 (2006)
7. Fagin, R., Lotem, A., Naor, M.: Optimal Aggregation Algorithms for Middleware. In: ACM PODS, USA, pp. 102–113 (2001)
8. Kontaki, M., Papadopoulos, A.N., Manolopoulos, Y.: Continuous k-Dominant Skyline Computation on Multidimensional Data Streams. In: Proceedings of ACM SAC, Brazil, pp. 16–20 (2008)

Effective Boosting of Naïve Bayesian Classifiers by Local Accuracy Estimation

Zhipeng Xie

School of Computer Science
Fudan University, Shanghai 200433, China
xiezp@fudan.edu.cn

Abstract. This paper investigates an effective boosting method for naïve Bayesian classifiers. Existing work has shown that the boosted naïve Bayesian classifier is not so effective in error rate reduction as the boosted decision tree (or boosted decision stump). This phenomenon may be caused by the combination of a couple of facts. To solve the problem, the local accuracies of a naïve Bayesian base classifier should be used to replace the global accuracy (or global error rate) in the traditional boosting methods. Based on the analysis, we propose an effective boosted naïve Bayesian method which uses a C4.5 decision tree as the local-accuracy evaluator for each base classifier. At each round, two classifiers are constructed: one for the naïve Bayesian base classifier, while the other for the C4.5 evaluator. The estimated local accuracy plays an important role, not only in updating the weights of training examples but also in determining the vote weights of base classifiers. Finally, it has been shown by experimental comparison that our method has achieved much lower error rate on average in a set of domains than the AdaBoost.M1 of naïve Bayesian classifiers.

Keywords: Naïve Bayesian classifier, AdaBoost, boosting, local accuracy estimation.

1 Introduction

Boosting is a kind of well known, effective technique for increasing the accuracy of base learning algorithms [6][7][11]. It has been shown to do well with algorithms such as C4.5 [9], Decision Stumps [6], and Naïve Bayes [4]. For example, it was reported that boosting for decision trees has achieved a relative error reduction of 27% on a set of domains [1]. However, some experiments have also revealed that the boosted naïve Bayesian classifier is not so effective in error rate reduction as the boosted decision tree (or boosted decision stump) [1][12]. One possible reason for this phenomenon is that naïve Bayesian classifier is less sensitive to small changes in training data than the decision tree classifier. Thus, it is preferred that we make a versatile change to the training data at each round.

The AdaBoost method uses a relatively simple weight-update rule. At each round, the weights of all the misclassified training examples are multiplied by the same factor which is larger than 1, while the weights of all the correctly-classified are divided by

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 849–858, 2009.
© Springer-Verlag Berlin Heidelberg 2009

this factor. The factor used at each round is determined only by the global error rate (or global accuracy) of the corresponding base classifier. However, the global accuracy provides only a rough measurement of the base classifier, because the base classifier may have different local accuracies in different areas (or instance subspaces).

Next, we would like to further justify the necessity of adopting local accuracies into the boosting process by considering the meaning of the weight of an instance. In boosting methods, the weight of a training example reflects its "hardness" for class label prediction. If two training examples have one same weight at the end of the round t and they are both misclassified at the round t+1, then they are of another same weight at the end of the round t+1. This situation is not satisfying in that the base classifier at the round t+1 may have significantly different local accuracies at these two instance points. The training example with higher local accuracy is clearly harder for the base classifier to make classification, and its weight should be increased to a higher value. Taking local accuracies into consideration may result in a more powerful weight-update rule, and then make versatile changes to the training data.

Thus, the main thrust of this paper is to design a novel and effective boosting method by integrating estimated local accuracies of the base classifiers into the boosting method. The local accuracies at different instance points provide more detailed performance information about a base classifier than its global accuracy, which makes it possible to make a versatile and significant change to the training data. To materialize this idea into a concrete method, a main problem exists in how to estimate the local accuracy of a naïve Bayesian base classifier at an instance point. This is solved by introducing an "evaluator", which is actually a C4.5 decision tree, for each nave Bayesian base classifier in the following steps: firstly, a Boolean meta-attribute is appended to each training example, and its value indicates whether a training example can be correctly classified by the base naïve Bayesian classifier with the leave-one-out technique; then, a C4.5 decision tree called "evaluator" is trained on the training set with each training example labeled by its meta-attribute value. This C4.5 decision tree can output a confidence which is actually an estimated local accuracy of the base classifier for a given test instance.

The proposed boosting method depends highly on these estimated local accuracies. During the boosting process, the estimated local accuracies for the training examples are used to determine the multiplying-factors in weight updating. On the other hand, during the decision fusion process for a given test instance, the estimated local accuracy of a base classifier not only determines whether this classifier will take part in the voting, but also determines its vote-weight if it takes part in the decision fusion process.

The whole paper is organized as follows. Section 2 introduces the naïve Bayesian base learning algorithm and describes what the meta-attribute is and how to calculate its value with the current base classifier using the leave-one-out technique. After filling up the meta-attribute value for each training example, section 3 gives out the details about how to use a C4.5 decision tree as the evaluator for estimating the local accuracy of the current base classifier at a given test instance. Section 4 explains how the boosting method works and presents its pseudo-code. Experimental results are shown in section 5. Finally, section 6 summarizes the whole paper.

2 Naïve Bayesian Classification and Meta-attribute

Consider a domain where instances are represented as instantiations of a vector $A=\{a_1, a_2, \ldots, a_m\}$ of m nominal variables. Here, each instance x takes a value $a_i(x)$ from $domain(a_i)$ on each a_i. Further, an example (or instance) x is also described by a class label $c(x)$ from $domain(c)=\{1, \ldots, K\}$. Let $S=\{(x_i, y_i=c(x_i)) \mid 1\le i\le n\}$ denote the training dataset of size n. The task of classification is to construct a model (or classifier) from the training set S, which is a function that assigns a class label to an unlabelled test instance.

Naïve Bayesian learning [3] is a probability-based simplest classification method. It has surprisingly good performance in a wide variety of domains and is robust to noise and irrelevant attributes. The underlying assumption that it takes is that attributes are conditionally mutually independent given the class label. According to the Bayes Theorem, the probability of a class label cl for a given unlabelled instance $x=(v_1, \ldots, v_m)$ consisting of m attribute values, where $v_k=a_k(x)$, $1\le k\le m$, is given by

$$P(c=cl \mid x)=\frac{P(c=cl)\times P(x \mid c=cl)}{P(x)}.$$

It then follows from the independence assumption that $P(x|c=cl)=\prod_{k=1}^{m} P(a_k=v_k \mid c=cl)$ holds. Thus, the class label with the highest probability given the instance x, is used as the predicted class. Note that we do not need to compute the value of $P(x)$. This is because $P(x)$ is a constant for a given x. To put it formally, the naïve Bayesian classifier trained on S can be expressed as

$$NB(x,S)=\arg\max_{cl}\left(P(c=cl)\times\prod_{k=1}^{m} P(a_k=a_k(x) \mid c=cl)\right)$$

The local accuracy of a classifier at a given test instance is usually measured as its accuracy on a small instance subspace containing the instance. In other words, the local accuracy can be estimated as the weighted percentage of the correctly-classified training examples in an instance subspace containing the test instance. To calculate this value, a basic job is to determine whether a given training example is classified correctly or not by the classifier. This can be done by simply applying the classifier on this example and comparing the predicted class label with its real label. However, to ensure the generalization ability, this method is not adopted here, and instead, the leave-one-out technique is used, which first updates the classifier by dropping the training example from it before applying the classifier to decision making. The results for all the training examples are recorded in a meta-attribute. Formally speaking, we append a meta-attribute c_{meta} to each training example (x_i, y_i) to indicate whether it can be classified correctly by the naïve Bayesian classifier or not:

$$c_{meta}(x_i)=\begin{cases} true & if\ NB(x_i, S-\{(x_i, y_i)\})=y_i; \\ false & if\ NB(x_i, S-\{(x_i, y_i)\})\neq y_i. \end{cases} \quad (1)$$

Here, we would like to explain in more detail how this work can be carried out. Let $NB(S)$ be the Naïve Bayes classifier trained on the training set S, and then $NB(x, S)$ be the predicted class label by $NB(S)$ for a test instance x. By updating the corresponding count information, we can efficiently obtain the Naïve Bayes classifier $NB(S-\{(x_i, y_i)\})$ which corresponds to the dataset $S-\{(x_i, y_i)\}$ in time $O(m)$. An instance x_i in S is said to be correctly classified by the naïve Bayes method, with $c_{meta}(x_i)=true$, if the predicted class label $NB(x_i, S-\{(x_i, y_i)\})$ for x_i equals to its real class label y_i; otherwise, it is said to be classified incorrectly and we have $c_{meta}(x_i)=false$. The calculation of the c_{meta} value for the instance takes the time $O(m)$. Finally, the training example (x_i, y_i) is put back into the naïve Bayesian classifier to re-obtain $NB(S)$, also in time $O(m)$. By repeating the above process for all the training examples, a derived dataset is got as: $S_{derived}=\{(x_i, c_{meta}(x_i))|1\le k\le m\}$. As we shall see in section 3, this derived dataset will be used as the input to the C4.5 decision tree algorithm, and the resulting decision tree classifier can be used to estimate the local accuracies of the naïve Bayesian classifier at different instance points.

3 Local Accuracy Estimation by Decision Trees

For a given base naïve Bayesian classifier, it would be interesting and also useful to measure its local accuracy a given instance, which can usually be estimated as the weighted proportion of correctly-classified training examples in an instance subspace around the instance. This section focuses on a solution by training a C4.5 decision tree on the derived dataset $S_{derived}=\{(x_1, c_{meta}(x_1)), (x_2, c_{meta}(x_2)), \ldots, (x_n, c_{meta}(x_n))\}$, where each instance x_i is labeled by its meta-attribute value $c_{meta}(x_i)$ indicating whether x_i can be correctly classified by the base classifier. This C4.5 decision tree, called the "evaluator" of the base classifier, can output its estimated local accuracy at a given instance.

In a decision tree, each node corresponds to an instance subspace and thus a training subset. The induction process starts with the root node which corresponds to the whole instance space and thus contains all the training examples. Then at each node, the decision-tree induction algorithm tries to select a test attribute on which to partition the training set is made on the basis of heuristic criteria such as gain or gain ratio. A nominal attribute a selected partitions the current subspace into $|domain(a)|$ reduced smaller subspaces, each of which corresponds to one value in the domain of the attribute, while a continuous attribute a together with a threshold partitions the current subspace into two smaller subspaces, one contains the instances whose values on a are smaller than or equal to the threshold, while the other contains the instances whose values on a are greater than the threshold. Finally, the resulting decision tree dt partitions the instance space X into N disjoint subspaces (or leaf nodes) $X_1, X_2, \ldots, X_N$ such that $X_1\cup X_2\cup\ldots\cup X_N=X$. Correspondingly, the training dataset $S_{derived}$ is partitioned into $S_1, S_2, \ldots, S_N$ with $S_j=train(X_j)=\{x_i\in S \mid x\in X_j\}$. For each j, the subspace (or the leaf node) X_j corresponds to a weight vector $W^j=(W^j_{true}, W^j_{false})$, where $W^j_{true}=\sum_{i:x_i\in X_j\wedge c_{meta}(x_i)=true} w(x_i)$ is the sum of the weights of the training examples that fall in subspace X_j and can be classified correctly by the current naïve Bayesian base classifier, and $W^j_{false}=\sum_{i:x_i\in X_j\wedge c_{meta}(x_i)=false} w(x_i)$ is the sum of the weights of the training examples that fall in

subspace X_j and can not be classified correctly. Therefore, for a given instance $x \in X_j$, the local accuracy is estimated as the Laplace ratio:

$$LocalAcc_{dt}(x) = \frac{W^j_{true} + 1}{W^j_{true} + W^j_{false} + 2}. \quad (2)$$

where *dt* denotes the evaluator decision tree corresponding to the base classifier.

The process described above is under the assumption that the values of each instance on all attributes are known, which may not be the case in reality. The real situation is complicated a little more by the fact that instances may have missing values on some attributes. The C4.5 algorithm [8] adopts a probabilistic way to deal with this situation. A test instance x with missing values on some attributes will traverse from the root node to a number of leaf nodes, and it belongs to each such leaf node with a certain probability that is estimated from the training set (As to the details of how to calculate this probability, please refer to the Chapter 3 in [8]). Let us use $J(x)$ to denote the set of leaf nodes corresponding to the instance x, $P(X_j, x)$ to denote the probability that x belongs to X_j for each $X_j \in J(x)$. Thus, the instance x corresponds to a weight vector $W(x) = \sum_{X_j \in J(x)} P(X_j, x) \times W^j$. The local accuracy of the current naïve Bayesian base classifier for the instance x is estimated as:

$$LocalAcc_{dt}(x) = \frac{W(x)_{true} + 1}{W(x)_{true} + W(x)_{false} + 2}. \quad (3)$$

It should be noted that the estimated local accuracies are measured with respect to the boosted weights at each round in the boosting process (not the original weights).

4 Effective Boosting of Naïve Bayesian Classifiers

The motivation of the proposed boosting method for naïve Bayesian classifiers is to integrate the estimated local accuracies (instead of the global accuracy, or equivalently global error rate, in original Adaboost method) into the boosting method. As we shall see in this section, estimated local accuracy plays an important role in both the boosting process and the decision fusion process. This paper uses C4.5 decision trees as the evaluators of local accuracies for the base naïve Bayesian classifiers, which has already been described in section 3. The task of this section is to develop an effective method for boosting naïve Bayesian classifiers, called EffectiveBoost-NB, with its pseudo-code listed in figure 1.

In EffectiveBoost-NB, two classifiers are constructed at each round *t*. One is the naïve Bayesian base classifier $NB^{(t)}$ trained on the set of weighted training examples: $\{(x_1, y_1=c(x_1)), (x_2, y_2=c(x_2)), \ldots, (x_n, y_n=c(x_n))\}$ (line 3 in Figure 1). Next, for each training example, the algorithm uses leave-one-out technique to judge whether this base classifier can make a correct classification, and updates its meta-attribute value with the result (lines 4-6 in Figure 1). Then, the other C4.5 classifier $dt^{(t)}$, serving as the local-accuracy evaluator of the base classifier, is trained on the derived dataset $S_{derived}=\{(x_1, c_{meta}(x_1)), (x_2, c_{meta}(x_2)), \ldots, (x_n, c_{meta}(x_n))\}$, where each example x_i is labeled by its meta-attribute value $c_{meta}(x_i)$ (line 7 in Figure 1).

After these two classifiers have been built, it is the time to update the weights of all the training examples in order to make necessary preparation for the next round. The way that the weight of a training example (x_i, y_i) is updated depends on two factors: (1) what is the local accuracy of the current base classifier (that is, the base classifier at round t) at the instance point x_i; and (2) whether the instance x_i can be correctly classified by the base classifier at round t, with the leave-one-out technique. To answer the first question, the current C4.5 evaluator is used to estimate the local accuracy; while for the second question, the meta-attribute value is checked to provide the answer. For each training example (x_i, y_i), the weight-update rule goes as follows:

(1) If the estimated local accuracy is larger than 0.5 and it can be correctly classified (line 13 in Figure 1), the weight is updated by multiplying the factor $\sqrt{(1-LocalAcc_{dt^{(t)}}(x_i))/LocalAcc_{dt^{(t)}}(x_i)}$, where $LocalAcc_{dt^{(t)}}(x_i)$ is the estimated local accuracy of the current base classfier. In this situation, the weight gets decreased because the factor is less than 1 due to the estimated local accuracy is larger than 0.5.
(2) If the estimated local accuracy is larger than 0.5 and it cannot be correctly classified (line 15 in Figure 1), its weight is updated by multiplying a factor $\sqrt{LocalAcc_{dt^{(t)}}(x_i)/(1-LocalAcc_{dt^{(t)}}(x_i))}$. Here, the weight gets increase because the factor is larger than 1.
(3) Otherwise (that is, when the estimated local accuracy is less than 0.5), the weight remains unchanged.

The weight-update procedure here differs from that of the classical AdaBoost algorithm in two aspects: (1) the local accuracies specific to individual training examples are used instead of the global accuracy, which makes the change of weights versatile; (2) we use leave-one-out technique in judging whether a training example can be correctly classifier or not, in order to guarantee the generalization ability, while AdaBoost applies the base classifier without dropping the training example out. On the other hand, it shares some similarity with the boosting with specialist models [10] by the fact that: for those training examples that have local accuracies less than or equal to 0.5, their weights are not subject to the weight-update. After the weights of all the training examples have been updated, all these weights are normalized to make them a distribution (line 20 in Figure 1).

In addition, there are two points to clarify. The first is about the Boolean variable *ConfidentEnough*, which is operated in the line 8, line 11, and line 19 of the algorithm. If none of the local accuracies (of the current base classifier) at all the instance points of the training examples are larger than 0.5, it means that all the instance weights will remain unchanged, and the boosting process has to terminate (line 19 in Figure 1). This is analogous to the situation in AdaBoost.M1 that the current base classifier has an error rate higher than 1/2. Assume that the current base classifier has an error rate of *err*. Generally speaking, its estimated local accuracies at some instance points are higher than *err*, while those at other points are lower. Therefore, even if the base classifier at the current round satisfies the termination condition (that is, $err \leq 0.5$) of AdaBoost.M1, it may not satisfy the termination condition of EffectiveBoost-NB and the boosting process may continue.

ALGORITHM EffectiveBoost-NB

INPUT: a data set S of n training examples, $S=\{(x_1, y_1), \ldots, (x_n, y_n)\}$
Integer T specifying number of iterations

1. Initialize $D_1(i)=1/n$ and $w(x_i)=n\times D_1(i)$ for all i.
2. FOR $t=1, \ldots, T$:
3. train a naïve Bayesian base classifier $NB^{(t)}=NB(S)$ on the training examples with weights D_t
4. FOR each instance x_i DO
5. $$c_{meta}(x_i):=\begin{cases} true & \text{if } NB(x_i, S-\{(x_i, y_i)\}) = y_i; \\ false & \text{if } NB(x_i, S-\{(x_i, y_i)\}) \neq y_i. \end{cases}$$
6. ENDFOR
7. train a C45 evaluator $dt^{(t)}=C45(S_{derived})$ on the derived data with weights w
8. *ConfidentEnough*:=*false*
9. FOR each instance x_i DO
10. IF $LocalAcc_{dt^{(t)}}(x_i) > 0.5$ THEN
11. *ConfidentEnough*:=*true*
12. IF $c_{meta}(x_i)$=*true* THEN
13. $$D_{t+1}(i):=D_t(i)\times\sqrt{(1-LocalAcc_{dt^{(t)}}(x_i))/LocalAcc_{dt^{(t)}}(x_i)}$$
14. ELSE
15. $$D_{t+1}(i):=D_t(i)\times\sqrt{LocalAcc_{dt^{(t)}}(x_i)/(1-LocalAcc_{dt^{(t)}}(x_i))}$$
16. ENDIF
17. ENDIF
18. ENDFOR
19. IF *ConfidentEnough*=*false* THEN T:=t; break; ENDIF
20. Normalize the distribution: $D_{t+1}(i):=\frac{D_{t+1}(i)}{Z_t}$, where Z_t is a normalization constant chosen so that D_{t+1} be a distribution.
21. Update the instance weights: $w(x_i):=D_{t+1}(i)\times n$ for each x_i
22. ENDFOR

OUTPUT: the final strong classifier:

$$f_{strong}(x) = \arg\max_{y\in\{1,\ldots K\}}\left(\sum_{\substack{t:LocalAcc_{dt^{(t)}}(x)>0.5 \\ and\ NB^{(t)}(x)=y}} \log\left(\frac{LocalAcc_{dt^{(t)}}(x)}{1-LocalAcc_{dt^{(t)}}(x)}\right)\right)$$

Fig. 1. EffectiveBoost-NB: an effective method for naïve Bayesian classifiers

The second point is about the usage of the two copies of the instance weight information used in the algorithm. Boosting methods usually require that the weights of training examples be a distribution (i.e. their sum should be equal to 1), while naïve

Bayesian method and C4.5 learning algorithm traditionally treats the weights as the numbers of instances (i.e. their sum should be equal to the total number of examples). Therefore, two copies (w and D) of the weight information are maintained in the algorithm, and they should satisfy the constraint: $w(x_i) = D_{t+1}(i) \times n$ for each x_i ($1 \le i \le n$), which is maintained as the line 21 in Figure 1 does at the end of each round..

For a given test instance, the decision fusion procedure is illustrated as the final "**OUTPUT**" line in Figure 1. For a given test instance x, a naïve Bayesian base classifiers $NB^{(t)}$ takes part in the voting on the class label if and only if its local accuracy at the test instance point (estimated by the corresponding C4.5 evaluator) is larger than 0.5. In other words, the voting committee for the test instance x consists of those base classifiers whose local accuracies estimated is high enough. If a base classifier $NB^{(t)}$ is selected into the committee and takes part in the voting, it will vote on the class label $NB^{(t)}(x)$ by the weight $\log\left(\frac{LocalAcc_{dt^{(t)}}(x)}{1-LocalAcc_{dt^{(t)}}(x)}\right)$, where $dt^{(t)}$ is the evaluator corresponding to the base classifier $NB^{(t)}$. Clearly, this vote weight also depends only on the estimated local accuracy. The class label with the highest sum of votes is the final decision for the test instance. This final composite classifier is has two characteristics which make it distinct from the classical AdaBoost methods: (1) a selective voting committee of base classifiers are formed *dynamically* for each given test instance; and (2) the vote weights of the classifiers in the subset are determined *dynamically* by the given test instance. The proposed EffectiveBoost-NB algorithm in this paper is actually a *dynamic* classifier ensemble method.

5 Experimental Results

To demonstrate and compare the performance of the proposed boosting method for naïve Bayesian classifiers, we randomly pick out 30 datasets from UCI machine learning repository [2], and use them in the experiments, with the detailed information summarized in the first column of table 1. These datasets have many distinct features, including the number of attributes, the number of classes, and the size of each dataset. On each dataset, a ten-fold cross validation is applied to estimate the error rate. The entropy-based discretization algorithm in [5] was employed to discretize the numeric attributes in the training sets for each fold, as pre-processing.

In this study, the proposed EffectiveBoost-NB algorithm is compared with other two algorithms: the naïve Bayesian algorithm and the AdaBoost.M1 of naïve Bayesian classifiers. The error rates of these algorithms over all datasets are listed in table 1. Compared with standard naïve Bayesian method, EffectiveBoost-NB has lower error rates than NB on 22 datasets and has higher error rates than NB on 7 datasets (with only 1 tie). Even when compared with AdaBoost-NB, EffecitveBoost-NB performs better on 20 datasets and worse on 6 datasets (with 4 ties).

The final row in table 1 shows the mean error rates across all the datasets. Among the three algorithms, EffectiveBoost-NB gets the best result, which is much better than both NB and AdaBoost-NB.

In addition, table 2 presents the results of the one-tailed pairwise t-test (with significance level set at 5%). It shows that EffectiveBoost-NB wins significantly on 13

datasets, and loses on none of the datasets when compared with NB. Even when compared with AdaBoost-NB, EffectiveBoost-NB also wins significantly on 9 datasets, also without loses. When comparing AdaBoost-NB with NB, we find that AdaBoost-NB significantly wins on 8 datasets and loses on 1 datasets.

As a conclusion, the proposed EffectiveBoost-NB provides an effective method for boosting naïve Bayesian classifier, and it has significantly reduced both the error rate of naïve Bayesian method and that of the AdaBoost.M1 of naïve Bayesian classifiers.

Table 1. Error rate comparison

Dataset(#examples, #attributes, #classes)	NB	AdaBoost-NB	EffectiveBoost-NB
Australian(690, 14, 2)	14.3%	14.3%	14.6%
Breast(699, 10, 2)	3.0%	4.2%	3.9%
Chess(3196, 36, 2)	12.0%	5.1%	0.8%
Cleve(303, 13, 2)	16.5%	15.2%	16.5%
Crx(690, 15, 2)	14.2%	15.7%	15.4%
Diabetes(768, 8, 2)	25.4%	25.3%	24.7%
German(1000, 20, 2)	25.2%	25.4%	26.9%
Heart(270, 13, 2)	17.0%	17.0%	15.9%
Hepatitis(155, 19, 2)	14.3%	16.9%	16.2%
Horse-Colic(368, 22, 2)	21.2%	22.0%	18.5%
Hypothyroid(3163, 25, 2)	1.5%	1.5%	1.2%
Ionosphere(351, 34, 2)	10.0%	9.6%	7.7%
Iris(150, 4, 3)	7.3%	6.7%	6.7%
Monk1(432, 6, 2)	25.0%	25.7%	0.5%
Monk2(432, 6, 2)	33.1%	33.3%	29.2%
Monk3(432, 6, 2)	2.8%	0.0%	0.0%
Mushroom(8124, 22, 2)	3.5%	0.0%	0.0%
Nursery(12960, 8, 5)	9.7%	8.1%	0.7%
Pendigits(10992, 16, 10)	12.3%	7.7%	4.8%
Pima(768, 8, 2)	25.3%	25.4%	24.7%
Satimage(6435, 36, 6)	17.9%	17.6%	14.0%
Segment(2310, 19, 7)	8.8%	8.3%	5.6%
Shuttle-Small(5800, 9, 7)	0.7%	0.2%	0.2%
Sick(2800, 29, 2)	2.9%	3.1%	2.5%
Solar(323, 12, 6)	30.7%	31.3%	32.5%
Soybean-Large(683, 35, 19)	7.0%	6.3%	7.3%
Tic-Tac-Toe(958, 9, 2)	29.8%	15.9%	5.7%
Vehicle(846, 18, 4)	39.7%	39.4%	33.2%
Vote(435, 16, 2)	9.7%	5.3%	6.0%
Waveform-21(5000, 21, 3)	18.9%	18.9%	18.0%
Mean	14.8%	13.7%	11.3%

Table 2. Summary on t-test of experimental results. Each entry indicates that the scheme of the row compares against the scheme of the column.

win/loss/tie	NB	AdaBoost-NB
AdaBoost-NB	8/1/21	
EffectiveBoost-NB	13/0/17	9/0/21

6 Future Work

This paper proposes an effective boosting method for naïve Bayesian classifiers by using C4.5 decision trees as local accuracy evaluators. Experimental results have manifested that it can significantly reduced the error rates of the naïve Bayesian classifier and of the boosted naïve Bayesian classifiers. As the possible future work, it would be interesting to compare with another boosting method for naïve Bayesian classifiers presented in [12] which introduces tree structures into naïve Bayesian classification to form a new kind of base classifier called "Leveled naïve Bayesian tree". We also plan to investigate the effect of combining the boosting technique in this paper with the leveled naïve Bayesian trees.

Acknowledgements. This work was funded in part by National Natural Science Foundation of China under grant number 60503025.

References

1. Bauer, E., Kohavi, R.: An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. Machine Learning 36(1/2), 105–139 (1999)
2. Blake, C., Keogh, E., Merz, C.J.U.: Repository of Machine Learning Databases. University of California, Department of Information and Computer Science, Irvine, CA (1998), http://www.ics.uci.edu/~mlearn/MLRepository.html
3. Duda, R.O., Hart, P.E.: Pattern classification and scene analysis. John Wiley, New York (1973)
4. Elkan, C.: Boosting and naïve Bayesian learning. Technical Report CS97-557, Depart. Of CS and Engineering, University of California, San Diego, USA (1997)
5. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued at-tributes for classification learning. In: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, pp. 1022–1027. Morgan Kaufmann, San Francisco (1993)
6. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Machine Learning: Proceedings of the Thirteenth International Conference, pp. 148–156 (1996)
7. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55(1), 119–139 (1997)
8. Quinlan, J.R.: C4.5: Programs for machine learning. Morgan Kaufmann, San Francisco (1993)
9. Quinlan, J.R.: Bagging, boosting, and C4.5. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence, pp. 725–730 (1996)
10. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Machine Learning 37(3), 297–336 (1999)
11. Schapire, R.E.: The boosting approach to machine learning: An overview. In: Denison, D.D., Hansen, M.H., Holmes, C., Mallick, B., Yu, B. (eds.) Nonlinear Estimation and Classification. Springer, Heidelberg (2003)
12. Ting, K.-M., Zheng, Z.: Improving the performance of boosting for naïve Bayesian classification. In: Zhong, N., Zhou, L. (eds.) PAKDD 1999. LNCS, vol. 1574, pp. 296–305. Springer, Heidelberg (1999)

COMUS: Ontological and Rule-Based Reasoning for Music Recommendation System*

Seungmin Rho[1], Seheon Song[2], Eenjun Hwang[3], and Minkoo Kim[2]

[1] School of Computer Science, Carnegie Mellon Univeristy, Pittsburgh, PA, USA
smrho@andrew.cmu.edu
[2] Graduate School of Information and Communication,
Ajou University, Suwon, 443-749, South Korea
{lego,minkoo}@ajou.ac.kr
[3] School of Electrical Engineering, Korea University, Seoul, 135-701, South Korea
ehwang04@korea.ac.kr

Abstract. In this paper, we propose Context-based Music Recommendation (COMUS) ontology for modeling user's musical preferences and context for supporting reasoning about the user's desired emotion and preferences. The COMUS provides an upper Music Ontology that captures concepts about the general properties of music such as title, artists and genre and also provides extensibility for adding domain-specific ontologies, such as Music Feature, Mood and Situation, in a hierarchical manner. The COMUS is music dedicated ontology in OWL constructed by incorporating domain specific classes for music recommendation into the Music Ontology. Using this context ontology, we believe that the use of logical reasoning rules by checking the consistency of context information, and reasoning over the high-level, implicit context from the low-level, explicit information. As a novelty, our ontology can express detailed and complicated relations among the music, moods and situations, enabling users to find appropriate music for the application. We present some of the experiments we performed as a case-study for music recommendation.

Keywords: Music Recommendation, Ontology, Reasoning.

1 Introduction

With recent advances in the field of music information retrieval, we face a new possibility that music can be automatically analyzed and understandable by the computer to some semantic level. Due to the diversity and richness of music content, many researchers have been pursuing a multitude of research topics in this field, ranging from computer science, digital signal processing, mathematics, and statistics applied to musicology. Most traditional content-based music retrieval (CBMR) techniques [1, 2] have focused on low-level features such as energy, zero crossing rate, audio spectrum, and etc. However, these features were not enough to give semantic information of music contents, so that these gave limitation in retrieving and recommending appropriate music to people.

* This research was supported by the ubiquitous Computing and Network (UCN) Project, the Ministry of Knowledge and Economy (MKE) 21st Century Frontier R&D Program in Korea.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 859–866, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Due to the abovementioned limitations of low-level feature-based music retrieval, some researchers have tried to bridge the semantic difference, which is also known as semantic gap, between the low-level features and high-level concepts such as human emotion and mood [3, 4]. With low-level feature analysis only, we might experience many difficulties in identifying the semantics of musical content. Similarly, it is difficult to correlate high-level features and the semantics of music. For instance, a user's profile, which includes educational background, age, gender, and musical taste, is one possible high-level feature. Semantic web technology is considered to be one promising method to bridge this semantic gap. Also, Emotional Effects of Music work [15] suggested that an emotion is experienced by a listener while listening to music is determined by a multiplicative function consisting of several factors such as structural, performance, listener, and contextual features. We adapt some features to our emotion reasoning model for inducing his/her emotion when he/she in placed.

Therefore, in this paper, we try to tackle the abovementioned problem in the domain of music recommendation by combining content-based music retrieval, music ontology and domain specific ontology for Mood and Situation. More specifically, we define more specific domain-specific ontology based on the basic concepts from the upper ontology which can be found in the previous ontology-related projects such as Music Ontology [5, 6], Kanzaki taxonomy [7], and MusicBrainz [8].

In our scenario for music recommendation, we noted musical terms as concepts. The relations consist of several types, including hierarchical relations such as 'is-a' and 'has-a.' We deal with these two formal relations to indicate the specialization of concepts and required parts. For example, mood, genre and music features are subclass of music and MFCC, tempo, onset, loudness and chroma are member of music feature. The other important relation is 'like-song/singer/genre/mood'. This relation is used to describe user's musical and emotional preference. It helps us to find implicit relationship between music and user's preference.

2 Ontology Model

In order to provide music recommendation service intelligently, we need a set of common ontologies such as person, time, location, and etc, for knowledge sharing and reasoning. We have developed ontology including music and its related ontologies in the music recommendation domain. We use the W3C recommendation ontology language, OWL (Web Ontology Language) to represent ontology.

We adopt basic concepts and relations from previous work - the Music Ontology; we expand it to include additional features such as musical feature, genre, instrument taxonomy, mood, and situation. We serialize this ontology in OWL so that we can retrieve some information using SPARQL query language.

2.1 COMUS Ontology

COMUS (Context-based Music Recommendation) ontology consists of about 500 classes and instances, and 52 properties definitions. Fig. 1 shows a graph representation of some of key COMUS ontology definition. This ontology describes music related information about relationships and attributes that are associated with people, genre, mood (e.g., sad, happy, gloomy), location (e.g., home, office, street), time (e.g., morning, spring), and situation events (e.g., waking-up, driving, working) in a daily life.

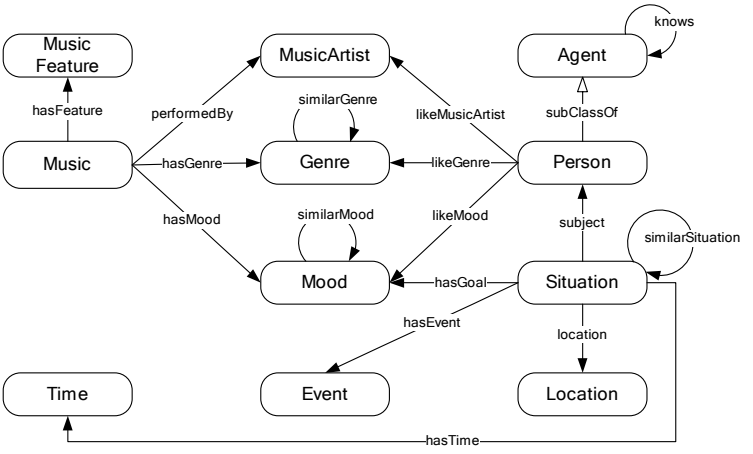


Fig. 1. This is a graph representation of the COMUS ontology. Each rounded rectangle with solid arrow represents an OWL class

The key top-level elements of the ontology consist of classes and properties that describe Person, Situation, Mood, Genre and Music classes.

Music. The "Music" class defines general properties of music such as title, released year, artists, genre, and musical features (e.g., MFCC, Tempo, Onset, Chroma, Segment, Spectra Centroid, Spectra Flux, Spectra Spread, and Zero Crossing Rate).

Genre. The "Genre" class defines the category of music. There have been many researches for the music genre classification. There exist several popular online systems such as All Music Guide, MusicBrainz and Moodlogic for annotating popular music genre and emotion. We create our own genre taxonomy based on All Music Guide [11] along with second level of industry taxonomy.

Person. The "Person" class defines generic properties of a person such as name, age, gender, hobby, socioeconomic background (e.g., job, final education) and music related properties for music recommendation such as musical education, favorite music, genre and singer.

Mood. The "Mood" class defines the state of one's mind or emotion. Each mood has a set of similar moods. For example, "aggressive" has similar moods like "hostile, angry, energetic, fiery, rebellious, reckless, menacing, provocative, outrageous, and volatile."

Situation. The "Situation" class defines person's situation in terms of conditions and circumstance, which are very important clues to effective music recommendation. Situation is described by time, location, subject, and goal to achieve. Hence, this class describes user's situational contexts such as whereabouts of the user (Location), what happens to the user (Event), and so on.

2.2 Scenario: Context-Based Music Recommendation

In this subsection, we will introduce a typical scenario that demonstrates how COMUS ontology can be used to support ontology reasoning for recommending appropriate music to users.

Example Scenario
John is a technical engineer and he is 35 years old. His favorite singer is "Stevie Wonder," and his favorite song is "Superstition", and he likes Pop and R&B style music. His hobby is playing baseball. He is a very positive person and likes bright, sweet music. When he feels sad, he usually listens to the music that might help cheer him up. The date is 12 January, 2009. John woke up late in the Monday morning and he is very tired due to working late last night. But he should go to work early to prepare for a presentation at today's meeting. Therefore, he asks the music recommendation system to look up some hard and fast beat music then he listens to the music like "Don't stop me now (Queen)," "You give a bad name (Bon Jovi)" and "Wake me up before you go go (Wham)." These kinds of beating music help him to hurry up to go to work in time. After a while, he was stuck in a traffic jam which started making him nervous on his way to work. In order to calm down that situation, he asks again the system to recommend some calm-down music of The Carpenters, Stevie Wonder and Simon & Garfunkel.

This scenario assumes that John has set his musical preferences such as singer, genre and mood to filter out the query result automatically. For example, the wake-up-late situation is analyzed and sent to the recommendation system. Using this information, the system will reason about John's situational context and his favorite mood from the user profile information. From this information, the system recommends a bunch of music which best fits John's interest and current situation.

Table 1. User-defined context reasoning rules

	Reasoning Rules
Wake up	(?u isLocatedIn Bedroom) ∧ (Datetime week MONDAY) ∧ (Datetime hour 7) ⇒ (?u event WAKE_UP)
Driving to work	(?u isLocatedIn Road) ∧ (?u isLocatedIn Car) ∧ (Datetime week MONDAY) ∧ (Datetime hour 8) ⇒ (?u event DRIVING_To_WORK)
Driving and Traffic Jam	(?u event DRIVING_TO_WORK) ∧ (?e TrafficJam) ⇒ (?u situation DRIVING_TO_WORK_TRAFFIC_JAM)
Situation goal	(?u situation DRIVING_TO_WORK_TRAFFIC_JAM) ∧ (?u likeMood) ⇒ (?s hasGoal ?goalMood)

A more flexible reasoning can be brought in by specifying user-defined reasoning rules towards defining high level conceptual contexts such as "What music does the user want to listen to when he/she is stressed?" can be deduced from relevant low-level context. Table 1 describes the user-defined context reasoning rules that are employed to derive user's mood and situation in the music recommendation scenario.

One can ask questions about the situation by using queries. The difference between queries to a database and queries to an OWL knowledge base is that are inferred as well as facts that have been explicitly asserted. When we get a situation goal (goal mood), we can query to music ontology to find a list of recommendable music as shown below:

```
PREFIX comus: <http://ceai.ajou.ac.kr/ontology/0.9/
comus.owl#>
SELECT DISTINCT ?Song ?UserMood
FROM <http://ceai.ajou.ac.kr/ontology/0.9/comus.owl>
WHERE
{ ?Person comus:hasName "John";
  comus:likeMood ?UserMood.
  ?Situation comus:hasSituationName "Wake Up Late";
  comus:hasUserMood ?UserMood.
  ?UserMood comus:hasSimilarMood ?SimiliarMood.
  {?Song comus:hasMood ?UserMood}
  UNION {?Song comus:hasMood ?SimiliarMood}
  } ORDER BY ?Song
}
```

3 System Implementation

In this paper, we develop a prototype music recommendation system and extended music ontology to enable mood and situation reasoning in a music recommendation system. Our system provides various types of query interfaces to the users. The user can formulate queries using one of three different query interfaces: they are query by situation (QBS), query by detailed situation (QBDS), and query by mood (QBM). Our COMUS ontology is described in OWL language using Protégé editor [9]. During the construction music ontology, we use RacerPro to check consistency of our ontology, and to retrieve instance. In that case, missing valuable information could be inferred from the partial information in the profile through finding implicit related classes. For the retrieval and recommendation, the Jena SPARQL engine is used to express and process necessary queries to our ontology.

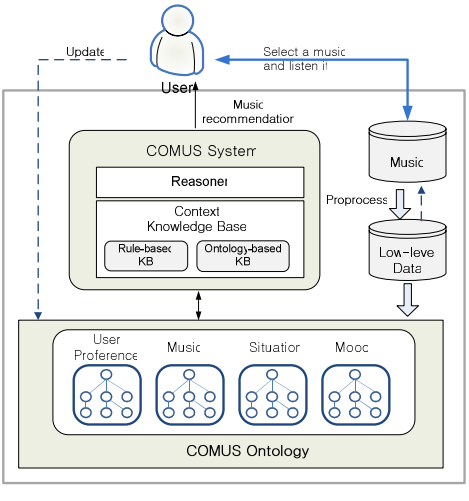


Fig. 2. Overview of our system

According to [10], the development of ontology is motivated by scenarios that arise in the application. A motivating scenario provides a set of intuitively possible solutions to the problems in the scenario. The COMUS ontology is a collection of terms and definitions relevant to the motivating scenario of music recommendation that we described above. Thus, to build ontology, it is important to start with describing the basic concepts and one or more scenarios in the specific domain of interest.

After building the ontology, the next step is to formulate competency questions. These are also based on the scenarios and can be considered as expressiveness requirements that are in form of questions. Ontology must be able to represent these questions using its domain-related terminology and characterize their answers using the axioms and definitions. Therefore, we asked participants to answer the competency questions through the online questionnaire system.

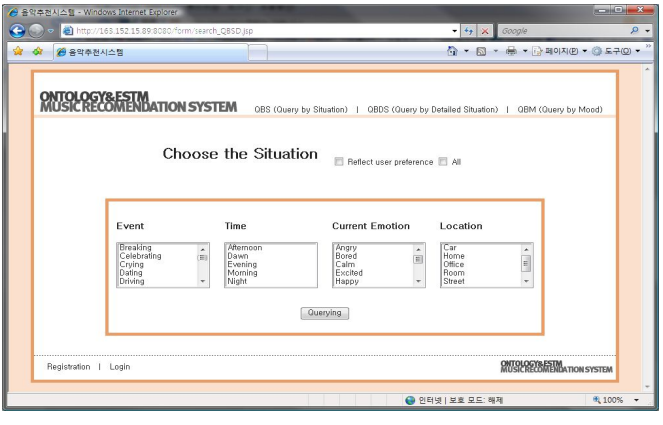


Fig. 3. Screenshots of query and result interface

In the experiment, we had about 30 participants. Some of them are musically trained and others are not. Participants were asked to fill out the questionnaire to collect suitable terms and definitions about the situation and mood. They were also requested to describe their own emotional state transition like current and desired emotions in the specific scenario. The description was based on one or more emotional adjectives such as happy, sad, angry, nervous, and excited those were collected from All Music Guide taxonomy [11]. Finally, the most frequently described adjectives were chosen to define the instances in the COMUS ontology.

After building the ontology, we performed an experiment to measure the level of user satisfaction using either our proposed COMUS ontology or AMG Taxonomy in our system. The procedure for the experiment was as follows.

1) The experimenter explained the procedure and the purpose of the experiment and demonstrated how to run our music recommendation system.
2) The participant should describe his profile (e.g., musical preferences) using web form interfaces such as buttons, textbox, checkbox and selection list.

3) All the participants were told to describe the situation, current emotion and their desired emotion or just select the predefined scenario using the query interfaces.
4) Then, the system returned recommended songs based on the ontology reasoning and the participant's profile. Then the participant judged which one was appropriate for their current emotion. Participants chose one of the 5 point rating scales (from 1 – strongly unsatisfied to 5 – strongly satisfied).
5) Finally, all the participants were asked to fill out a questionnaire.

Table 2. Participants' opinion with regard to the system using either COMUS ontology or AMG taxonomy

	1 (unsatisfied)	2	3 (neutral)	4	5
AMG Taxonomy	1	3	19	5	2
COMUS Ontology	0	2	4	16	8

As shown in Table 1, over 80% of the participants responded positively to the overall satisfaction of the system using ontology instead of AMG taxonomy. The result of the satisfaction ratings shows most of the users were satisfied with the query results recommended by the system.

Table 3. Participants' preferred emotional adjectives

	AMG Taxonomy					COMUS Ontology				
	1	2	3	4	5	1	2	3	4	5
Angry	4	5	9	8	4	1	6	11	9	3
bored	9	12	6	2	1	1	3	6	11	9
Calm	2	6	9	9	4	2	5	12	7	4
excited	3	4	6	8	9	0	3	6	8	13
Happy	4	8	10	6	2	0	2	2	9	17
Nervous	3	15	8	4	0	4	5	8	8	5
Peaceful	3	7	8	6	6	0	3	5	14	8
Pleased	6	7	13	3	1	0	1	5	9	15
Relaxed	6	7	12	3	2	1	4	12	6	7
sad	4	8	16	2	0	0	1	14	11	4
Sleepy	0	9	11	6	4	2	4	6	12	6

With regard to the satisfaction of the participant's preferred emotional adjectives depicted in Table 2, positive adjectives (such as happy and excited) are found more satisfactory by about 78% of the participants whereas ambiguous adjectives like nervous by 43% of the participants (in the case of using COMUS ontology).

4 Conclusion and Outlook

In this paper, we presented ontology based context model that is feasible and necessary for supporting context modeling and reasoning in music recommendation We modeled musical domain and captured low-level musical features and several musical

factors to describe music moods and music-related situations composed of time, location and subject to build ontology. We have constructed musical ontology based on the current music ontology as part of ongoing project on building intelligent music recommendation system. To show its feasibility, in addition, we set up a usage scenario and presented several queries for reasoning useful information from the ontology. We are currently working on an extended our reasoning model and ontology that support more precise music recommendation.

References

1. Birmingham, W., Dannenberg, R., Pardo, B.: An Introduction to Query by Humming with the VocalSearch System. Communications of the ACM 49(8), 49–52 (2006)
2. Rho, S., Han, B., Hwang, E., Kim, M.: MUSEMBLE: A Novel Music Retrieval System with Automatic Voice Query Transcription and Reformulation. Journal of Systems and Software 81(7), 1065–1080 (2008)
3. Oscar, C.: Foafing the Music: Bridging the semantic gap in music recommendation. In: Proceedings of 5th International Semantic Web Conference (ISWC) (2006)
4. Oscar, C., Perfecto, H., Xavier, S.: A multimodal approach to bridge the Music Semantic Gap. In: Semantic and Digital Media Technologies (SAMT) (2006)
5. Yves, R., Frederick, G.: Music Ontology Specification, `http://www.musicontology.com/`
6. Yves, R., Samer, A., Mark, S., Frederick, G.: The Music Ontology. In: Proceedings of the International Conference on Music Information Retrieval, ISMIR 2007, pp. 417–422 (2007)
7. Kanzaki Music Vocabulary, `http://www.kanzaki.com/ns/music`
8. MusicBrainz, `http://musicbrainz.org`
9. Protégé Editor, `http://protege.stanford.edu`
10. Grüninger, M., Fox, M.S.: The Role of Mariano Fernández López 4-12 Competency Questions in Enterprise Engineering. In: IFIP WG 5.7 Workshop on Benchmarking. Theory and Practice, Trondheim, Norway (1994)
11. All Music Guide, `http://allmusic.com`
12. W3C. RDF Specification, `http://www.w3c.org/RDF`
13. Francois, P., Daniel, C.: A Taxonomy of Musical Genres. In: Content-Based Multimedia Information Access Conference (RIAO) (2000)
14. Samer, A., Yves, R., Mark, S.: An ontology-based approach to information management for music analysis systems. In: Proceedings of the 120th AES convention (2006)
15. Klaus, R.S., Marcel, R.Z.: Emotional Effects of Music: Production Rules. In: Music and emotion: theory and research. Oxford University Press, Oxford (2001)
16. Song, S., Rho, S., Hwang, E., Kim, M.: Music Ontology for Mood and Situation Reasoning to Support Music Retrieval and Recommendation. In: ICDS 2009 (to appear, 2009)

Spatial Weighting for Bag-of-Visual-Words and Its Application in Content-Based Image Retrieval

Xin Chen[1], Xiaohua Hu[1,2], and Xiajiong Shen[2]

[1] College of Information Science and Technology, Drexel University, Philadelphia, PA, USA
[2] College of Computer and Information Engineering, Henan University, Henan, China
bruce.chen@drexel.edu, thu@ischool.drexel.edu

Abstract. It is a challenging and important task to retrieve images from a large and highly varied image data set based on their visual contents. Problems like how to fill the semantic gap between image features and the user have attracted a lot of attention from the research community. Recently, the 'bag of visual words' approach exhibits very good performance in content-based image retrieval (CBIR). However, since the 'bag of visual words' approach represents an image as an unordered collection of local descriptors which only use the intensity information, the resulting model provides little insight about the spatial constitution and color information of the image. In this paper, we develop a novel image representation method which uses Gaussian mixture model (GMM) to provide spatial weighting for visual words and apply this method to facilitate content based image retrieval. Our approach is a simple and more efficient compared with the order-less 'bag of visual words' approach. In our method, firstly, we extract visual tokens from the image data set and cluster them into a lexicon of visual words. Then, we represent the spatial constitution of an image as a mixture of *n* Gaussians in the feature space and decompose the image into *n* regions. The spatial weighting scheme is achieved by weighting visual words according to the probability of each visual word belonging to each of the *n* regions in the image. The cosine similarity between spatial weighted visual word vectors is used as distance measurement between regions, while the image-level distance is obtained by averaging the pair-wise distances between regions. We compare the performance of our method with the traditional 'bag of visual words' and 'blobworld' approaches under the same image retrieval scenario. Experimental results demonstrate that the our method is able to tell images apart in the semantic level and improve the performance of CBIR.

Keywords: Image representation, content based image retrieval, visual words, Gaussian mixture model, spatial weighting.

1 Introduction

Retrieving images from a large and highly varied image data set based on their visual contents is a very challenging problem. The content-based image retrieval (CBIR) has been studied for decades and many good approaches have been proposed [2, 3]. However, as Arnold mentioned, there is a 'semantic gap' between image features and

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 867–874, 2009.
© Springer-Verlag Berlin Heidelberg 2009

the user [2], in another word, similar visual features (such as similar color and shape) do not guarantee similar semantic meanings. The semantic gap between low-level image features and high level semantic concepts has been a major problem in CBIR.

Recently, the 'bag-of-visual-words' [6] approach exhibits very good performance in image categorization and semantic image retrieval across several well-known databases [4, 8, 14]. The underlying assumption of this approach is that, the patterns of different image categories can be represented by different distributions of microstructures (key-points). As an image document can be constantly represented as an unordered collection of key-points which contain rich local information, it can to some extent be regarded as a 'bag' of 'visual words'. Image patches containing key-points can be quantified based on affine invariant local descriptors [9, 11-13]. Sivic et al. further proposed the idea of assigning all the patch descriptors into clusters to build a 'vocabulary' of 'visual words' for a specific image set [6].

Inspired by the success of vector-space model of the text document representation, the 'bag-of-visual-words' approach usually converts images into vectors of visual words based on their frequency [4, 6]. Many effective text mining and information retrieval algorithms like *tf-idf* weighting, stop word removal and feature selection have been applied to the vector-space model of visual-words. Problems such as how vocabulary size and term weighting schemes affect the performance of 'bag-of-visual-words' representation are also studied in recent research works [4, 15].

Despite the success of 'bag-of-visual-words' in recent studies, however, there are two problems to be concerned. Firstly, since the 'bag of visual words' approach represents an image as an unordered collection of local descriptors, the resulting vector-space model provides little insight about the spatial constitution of the image. Secondly, as most local descriptors are based on the intensity information of images, no color information is used.

We have seen some works aiming at incorporating spatial information and color information in the 'bag-of-visual-words' model, such as dividing an image into equal-sized rectangular regions and computing visual word frequency from each region [4], using multi-scale spatial grids for locally order-less description of visual words [16] and using color SIFT descriptors [17]. However, to the best of our knowledge, there hasn't been any study combining visual-word features with the real spatial constitution of image content.

In this paper, we develop a novel image representation method which uses Gaussian mixture model to provide spatial weighting for visual words. The method has been applied to CBIR. As illustrated in Fig. 1, we firstly extract visual tokens from the images data set and cluster them into a lexicon of visual words. Then, we represent the spatial constitution of an image as a mixture of n Gaussians in the feature space and decompose the image into n regions. The spatial weighting scheme is achieved by weighting visual words according to the probability of each visual word belonging to each of the n regions in the image. The cosine similarity between spatial weighted visual word vector pairs is used as distance measurement between regions. The image-level distance is obtained by averaging the pair-wise distances between regions. We compare the performance of our method with the traditional 'bag of visual words' and the 'blobworld' approaches under the same image retrieval scenario. Experimental results demonstrate that our method is able to tell images apart in the semantic level and improve the performance of content-based image retrieval.

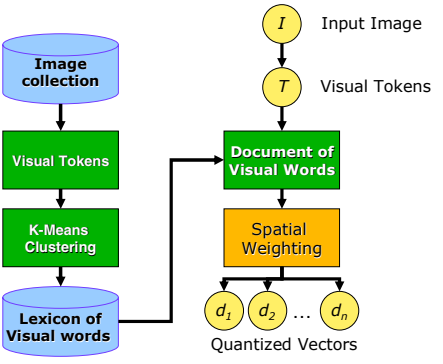


Fig. 1. The framework of proposed method

The remainder of this paper is organized as follows. In Section 2, we describe the procedure to generate a lexicon of the visual words for an image data set. In Section 3, we briefly review the process of using Expectation-Maximization (EM) algorithm to iteratively model the parameters of a mixture of Gaussians in the feature space and introduce our spatial weighting schemes. Section 4 reports the experimental results of the proposed method and compares our approach to the traditional 'bag of visual words' approach and the 'blobworld' approach. We conclude the paper in Section 5.

2 Lexicon of Visual Words of an Image Set

In our approach, we adopt the Difference-of-Gaussian (DoG) salient point detector [13] to detect salient points from images. The detection is achieved by locating scale-space extreme points in the difference-of-Gaussian images. The main orientations of salient points are determined by image gradient. After that, image patches containing the salient points are rotated to a canonical orientation and divided into 4×4 cells. In each cell, the gradient magnitudes at 8 different orientations are calculated. Finally, each salient point is described by a 128-dimensional SIFT descriptor. Compared to other local descriptors, the SIFT descriptor is more robust and invariable to rotation and scale/luminance changes [11]. In this paper, we name the extracted 128-dimension SIFT descriptors as ‘visual tokens’.

At this stage, although we have extracted visual tokens from images, these visual tokens are still too diverse to be analyzed. It is necessary to convert those variant visual tokens into uniform forms (visual words) to facilitate our analysis. Thus, we adopt the K-Means algorithm to cluster visual tokens into groups; the Euclidian distance is used as the distance measurement.

We use the R-Square (RS) as an indicator of appropriate cluster number. The RS is defined as the ratio of between-group sum-of-square over total sum-of-square. Generally, the RS value increases as the cluster number increases. On one hand, the larger RS value we have, the more patterns we are able to distinguish. On the other hand, the cluster number can not increase without limit due to the computational efficiency. After extensive experimental study, we take RS=0.7 as threshold and set the cluster number to be 1000, which is large enough to represent various images

while still computationally efficient. In this way, we establish a lexicon of 'visual words' whose vocabulary size is 1000, with each cluster center as a 'visual word'.

3 Gaussian Mixture Model for Spatial Weighting

The Gaussian mixture model (GMM) [12] has been used to model the feature space and the spatial distribution of images [5]. In this paper, based on GMM, we present a novel spatial weighting scheme for visual words as follows.

Firstly, each pixel is represented by 5-dimensional normalized feature vector, including 3-dimensional LUV color features plus 2-dimensional (x,y) position. In an image with m pixels, a total of m feature vectors: $\mathbf{y}_1,...,\mathbf{y}_m$ ($\mathbf{y}_j \in R^5$) will be obtained. Then, each image is assumed to be a mixture of n Gaussians in the 5-dimensional feature space and the Expectation-Maximization (EM) algorithm is used to iteratively estimate the parameter set of the Gaussians. The parameter set of Gaussian mixture is: $\theta = \{\mu_i, \Sigma_i, \alpha_i\}_{i=1}^n$, in which $\mu_i \in R^d$ is the mean of the i^{th} Gaussian; Σ_i denotes the $d \times d$ covariance matrix; while α_i represents the prior probability of the i^{th} Gaussian.

At each E-step of the EM algorithm, we estimate the probability of a particular feature vector $\mathbf{y}_j$ belonging to the i^{th} Gaussian according to the outcomes from the last maximization step (eq. 1)

$$p(i \mid j, \theta_t) = p(z_j = i \mid y_j, \theta_t) = \frac{p(y_j \mid z_j = i, \theta_t) p(z_j = i \mid \theta_t)}{\sum_{k=1}^{n} p(y_j \mid z_j = k, \theta_t) p(z_j = k \mid \theta_t)} . \tag{1}$$

In which z_j denotes which Gaussian $\mathbf{y}_j$ comes from and θ_t is the parameter set at the t^{th} iteration. At each M-step, the parameter set of the n Gaussians is updated toward maximizing the log-likelihood, which is:

$$Q(\theta) = \sum_{j=1}^{m} \sum_{i=1}^{n} p(z_j = i \mid y_j, \theta_t) \ln(p(y_j \mid z_j = i, \theta) p(z_j = i \mid \theta)) . \tag{2}$$

When the algorithm converges, the parameter sets of n Gaussians as well as the probability $p(i \mid j)$ are obtained. Based on the estimated GMM model, an image can be decomposed into n regions and $p_j^{\max} = \arg\max_i p(i \mid j)$ will indicate which region the given pixel point most likely belongs to (Fig. 2c).

The spatial weighting scheme is achieved by weighting visual words with regard to each region according to the location of the detected salient points. Supposing that local descriptors obtained from the salient point set $\{j_1, j_2, ..., j_M\}$ are assigned to

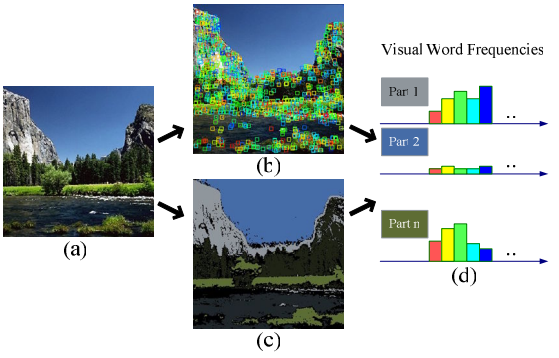


Fig. 2. The process of spatial weighting. (a) original image; (b) extracted visual words; (c) GMM modeling result; (d) spatial weighted visual word frequencies.

visual word V, then the summation of $p(i \mid j_k), k = 1,...,M$ will indicate the contribution of visual word V to region i. Therefore, the weighted term frequency of V with regard to region i can be defined as: $tf_V = \sum_{k=1}^{M} p(i \mid j_k)$ (Fig. 2d).

Supposing that $\boldsymbol{d_i}$ and $\boldsymbol{d_j}$ are two D-dimensional (D equals the vocabulary size of visual words) vectors of spatial weighted visual word frequencies, which come from region i and region j, respectively. Then the most natural way to measure the similarity between vectors d_i and d_j is using the cosine similarity (eq. 3).

$$Sim_{\cos ine}(d_i, d_j) = \frac{\mathbf{d}_i^t \mathbf{d}_j}{\|\mathbf{d}_i\| \|\mathbf{d}_j\|}. \tag{3}$$

Assuming that I_q is the query image and I_r is an image from the retrieval set. For each region in image I_q, we find its closest region in image I_r based on eq. 3. After that, the image-level similarity is obtained by taking the average of the pair-wise similarity between regions in I_q and their closest regions in I_r.

4 Experimental Results

In this section, we conduct a content-based image retrieval experiment based on the proposed image representation method and compare the experiment results with the 'bag-of-visual-words' and the 'blobworld' approaches.

The image dataset we used is composed of 8 categories, a total of 2,689 outdoor images from the LabelMe dataset [1]. In our experiment, we randomly select 1/6 images from each image category to build the lexicon of visual words. In total, we extract 175,535 visual tokens. At the retrieval stage, we use the selected 1/6 images as query images to retrieve images from the remaining 5/6 images in the data set. The retrieval results are ranked according to the similarity between the query image and all the images in the retrieval set following the similarity measurement in section 3.

For comparative study, we implement the basic 'bag of visual words' approach and the 'blobworld' approach. In recent study, there has been an intense focus on applying term weighting schemes (like *tf*, *idf*) to the 'bag-of-visual-words' feature vectors [4, 6]. Extensive study in [15] suggests that when the vocabulary size of visual words is around 1000, the *tf-idf* weighting performs best. Recall that in our approach, the vocabulary size of visual words for the data collection is D=1000, to make the experimental results comparable, we chose to compare our spatial weighting approach with the *tf-idf* weighted 'bag-of-visual-words' approach [4].

The 'blobworld' approach is another well-known image representation method, which simply represents images by the parameter sets of Gaussian mixture models. Following the method in [5], each coherent region is modeled as a multivariate Gaussian. After learning the parameters sets (that is, the mean vector $\boldsymbol{\mu}_i$ and the covariance matrix $\boldsymbol{\Sigma}_i$.) of Gaussians, the KL-divergence for multivariate Normal densities is used as the similarity measurement.

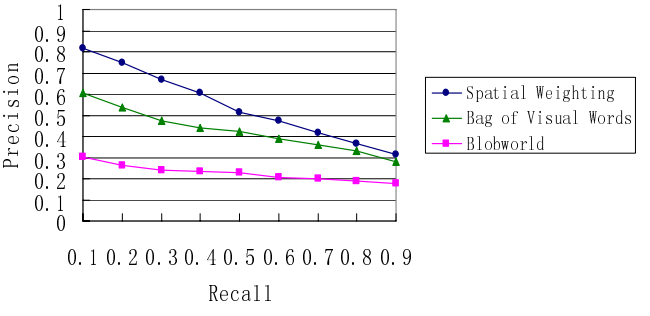
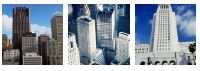


Fig. 3. Comparison of over-all precision-recall of our approach and the comparative approaches

Fig. 3 represents the over-all precision-recall of our approach, the 'bag-of-visual-words' approach and the 'blobworld' approach. As expected, our approach achieves high semantic consistency in CBIR and outperforms both two comparative approaches. Since the contents of different image categories are widely different, it is helpful to compare the performances in individual categories (a briefly description of the three selected categories is represented in Table 1.)

The precision-recalls of our approach (spatial weighting, short for S), the 'bag-of-visual-words' approach (short for V) and the 'blobworld' approach (short for B) in selected categories are shown in Fig. 4. In categories whose image compositions are highly varied and thus more complicated (such as 'coast' and 'tall-building'), our approach is about 10-20 percentage points better than the 'bag-of-visual-word' approach, while in the category whose image compositions are relatively uniform (like 'forest'), the 'bag-of-visual-word' approach performs as well as our approach. Compared to the other two approaches, the 'blobworld' approach works well only when the colors and image compositions are uniform.

Table 1. Description of three selected categories

Image Category	Coast	Forest	Tall-building
Number of Images	360	328	356
Examples			

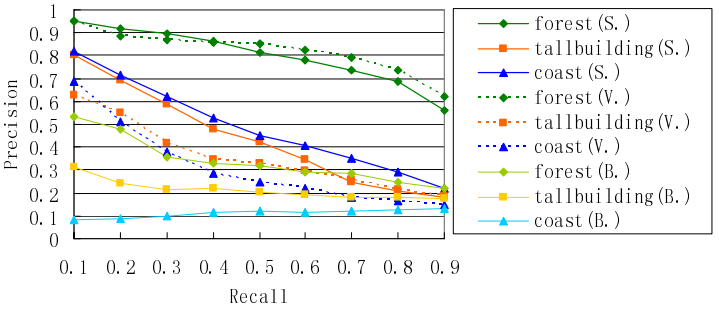


Fig. 4. Precision-recall of our approach and comparative approaches in different image categories

The experiment results suggest that, visual words from different kinds of regions may make the 'bag of visual words' noisy and thus less differentiable. Take the 'coast' images for example, the 'primary' information about sea and sand beach may be 'contaminated' by visual words from other 'inessential' parts like boats, buildings and coconut trees. Therefore, the significant improvement in our approach can be explained by the introducing of spatial weighting, which weights visual words according to actual spatial constitution of regions in images. Moreover, the experiment results also suggest that the Gaussian mixture model alone is insufficient to distinguish images which are highly varied in colors and compositions. However, the Gaussian mixture model is still able to provide enough information about the spatial constitutions of images.

5 Conclusions

Although recent studies about the 'bag-of-visual-words' have achieved many good results, the performance of visual words based image representation can be further improved if we properly weight the visual words according to the spatial constitution of image content. In this paper, the proposed spatial weighting method achieves high semantic consistency in content based image retrieval and outperforms the basic 'bag-of-visual-words' approach in image categories with high inner-variation. The experiment results demonstrate that the GMM model is suitable to model the spatial constitution of regions in images and the spatial weighting method can help us more accurately represent the content of images.

Acknowledgments. This work is supported in part by NSF Career grant (NSF IIS 0448023), NSF CCF 0514679.

References

1. Russell, B.C., Torralba, A., Murphy, K.P., Freeman, W.T.: LabelMe: a database and web-based tool for image annotation. MIT AI Lab Memo AIM-2005-025 (September 2005) (Revised on April 2007)
2. Smeulders, A.W.M., Worring, M., Santini, S., Gupta, A., Jain, R.: Content-based image rerieval at the end of the early years. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(12), 1349–1380 (2000)
3. Lew, M.S., et al.: Content-based multimedia information retrieval: State of the art and challenges. ACM Trans. Multimedia Comput. Commun. Appl. (2006)
4. Yang, J., Jiang, Y.G., Hauptmann, A.G., Ngo, C.W.: Evaluating Bag-of-Visual-Words Representations in Scene Classification. In: ACM SIGMM Int'l Workshop on Multimedia Information Retrieval (MIR 2007), Augsburg, Germany (September 2007)
5. Belongie, S., Carson, C.: Color- and Texture-Based Image Segmentation Using EM and Its Application to Content-Based Image Retrieval[C]. In: ICCV 1998, pp. 675–682 (1998)
6. Sivic, J., Zisserman, A.: Video Google: A Text Retrieval Approach to Object Matching in Videos. International Conference on Computer Vision, pp. 1470– 1477 (2003)
7. Csurka, G., Bray, C., Dance, C., Fan, L.: Visual categorization with bags of keypoints. In: Proc. of ECCV Workshop on Statistical Learning in Computer Vision, pp. 1–22 (2004)
8. Zhang, J., Marszalek, M., Lazebnik, S., Schmid, C.: Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study. International Journal of Computer Vision 73(2), 213–238 (2007)
9. Kadir, T., Brady, M.: Scale, Saliency and Image Description. International Journal of Computer Vision 45(2), 83–105 (2001)
10. Kadir, T., Zisserman, A., Brady, M.: An affine invariant salient region detector. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3021, pp. 228–241. Springer, Heidelberg (2004)
11. Mikolajczyk, K., Schmid, C.: A Performance Evaluation of Local descriptors. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 257–264 (2003)
12. Bilmes, J.A.: A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. technical report, Univ. of California, Berkeley (April 1998)
13. Lowe, D.: Distinctive Image Features from Scale-Invariant Key Points. International Journal of Computer Vision 60(2), 91–110 (2004)
14. Bosch, A., Zisserman, A., Muoz, X.: Scene Classification Using a Hybrid Generative/ Discriminative Approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(4), 712–727
15. Jiang, Y.-G., Ngo, C.-W., Yang, J.: Towards optimal bag-of-features for object categorization and semantic video retrieval. In: CIVR 2007, pp. 494–501 (2007)
16. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: Proc. CVPR (2006)
17. van de Weijer, J., Schmid, C.: Coloring Local Feature Extraction. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 334–348. Springer, Heidelberg (2006)

Item Preference Parameters from Grouped Ranking Observations

Hideitsu Hino, Yu Fujimoto, and Noboru Murata

School of Science and Engineering, Waseda University,
3-4-1 Ohkubo, Shinjuku, Tokyo 169-8555, Japan
{hideitsu.hino,fyu}@toki.waseda.jp, noboru.murata@eb.waseda.ac.jp

Abstract. Given a set of rating data for a set of items, determining the values of items is a matter of importance. Various probability models have been proposed to solve this task. The Plackett-Luce model is one of such models, which parametrizes the value of each item by a real valued preference parameter. In this paper, the Plackett-Luce model is generalized to cope with the grouped ranking observations such as movies or restaurants ratings. Since the maximization of the likelihood of the proposed model is computationally intractable, the lower bound of the likelihood which is easy to evaluate is derived, and the *em* algorithm is adopted to the maximization of the lower bound.

1 Introduction

Over the years, there have been a number of efforts to develop models and analytic techniques for item rankings or orderings of objects. So far, various models of ranking have been proposed according to the assumption of the underlying mechanism how rankings are generated. Bradley and Terry [1] have proposed a model in which each item I_i has a real valued parameter θ_i and the probability of being chosen item I_i over item I_j is given by $P(I_i \succ I_j) = \frac{\theta_i}{\theta_i+\theta_j}$, where $I_i \succ I_j$ means I_i is preferred to I_j. This model is based on Luce's choice axiom [2], and referred to as the Bradley-Terry-Luce model. The natural extension of the Bradley-Terry-Luce model is given by Plackett [3] and we refer to the model as the Plackett-Luce model henceforth. With the item preference level parameter $\theta = \{\theta_i\}_{i=1}^N$ for N items, in the Plackett-Luce model, the probability of ranking is defined as

$$P(I_{a(1)} \succ I_{(2)} \succ \cdots \succ I_{a(N)}) = \prod_{i=1}^{N-1} \frac{\theta_{a(i)}}{\sum_{j=i}^{N} \theta_{a(j)}}, \tag{1}$$

where $a(j)$ denotes the index of the item that occupies the j-th position in the ranking. In this model, greater value of θ_i implies that the item I_i is highly ranked.

In this paper, we generalize the Plackett-Luce model to cope with the grouped ranking observations, in which each of U judges (users, henceforth) rates N items

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 875–882, 2009.
© Springer-Verlag Berlin Heidelberg 2009

on a scale of 1 to M, $M \leq N$. We suppose there is a latent ordering in a set of the same rated items, but we only observe M groups of items which are divisions of N items. For example, when 7 items $I = \{I_1, \ldots, I_7\}$ are rated with a scale of $M = 3$, we get a grouped ranking observation from a user u such as $D_u = \{G_1^u, G_2^u, G_3^u\}$, $G_1^u = \{3, 5\}$, $G_2^u = \{2, 6, 7\}, G_3^u = \{1, 4\}$. Those observations are available from U individual users, i.e. we get a set of observations $\{D_u\}_{u=1}^U$. We assume that each of N items has preference level parameter θ_i such that $\theta_i > 0$, $\sum_{i=1}^N \theta_i = 1$, and our goal is to estimate the parameter $\theta = (\theta_1, \ldots, \theta_N)$ using only a set of the grouped ranking observations $\{D_u\}_{u=1}^U$.

2 Grouped Ranking Model

Model Description and the Likelihood Function
Suppose that U users independently give ratings of N items $I = \{I_1, \ldots, I_N\}$, ranging from 1 to M. The indexes of the most preferred items by user u are grouped into G_1^u, and that of next preferred items are grouped into G_2^u, and so on for $G_1^u, \ldots, G_M^u$. We denote all the groupings by user u as $D_u = \{G_1^u, \ldots, G_M^u\}$, where $G_m^u = \{i \in \{1, \ldots, N\} |\ I_i \in m\text{-th group}\}$. In this paper, D_u is called a grouped ranking observation henceforth.

When we need to consider the order of elements in group G_m^u, we use the action of a permutation π on G_m^u. For example, when we write $G_m^u = \{2, 6, 7\}$, we do not consider its order in the group. By the action of a permutation $\pi = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}$, we get $\pi(G_m^u) = (7, 2, 6)$ as an ordered index set, and the operations such as summation or product over its elements should be taken in the order of $\pi(G_m^u)$, such as $\sum_{i \in \pi(G_m^u)} \theta_i = \theta_7 + \theta_2 + \theta_6$.

Given U grouped ranking observations, we consider the problem of inferring the preference level parameters $\{\theta_i\}_{i=1}^N$ each of which is assigned to each item I_i. We can not make comparison between items within the same rated group in the observations, however, we assume that there is a latent ordering within a group and a user makes a grouped ranking observation as the following procedure:

1. Give a full ranking to all N items,
2. Divide N items into M groups without changing item ordering.

In our *grouped ranking model*, the given data set is composed of grouped ranking observations $\{D_u = \{G_1^u, \ldots, G_M^u\}\}_{u=1}^U$, and the ordering within each group is not observable.

Let us explain details of the model, and derive the likelihood function for our model. Suppose we have a grouped ranking observation $D_u = \{G_1^u, G_2^u, G_3^u\}$ for 7 items. We know that any item in G_1^u is preferred to any item in G_2^u by this user, however, there are no information for the order of items in the same groups. Suppose the user gave the ranking $I_7 \succ I_2 \succ I_6$ in G_2^u, for example. Since the items in G_1^u are already chosen and excluded, the probability that the items in G_2^u are chosen in this order from remaining items is given by $P(G_2^u, \pi) = \frac{\theta_7}{1-\sum_{j \in G_1^u} \theta_j} \frac{\theta_2}{1-\sum_{j \in G_1^u} \theta_j - \theta_7} \frac{\theta_6}{1-\sum_{j \in G_1^u} \theta_j - (\theta_7 + \theta_2)}$, where $P(G_m^u, \pi)$ denotes the joint

probability that the items in the group G_m^u are chosen in the order specified by π. Generalizing the above example, we get the general form of the joint probability as

$$P(G_m^u, \pi) = \prod_{i \in \pi(G_m^u)} \frac{\theta_i}{\sum_{n \geq m} \Theta_n^u - \sum_{i>j} \theta_j}, \tag{2}$$

where $\Theta_m^u := \sum_{i \in G_m^u} \theta_i$ is the sum of the parameters θ_i whose indexes belong to the group G_m^u. We call Θ_m^u a group parameter of the group $G_m^u \in D_u$ henceforth.

As the latent order in each group is expressed by the action of a permutation π on G_m^u, we let $\mathcal{S}(G_m^u)$ be the set of all possible $|G_m^u|!$ permutations of the items in the group G_m^u. Then the expectation of occurrence for the group G_m^u is a sum of the joint probabilities over possible latent orders

$$P(G_m^u) = \sum_{\pi \in \mathcal{S}(G_m^u)} P(G_m^u, \pi), \tag{3}$$

which is nothing but the marginalization of permutations. As a result, from equations (2) and (3), we get a log likelihood of a group G_m^u as

$$l(\theta, m, u) = \log \left(\sum_{\pi \in \mathcal{S}(G_m^u)} \prod_{i \in \pi(G_m^u)} \frac{\theta_i}{\sum_{n \geq m} \Theta_n^u - \sum_{i>j} \theta_j} \right). \tag{4}$$

Obviously, if every group consists of only one item and $M = N$, our model is reduced to the original Plackett-Luce model. By summing up $l(\theta, m, u)$ for all groups and for all users, we get the likelihood of the given data as

$$L(\theta) = \sum_{u=1}^{U} \sum_{m=1}^{M} \log \left(\sum_{\pi \in \mathcal{S}(G_m^u)} \prod_{i \in \pi(G_m^u)} \frac{\theta_i}{\sum_{n \geq m} \Theta_n^u - \sum_{i>j} \theta_j} \right). \tag{5}$$

It is worth noting that Huang *et al.*[4] have mentioned a similar generalization of the Plackett-Luce model, which they call the *multiple team comparison model*. They considered the ranking of the grouped items (teams) instead of items itself. In their model, the number of groups can vary depending on each user, which is denoted by M_u, and the group ranking probability is defined as $P(G_1^u \succ G_2^u \succ \cdots \succ G_{M_u}^u) = \prod_{m=1}^{M_u} \frac{\Theta_m^u}{\sum_{n \geq m} \Theta_n^u}$. The log likelihood of this model is given by

$$L_{\text{Huang}}(\theta) = \sum_{u=1}^{U} \sum_{m=1}^{M_u} \left(\log \Theta_m^u - \log \sum_{n \geq m} \Theta_n^u \right), \tag{6}$$

which looks similar to a lower bound of the likelihood (5) of our model which will be derived later in this section. The difference between this model and our proposed model is whether the model considers the orderings in the groups. In their model, in other words, they assume that groups of items are somehow predefined and users only give rankings of those groups.

Note that the maximization of the likelihood for our model is apparently a hard task, especially for a large size of items and/or users. The major source of the complexity is the existence of latent orderings in the model.

Lower Bound of the Log Likelihood

We derive a lower bound of the log likelihood function (5) in which no marginalization or no permutation is contained. The denominator in the expression (4) reflects the normalization of parameters θ_i in the sequential item selection. As an approximation of this normalization, we replace the denominator in (4) by $\sum_{n \geq m} \Theta_n^u$. With this replacement, we get a lower bound of (4) as

$$\underline{l}(\theta, m, u) = \log \left(\sum_{\pi \in \mathcal{S}(G_m^u)} \prod_{i \in \pi(G_m^u)} \frac{\theta_i}{\sum_{n \geq m} \Theta_n^u} \right) = \log \left(|G_m^u|! \prod_{i \in G_m^u} \frac{\theta_i}{\sum_{n \geq m} \Theta_n^u} \right)$$
$$= \sum_{i \in G_m^u} \log \theta_i - |G_m^u| \log \sum_{n \geq m} \Theta_n^u + \log \left(|G_m^u|! \right) \quad \leq \quad l(\theta, m, u). \tag{7}$$

Note that by replacing the denominator of (4) in such a way that it does not contain terms depending on the permutation π, the marginalization in (4) is reduced to a positive constant factor $\log(|G_m^u|!)$. For notational simplicity, we omit this constant factor henceforth. By summing up $\underline{l}(\theta, m, u)$ for all groups and all users in the observations, we get a lower bound of the log likelihood function of the given data as

$$\underline{L}(\theta) = U \sum_{i=1}^{N} \log \theta_i - \sum_{u=1}^{U} \sum_{m=1}^{M} |G_m^u| \log \sum_{n \geq m} \Theta_n^u \quad \leq \quad L(\theta). \tag{8}$$

As $L(\theta) \geq \underline{L}(\theta)$ holds, we can expect that maximization of $\underline{L}(\theta)$ indirectly leads to a maximization of $L(\theta)$. We note that for the original Plackett-Luce model, [5] gave a lower bound of the likelihood and proposed an iterative algorithm to maximize the function with respect to the item preference level parameter θ.

We can now apply any non-linear optimizer to maximize $\underline{L}(\theta)$ with respect to θ, however, still direct optimization of $\underline{L}(\theta)$ may cause two possible problems. The first problem is computational complexity. As the lower bound is a non-linear function of θ, its complexity increases with the number of items N. The second problem is the need for the remaximization of the likelihood when the new user join the system with new ranking data. In the next section, we will propose an alternative approach to estimate the item preference parameter. As is clear from its formulation, the computational time of the algorithm, which will be derived in the next section, mainly depends on the number of users U, and scales linearly with U.

3 Algorithm

Remember that our motivation of maximizing the likelihood is to find the item preference level parameter θ which is consistent with the observations as much

as possible. With this notion, we decompose the second term of $\underline{L}(\theta)$ to U independent optimization problems with respect to group parameters Θ^u_m, $\Theta^u_m \geq 0$, $\sum_{m=1}^M \Theta^u_m = 1$ which correspond to the given grouped ranking observation $D_u = \{G^u_1, \ldots, G^u_M\}$:

$$\min_{\{\Theta^u_m\}} \sum_{m=1}^{M} |G^u_m| \log \sum_{n \geq m} \Theta^u_n, \quad \text{subject to} \quad \Theta^u_m \geq 0, \ \sum_{m=1}^{M} \Theta^u_m = 1. \tag{9}$$

These problems are relatively small-sized optimization problems with linear constraints, and efficiently solved with arbitrary nonlinear optimizers.

We will show that the problem of finding the optimal parameter θ which is most consistent with the observations is solved by the *em* algorithm in the information geometry literature[6]. The solutions of the optimization problems (9) can be seen as incomplete observations of the item preference level parameter. As the item preference parameters $\{\theta_i\}$ are constrained to be positive and add up to 1, $\theta = \{\theta_i\}_{i=1}^N$ forms a manifold known as the standard $(N-1)$-simplex Δ_{N-1}. The solution $\{\hat{\Theta}^u_m\}_{m=1}^M$ defines a submanifold $\mathcal{D}_u := \{\theta | \sum_{i \in G^u_m} \theta_i = \hat{\Theta}^u_m\}$ of Δ_{N-1} (Figure 1(a)). The optimal parameter θ is a point in Δ_{N-1} that is the nearest to all the submanifolds $\{\mathcal{D}_u\}_{u=1}^U$ in terms of Kullback-Leibler (KL) divergence $KL(\theta, \theta') := \sum_{i=1}^N \theta_i \log(\theta_i / \theta'_i)$. The *em* algorithm gradually minimizes the KL divergence by repeating the *e*-step and *m*-step alternately.

Suppose we have an estimated parameter $\theta(t)$ after t times iterations. In the *e*-step, we find the points $\hat{\theta}^u(t)$ on the submanifolds $\mathcal{D}_u$ which are nearest in terms of KL divergence from the previous estimate $\theta(t)$(Figure 1(b)). That is, $\hat{\theta}^u(t) := \arg\min_{\theta' \in \mathcal{D}_u} KL(\theta', \theta(t))$. This procedure is called the *e*-projection and written as

$$\hat{\theta}^u_i(t) := \frac{\theta_i(t)}{\sum_{j \in G^u_{m|i}} \theta_j(t)} \hat{\Theta}^u_{m|i}, \quad (i = 1, \ldots, N, \ u = 1, \ldots, U) \tag{10}$$

where $G^u_{m|i}$ denotes the group to which I_i belongs, and $\hat{\Theta}^u_{m|i}$ denotes the corresponding group parameter. In the *m*-step, we find the point $\theta(t+1)$ on Δ_{N-1} which minimizes the sum of KL divergences from the points $\hat{\theta}^u(t)$ on the submanifolds $\mathcal{D}_u$ (Figure 1(c)). That is, $\theta(t+1) := \arg\min_{\theta'} \sum_{u=1}^U KL(\hat{\theta}^u(t), \theta')$. This procedure is called the *m*-projection and written as

$$\theta_i(t+1) := \frac{1}{U} \sum_{u=1}^{U} \hat{\theta}^u_i(t), \quad (i = 1, \ldots, N). \tag{11}$$

Iterating these *e*- and *m*-steps until neither $\theta(t)$ nor $\hat{\theta}^u(t)$ do not change, we get (locally) optimal parameter $\theta(t)$. To save space, we omit details of calculus that show the *e*- and *m*-projection , i.e. minimization of sum of KL divergences from submanifolds to a point θ are described as (10) and (11) respectively. Figure 1 shows these procedures in the case of $N = 3$, $U = 3$, $M = 2$. The *em* algorithm for our model is summarized as follows:

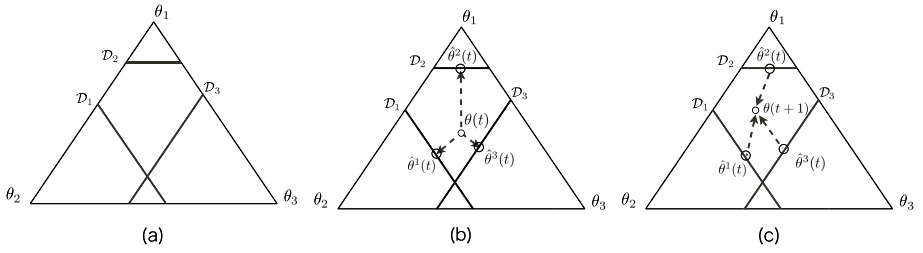


Fig. 1. (a): Each observation defines submanifold $\mathcal{D}_u$ of Δ_{N-1}. (b): Find the nearest points $\hat{\theta}^u(t)$ from the previous estimate $\theta(t)$ to each submanifold (e-step). (c): Find the nearest point $\theta(t+1)$ from each submanifold (m-step).

Algorithm 1 (The *em* algorithm for grouped ranking model)

input *grouped ranking observations* $\{D_u = \{G_1^u, \ldots, G_M^u\}\}_{u=1}^U$.
initialize *choose initial parameter* $\theta(0)$*, and solve the optimization problems (9) to get* U *sets of grouped parameter values* $\{\hat{\Theta}_m^u\}_{m=1}^M$.
repeat *from* $t = 0$*, until convergence*
  e**-step** *update* $\hat{\theta}^u(t)$ *by the e-projection (10).*
  m**-step** *update* $\theta(t)$ *by the m-projection (11).*
output *converged parameter* $\theta(t)$.

4 Experiments on Preference Parameter Estimation

Synthetic Data
We apply the proposed algorithm to synthetic data to show how the proposed algorithm works. We fixed the number of groups to 5, and the number of items N to 10 for simplicity. We vary the number of users as $U = 10, 20, 50, 100$, and show how the number of observations affects to parameter estimation. We randomly choose 100 true parameters for data generation and averaged the KL divergences of the estimated parameter from the true parameter.

The data are generated as described in Section 2. We first set the true item preference parameter $\theta^* = (\theta_1^*, \ldots, \theta_{10}^*)$. Then each user $u = 1, \ldots, U$ makes full rankings, and divides N items into $M = 5$ groups without changing item ordering. To these U grouped ranking observations, we apply our proposed algorithm and get the estimated parameter. For each true parameter value and for each user size, we generate grouped ranking observations 100 times. In Table 1, we show the mean and standard deviation of the KL divergence of the estimated parameters from the true parameters. The right most column of the Table 1 is the average of the KL divergence of the uniform distribution from the true parameters as a baseline value. We can see that our algorithm can get closer to the true parameter in average.

Table 1. KL divergence from true parameter

$U = 10$	$U = 20$	$U = 50$	$U = 100$	Baseline
0.198 ± 0.099	0.179 ± 0.091	0.157 ± 0.078	0.151 ± 0.076	0.362

We also show how the average KL divergence from the true parameter decreases and how the lower bound of the log likelihood $\underline{L}(\theta)$ increases as the algorithm progresses. Figure 2 shows the change of the average KL divergence of the estimated parameter from the true parameter (solid line), and the average lower bound of the likelihood (dashed line) for the estimated parameter when the number of users is fixed to 100. We can see that with only a few iterations, the algorithm converges and successfully finds estimates of the parameters.

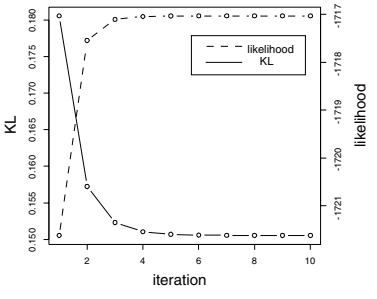


Fig. 2. KL divergence and $\underline{L}(\theta)$

Table 2. Comparison of the rating prediction accuracy. The numbers in the column "1st" and "2nd" are % of the correct rating predictions by the largest and the second largest parameters.

model \ item rank	1st [%]	2nd [%]
Grouped ranking	61.99	57.14
MTC	57.14	50.71

Real-world Data: Application to MovieLens Data

We show a preliminary result of the application of preference level parameter estimation to find the most commonly preferred movies in the MovieLens data set. The MovieLens data set consists of 100,000 ratings ranging from 1 to 5 for 1682 movies by 943 users. We first selected 100 most frequently rated movies, and then extracted 554 users who rated more than 20 of the 100 movies. The evaluation criterion for the parameter estimation is the accuracy of the rating predictions by highly estimated parameter values. We get estimated parameter values for 100 movies, and we suppose that the movies with the highest and the second highest parameter value may be rated as 5 (very good) by most of the users. That is, the accuracy of the rating prediction means that the frequency of the user's high ratings for the highest and the second highest estimated movies. We set aside one user's rating data and estimate the preference level parameter using the rest of the data. If the user set aside gave rating 5 to the movie which has the highest and second highest parameter value, we count this case as a positive case, and if the user have given rating 4 or below, we count as negative case. To examine the appropriateness of the model, we also implemented the log likelihood function (6) of the multiple team comparison (MTC) model proposed in [4], and maximized the likelihood to find the optimal parameter. Table 2 shows

the accuracy of the rating predictions by estimated parameter of our grouped ranking model and the MTC model. From Table 2, we see that the highest ranked movie we found by our model is a movie to which more than 60 % of users have given the highest rating. We can also see that the ability of our model to predict the item ratings by the estimated parameter is superior to that of the MTC model when applied to the MovieLens data set, and we can infer that the movie rating mechanism in this data set is closer to our grouped ranking model than the MTC model as expected.

5 Conclusion

In this paper, we proposed a novel model for grouped ranking observations parametrized by item preference level parameter. We derived the likelihood function of the parameter, and gave a lower bound of the function which is easy to estimate. We also proposed the *em* algorithm to find the item preference level parameter which is the most consistent with observations. We conducted small numerical experiments both for synthetic and real-world data. From the result for synthetic data, we can conclude that the proposed algorithm can reduce the KL divergence from the true parameters by iterative optimization, and from the result of real-world data, we got promising accuracy to predict the most popular items.

We suggested that we can apply our model and algorithm to recommender systems, however, it is not able to make personalized recommendations yet. To make a recommender system with the proposed model, we think we need the mixture of ranking models and the way to combine the multiple preference parameters. The work in this direction is now in progress, and it will enable us to compare other recommender systems with our method.

References

1. Terry, M., Bradley, R.A.: The rank analysis of incomplete block designs: I. the method of paired comparisons. Biometrika 39, 324–345 (1952)
2. Luce, R.D.: Individual Choice Behavior. John Wiley & Sons, Inc., New York (1959)
3. Plackett, R.L.: The analysis of permutations. Applied Statistics 24(2), 193–202 (1975)
4. Huang, T., Weng, R.C., Lin, C.: Generalized Bradley-Terry models and multi-class probability estimates. J. Mach. Learn. Res. 7, 85–115 (2006)
5. Hunter, D.R.: MM algorithms for generalized Bradley-Terry models. The Anaals of Statistics 32(1), 384–406 (2004)
6. Amari, S.: Information geometry of the EM and *em* algorithms for neural networks. Neural Networks 8(9), 1379–1408 (1995)

Cross-Channel Query Recommendation on Commercial Mobile Search Engine: Why, How and Empirical Evaluation

Shunkai Fu[1], Bingfeng Pi[1], Ying Zhou[1], Michel C. Desmarais[2], Weilei Wang[1], Song Han[1], and Xunrong Rao[1]

[1] Roboo Inc.
[2] Ecole Polytechnique de Montreal

Abstract. Mobile search not only inherits some features of traditional search on PC, but also has many of its own special characteristics. In this paper, we firstly share some unique features about mobile search and discuss why vertical search is preferred. Providing multiple vertical searches is proved to be convenient to users but causes some minor problem as well. This plays as the initiative for us to propose cross-channel query recommendation. Secondly, we briefly introduce how to realize the cross-channel recommendation effectively and efficiently online. Finally, we analyze the performance of the proposed method from three different but related metrics: expected effect, off-line evaluation and on-line evaluation. All three studies together indicate that the proposed cross-channel recommendation is quite useful. Being the first study about query recommendation on mobile search, it is believed that the findings, proposed solution and collected feedback as presented here will be beneficial to both researchers and industry companies while considering how to provide better mobile search service.

Keywords: Cross query recommendation, mobile search, empirical evaluation.

1 Introduction

Mobile search is an emerging and promising application in China currently considering the potential huge user population, over 600-million Chinese are consuming mobile communication service. Being a start-up company, we are growing up quickly in the Chinese market by providing easy-to-use mobile search service. A large amount of first-hand usage data allow us to understand Chinese search behavior, then know how to do better, and finally evaluate how well we are doing with the feedback collected.

In this paper, we refer to the traditional search over computer as *PC search*, and the search over a hand-held device as *mobile search*. We discuss in the following paragraphs why the two kinds of search have to be treated in different way.

Although the fundamental search algorithms and architecture are similar for both PC and mobile search on the backend, more challenge actually is imposed on the latter one, and they can be summarized briefly as below:

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 883–890, 2009.
© Springer-Verlag Berlin Heidelberg 2009

1. Slow wireless access and processing speed: Neither GSM, GPRS nor EDGE as available in China has competitive transmission speed compared with typical wired Internet access, like ADSL. Meanwhile, the computing capacity, in terms of CPU and memory, is quite limited on mobile devices;
2. Expensive connection fee: Compared with wired connection, the charge of wireless transmission flow is quite high, more than 1 US dollar per MB;
3. Small screen: Quite limited information can be accessed by users without extra scrolling down operation (refer to Figure 1 for an example);
4. Difficult input: Typing on a standard 12-key cell phone is not convenient. Our analysis indicates that the average query on our mobile search engine is less than 4 Chinese characters, which is much shorter than that on a typical PC search engine and is consistent with the finding as reported in [1, 2];
5. Mobile search is brand new for many: One telephone survey made by us on a random selected group of users indicates that most mobile search users even never use PC search before, which is consistent with the fact that there are over 600 millions of wireless communication users but only 200 millions of PC users. The survey also tells us that many experienced PC searchers neither have long history of accessing mobile search;
6. Much lower average education level for mobile search users: The survey mentioned above indicated this point as well;
7. One important way of amusement: Mobile phone is becoming one of the most favorite pastimes in China because of its portability and intelligent functions available. Looking for fun is one important goal for those who are using our mobile search service, as indicated by our search log data.

Figure 1(1) is the home page of Roboo mobile search engine (http:// wap. roboo. com). Several vertical channels are provided in addition to the page search, e.g., image, ring, MP3, short message, game, software etc. Here, we refer to those non-page channels as vertical channels considering they are designed for a specific kind of resource. It is built on the following points:

1. Page search is not enough for mobile search. Given current keyword-based relevance model and the fact that there are many duplicated or near-duplicate pages [1], a mobile searcher may be hard to find what s/he wants within the long list of results, which becomes worse due to the small screen slow processing rate;
2. Different from PC search, looking for fun is one important initiative for those mobile search users currently. Reading some news, playing a while of game, or browsing some images is usual pastime way for most mobile phone owners while they are waiting for bus, train, plane, etc. Separating these resources can assist the users to find what they prefer in an easier way;
3. Slow access speed and small screen are critical factors for the necessity of these vertical channels. Again, asking the users to scroll down to locate what they need among a long list of page search result is not convenient. The poor transmission speed will make it worse since turning to the next page requires additional connection request. Indexing different resources with independent vertical channel is one solution for this problem since it allows users to find MP3 or image regarding to someone, for example, in a direct way;

4. Multimedia resources, like image, video and music, are welcome by current mobile search users, at least when they are facing mobile device. Our search log indicates that over 70% of the total search flow comes from non-page channels, which makes us more confident about the decision.

We have covered so far the necessary background about mobile search as well as the search service provided by us. Although query recommendation are discussed by many research work [3,4,5], and can be found on most popular commercial search engines, like Google and Baidu (http://www.baidu.com), no effort, to our best knowledge, ever be spent on cross-channel recommendation and, more importantly, its importance on mobile search. Therefore, in this paper, we discuss the necessity of providing cross-channel recommendation, how to do that and expected outcome in Section 2. Then, in Section 3, we demonstrate the effect of our job based on log data. Section 4 gives some concluding remarks.



Fig. 1. A series of examples about Roboo mobile search engine and snapshot about cross-channel recommendation as discussed. (1) The primary user interface of Roboo mobile search engine; (2) Results as retrieved about query "Andy Liu" in page search channel; (3) Recommend an appropriate channel, different from the original one (highlighted with red rectangle); (4) Related resources are retrieved and presented if the user agrees and clicks the recommendation in (3); (5) Recommend more related resource about a target (highlighted with red rectangle); (6) Video results retrieved corresponding to the recommendation in (5).

2 Cross-Channel Recommendation

2.1 Why We Need It

Although the design of vertical channels allows for easier search and provides convenience for mobile searchers, it brings some problems as well:

1. Users search for some target within a wrong channel. For example, we find query of "images of Andy Liu" (one famous star) in the MP3 channel, and "songs of Andy Liu" in the image channel. This may happen on new users who are not familiar with our product, or happen accidentally due to mistaken operation. Though it occupies a small portion of the total query in the corresponding channel, it has to be paid with enough attention because we believe user experience will never be over emphasized, especially on the mobile search engine;
2. There are several related dimensions of information about one specific object. For instance, given a query "Andy Liu" in page channel, it indicates that the searcher is interested with related information about him. Typically, news relating to the queried object will be retrieved and ranked by some algorithm, with title and summary presented (Figure 1(2)). Actually, regarding to this star, there exist more resources about him, including images, MP3, ring, or video, but they are normally mixed in the results, and the searchers have to go through the results manually to find something interesting.

To solve these two problems, we propose cross-channel recommendation which may help us to achieve the following benefits:

1. Giving timely appropriate tips to searchers while they are searching in a wrong channel. For example, given the query "images of Andy" in MP3 channel, no result will be found, but we would ask the searcher "if you are searching for Andy's images." (See Figure 1(3), the part highlighted in red. How it works will be introduced in Section 2.2) If this recommendation is clicked, the user will be transferred to the image channel, and the related images about Andy will be retrieved and presented to the user (Figure 1(4)). We won't implicitly do this for the user, but asks her/him explicitly, considering that (1) the users may make some mistaken operations sometimes, and our recommendation allows them to fix the error with no cost (i.e., no more typing); (2) this reminding helps our users to get familiar with our product and avoid the same mistake next time; (3) algorithm will not always work correctly, so the recommendation is better to be confirmed by the user before it is processed. This may save time and money for the users by preventing non-appropriate service and transmission request; and (4) it shows our respect and emphasis towards the users;
2. Allowing more channels to be introduced to the users, providing them greater satisfaction as well as gaining more PV (page-visit). Given a query "Andy" in image channel, other than the images retrieved, we will ask "Interested with Andy's related video?" as shown in Figure 1(5). If this recommendation is accepted by the users (and be clicked), the related video resources about this star is retrieved and presented (Figure 1(6)). By this way, the users (1) have chance

to access more relevant information; (2) can browse the related information easily since it is well constructed; (3) don't have to re-enter the query again and again since everything is one-click away; (4) are able to control the possible cost by rejecting any recommendation not necessary to them;

3. The combination of the above two aspects will greatly increase the user experience since (1) mistaken operation is detected automatically and correct recommendation is presented in a friendly manner, and (2) more relevant information is recommended to searchers without requesting extra typing, just one-click away. In short, user-centered design is believed to be the most important factor for the success of a mobile search engine.

2.2 Algorithms for the Cross-Channel Recommendation

How to realize the cross-channel recommendation in a quick and effective manner online will be addressed in this section. The overall procedure is divided into the following steps:

1. Indexing queries as they appear in each channel's search log separately, by filtering out those queries with no results. New queries can be added into their corresponding channel's index online;
2. Given $< Q, Ch >$, i.e. query Q submitted in a specific channel Ch, do a search over each channel's query index as prepared in step 1;
3. If there is not exact match, go to step 4; else, $Q_{rec} = Q$. Then, if Q appears in more than one channel, select the channel with highest frequency of this query; otherwise, directly go to step 6.;
4. Filter out recommended candidates with frequency lower than threshold value θ_1 and those with similarity score smaller than θ_2;
5. Randomly select a recommended query as Q_{rec} if there is at least one match by the end of step 4. Otherwise, $Q_{rec} = null$;
6. If $Q_{rec} \neq null$, retrieve Q_{rec}'s corresponding channel label, Ch_{rec}, to compose the final recommendation $< Q_{rec}, Ch_{rec} >$. On the user interface, this recommendation appears as "If you are looking for Q_{rec}'s Ch_{rec}";
7. When "If you are looking for Q_{rec}'s Ch_{rec}" is clicked, a search about Q_{rec} is done in channel Ch_{rec}, and the retrieved results are presented to the searcher, see Figure 1(3) and Figure 1(5).

From this brief description, it can be seen that two kinds of information are necessary for the cross-channel query recommendation:

1. Search log data. It records queries and their corresponding frequency;
2. The similarity between the active query (i.e. the one being studied) with any other queries existing in the search log. Vector space model (VSM) is used here with cosine similarity as the basic measure of the distance between a pair of queries. Although it is known that even two queries may share terms, they may

be using the term in different contexts [6], we still depend on this measure considering that (a) it is quite simple; (b) the computation is fast enough for online application; (c) we can filter out too marginal matches by increasing θ_2; (d) our test indicates that it works well in most cases.

2.3 More Features about the Proposed Solution

Besides the primary procedure covered above, some not-so-obvious features have to be introduced as well since they are there to ensure the success of this solution:

1. Queries with no results are filtered out firstly in each channel's query log, which is necessary to avoid any fruitless recommendation;
2. Exact match is considered with highest priority. This maximizes the probability that the recommendation will be accepted;
3. Ties due to multiple exact matches are solved by choosing the more frequent one. For example, if a query is found in both image and video channel, but more frequent in previous one, image channel will be suggested with priority;
4. Randomly selecting one channel when exact match is not found, but there are several similar ones. Random decision has two obvious advantages, (a) each channel has fair opportunity to be recommended; (b) the risk of wrong recommendation is decreased.

However, due to the different releasing date for different channel, the volume of queries of different channel may vary greatly. Therefore, bias still exists even after random selection is applied: channels with longer operating history are more likely to be recommended.

3 Empirical Evaluation

Since it was released online, the cross-channel recommendation has been operating for about five months. Our evaluation covers the following three aspects:

1. Expected effect: Given the current algorithm, what the expected response will be under different condition given the current algorithm?
2. Off-line evaluation: Manual voting of recommendation;
3. On-line evaluation: Assessment of recommendations using log data.

Expected effect

Different from the discussion in Section 2.3, here, we care more about the possible performance from the user side: if it only works on the right time, i.e. recommendation should appear only when there is indeed related resource about the query in a different channel. Based on the description and discussion in Section 2.2 and Section 2.3, we know that, given a query in a channel, recommendation is presented only when exactly or similarly matched query is found in a different channel. By filtering out queries with no results or with too low frequency, the recommendation is expected to be reasonable and useful.

Off-line evaluation

When the recommendation is made and presented to the users, we validate if the suggested query Q_{rec} and the new channel, Ch_{rec} , are appropriate. To do this evaluation, we randomly choose 200 queries with exact matched results, and 200 with similar matches from the query log. Three professionals are employed to vote on the suggestions: < Q_{rec} , Ch_{rec} >. If they think the recommendation is appropriate, "Agree" it; if not, "Disagree" it; if not sure about the result, mark as "No idea". If there are at least two "Agree", "Disagree" or "No idea" with the recommendation, the result is obvious; if there is one "agree", one "disagree" and one "No idea", the result is viewed as "No idea" as well.

From Table 1, we see that 187 out of 200 (about 93.50%) recommendations are thought as appropriate when "exact match" is satisfied. The same index decreases to 69.50% (139 out of 200) for "similar match". This exactly reflects that (a) our method is useful to recommend more dimensions of information to the searchers, and (b) the similarity measure (VSM) used here that requires future work.

Table 1. Breakdown of the voting assessment of 400 randomly selected queries by three persons

	Agree	Disagree	No idea
Exact match	187 (out of 200)	12 persons	1 person
Similar match	139 (out of 200)	49 persons	12 persons
Summary	326 (out of 400)	61 persons	13 persons
Percentage	81.50%	15.25%	3.25%

On-line evaluation

Large amount of query log data is accumulated for the analysis of its real performance. We initially considered the following measures for on-line evaluation:

1. The frequency of cross-channel recommendation is clicked by the searchers when they are presented given a query;
2. The frequency of clicking happened within the results retrieved by the cross-channel recommended queried.

However, we afterwards realized that they are not as suitable as expected. Firstly, although the first measure could directly indicate if the recommendation is accepted or not, there are at least three factors ignored here: (1) The searchers don't notice the recommendations appearing below the retrieved results; (2) Even it is seen, the users may not click it since the results retrieved may be satisfying to them; (3) The contribution may not be measured completely if it is clicked or not since the presence of the recommendation itself may impress the searchers and keep them on our search engine. So, we can't just conclude that the recommendations are not welcome if they are not clicked. Secondly, with a similar inference, we could not depend on if the results retrieved are clicked or not to evaluate the recommendation.

Even so, some remarks are still shared based on our study of query log:

1. The recommendation works when there is no result found in some channel given a query. The user will click the suggestion, and be led to a new channel with related information presented, as the example of Figure 1(3)(4);
2. The recommendation also works even when there are results retrieved given a query in a channel. The recommendation will trigger a new search, and take the user to a new channel, where some results are clicked;
3. The absolute amount of times of clicking on the cross-channel recommendation is increasing, about 50% more per month;
4. With most recent observations, it is estimated that about 8.0% of our monthly PVs come directly or indirectly from the cross-channel recommendation.

4 Conclusion

In this paper, we start with mobile search as the big background, along with some discussions of its special characteristics based on our experience. Then, the discussion is narrowed down to our own mobile search engine, a popular one in China currently. Although our product strategy is proved to be suitable for Chinese mobile searchers, it also causes some minor problem. To fix this problem, a simple cross-channel query recommendation solution is proposed on the basis of (1) our understanding of user habit, (2) large scale of query log collected till now, and (3) efficient and effective strategy. The overall procedure is clearly described, and our evaluation covers (1) expected effect, (2) off-line evaluation and (3) on-line evaluation. All of these work together to present readers a comprehensive picture about how it works and how well it performs.

Some future work is deemed. We may do more research to find a finer model to measure the similarity between queries. Besides, more log data is required for more comprehensive on-line performance evaluation.

References

1. Fu, S., Pi, B., Han, S., Guo, J., Zou, G., Wang, W.: User-centered solution to detect near-duplicate pages on mobile search engine. In: Proceedings of SIGIR Workshop on Mobile IR (MobIR), Singapore (2008)
2. Kamvar, M., Baluja, S.: Query suggestions for mobile search: Understanding usage patterns. In: Proceedings of the SIGCHI Conference on Human Factors in Computing (CHI) (2008)
3. Zhang, Z., Nasraoui, O.: Mining search engine query logs for query recommendation. In: Proceedings of 15th International World Wide Web Conference, W3C (2006)
4. Yates, R.B.: Query usage mining in search engines. In: Scime, A. (ed.) Web Mining: Applications and Techniques, Idea Group (2004)
5. Wen, J., Nie, J., Zhang, H.: Clustering user queries of a search engine. In: Proceedings at 10th International World Wide Web Conference, W3C (2001)
6. Sahami, M., Heilman, T.: Mining the Web to determine similarity between words, objects, and communities. In: Proceedings at 22nd ICML, Bonn, Germany (2005)

Data Mining for Intrusion Detection: From Outliers to True Intrusions

Goverdhan Singh[1], Florent Masseglia[1], Céline Fiot[1], Alice Marascu[1], and Pascal Poncelet[2]

[1] INRIA Sophia Antipolis, 2004 route des lucioles - BP 93, FR-06902 Sophia Antipolis, France
First.Last@sophia.inria.fr
[2] LIRMM UMR CNRS 5506,161 Rue Ada, 34392 Montpellier Cedex 5, France
poncelet@lirmm.fr

Abstract. Data mining for intrusion detection can be divided into several sub-topics, among which unsupervised clustering has controversial properties. Unsupervised clustering for intrusion detection aims to i) group behaviors together depending on their similarity and ii) detect groups containing only one (or very few) behaviour. Such isolated behaviours are then considered as deviating from a model of normality and are therefore considered as malicious. Obviously, all atypical behaviours are not attacks or intrusion attempts. Hence, this is the limits of unsupervised clustering for intrusion detection. In this paper, we consider to add a new feature to such isolated behaviours before they can be considered as malicious. This feature is based on their possible repetition from one information system to another.

1 Introduction

Intrusion detection is a very important topic of network security that has received much attention [5,9,4,7] since potential cyber threats are making the organizations vulnerable. *Intrusion Detection Systems (IDS)* are intended to protect information systems against intrusions and attacks and are traditionally based on signatures of known attacks [8,1]. Therefore, new kinds of attacks regularly have to be added to the signature list. The main drawback is that in case of an emerging attack, based on the recent discovery of a new security hole for instance, the IDS will ignore it since this new attack has not yet been listed in the base of signatures.

Protecting a system against new attacks, while keeping an automatic and adaptive framework is an important topic in this domain. One answer to that problem could rely on data mining. Data mining tools have been used to provide IDS with more adaptive detection of cyber threats [2,10]. Among those data mining approaches, anomaly detection tries to deduce intrusions from atypical records [4,3]. The overall principle is generally to build clusters, or classes, of usage and find outliers (*i.e.* events that do not belong to any class or group identifying normal usage). However, the main drawback of detecting intrusions

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 891–898, 2009.
© Springer-Verlag Berlin Heidelberg 2009

by means of anomaly (outliers) detection is the high rate of false alarms since an alarm can be triggered because of a new kind of usages that has never been seen before (and is thus considered as abnormal). Considering the large amount of new usage patterns emerging in the Information Systems, even a weak percent of false positive will give a very large amount of spurious alarms that would be overwhelming for the analyst. Therefore, the goal of this paper is to propose an intrusion detection algorithm that is based on the analysis of usage data coming from multiple partners in order to reduce the number of false alarms. Our main idea is that a new usage is likely to be related to the context of the information system on which it occurs (so it should only occur on this system). On the other hand, when a new security hole has been found on a system, the hackers will want to use it in as many information systems as possible. Thus a new anomaly that occurs on two (or more) information systems is probably not a new kind of usage, but rather an intrusion attempt. Let us consider A_x, an anomaly detected in the usage of web site S_1 corresponding to a php request on the staff directory for a new employee: John Doe, who works in room 204, floor 2, in the R&D department. The request will have the following form: `staff.php?FName=John\&LName=Doe\&room=204\&floor=2\&Dpt=RD`. This new request, due to the recent recruitment of John Due in this department, should not be considered as an attack. On the other hand, let us consider A_y, an anomaly that corresponds to a true intrusion. A_y will be based on a security hole of the system (for instance a php vulnerability) and might, for instance, look like: `staff.php?path=../etc/passwd%00`. One can see in this request that the parameters are not related to the data accessed by the php script, but rather to a security hole that has been discovered on the *staff* script. If two or more firms use the same script (say, a directory resquesting script bought to the same software company) then the usage of this security hole will certainly be repeated from one system to another and the request having parameter `../etc/passwd%00` will be the same for all the victims. In this paper, we propose to provide the end-user with a method that takes only one parameter: n, the number of desired alarms. Then, based on the analysis of the usage data coming from the different partners, our algorithm will detect n common outliers they share. Such common outliers are likely to be true attacks and will trigger an alarm.

The paper is organized as follows. In Section 2 we present the motivation of this approach and our general framework. Section 3 presents Cod, our method for detecting outliers and triggering true alarms. Eventually, our method is tested through a set of experiments in Section 4 and Section 5 gives the conclusion.

2 Motivation and General Principle

In this paper we present Cod (Common Outlier Detection) a framework and algorithm intended to detect the outliers shared by at least two partners in a collaborative IDS. Outliers are usually small clusters and our goal is to use outlier lists from different systems (based on a similar clustering, involving the same similarity measure). If an outlier occurs for at least two systems, then

it is considered as an attack. COD is indeed based on the assumption that an intrusion attempt trying to find a weakness of a script will look similar for all the victims of this attack. For clarity of presentation we present our framework on the collaboration of two Web sites, S_1 and S_2 and we consider the requests that have been received by the scripts of each site (cgi, php, sql, etc). Our goal is to perform a clustering on the usage patterns of each site and find the common outliers. However, that would not be enough to meet the second constraint of our objective: to require only one parameter, n, the number of alarms to return. Our similarity measure (presented in section 3.1) will allow normal usage patterns to be grouped together rather than grouped with intrusion patterns. On the other hand, our similarity measure also has to ensure distinguishing an intrusion pattern from normal usage patterns and from other intrusion patterns (since different intrusion patterns will be based on a different security hole and will have very different characteristics). Our algorithm performs successive clustering steps for each site. At each step we check the potentially matching outliers between both sites. The clustering algorithm is agglomerative and depends on the maximum dissimilarity (MD) that has to be respected between two objects.

This work is intended to explore the solutions for monitoring a network in real time. Then, the potential alarms will be triggered at each step of the monitoring (for instance with a frequency of one hour). Depending on the number of true or false alarms, the user might want to adjust n for the next step, until no (or very few) false alarm is returned. Our assumption is that common outliers, sorted by similarity from one site to another, will give the intrusions at the beginning of the list.

3 COD: Common Outlier Detection

The principle of COD is to perform successive clustering steps on usage patterns of different partners sites, until the number of common outliers meets the number of alarms desired by the user. We present in this section an algorithm designed for two information systems. Extending this work to more than two systems would require a central node coordinating the comparisons and triggering the alarms, or a peer-to-peer communication protocol. This is not the goal of this paper. Our objects are the parameters given to script files in the requests received on a Web site. In other words, the access log file is filtered and we only keep lines corresponding to requests with parameters to a script. For each such line, we separate the parameters and for each parameter we create an object. Let us consider, for instance, the following request: `staff.php?FName=John&LName=Doe`. The corresponding objects are o_1 =`John` and o_2 =`Doe`. Once the objects are obtained from the usage data of multiple Web sites, COD is applied and gives their common outliers.

3.1 Main Algorithm

As explained in section 2, COD algorithm will process the usage patterns of both sites step by step. For each step, a clustering result is provided and analyzed for

intrusion detection. First, MD is set to obtain very tight and numerous clusters (very short similarity is allowed between two objects in a cluster). Then, MD is relaxed by an amount of 0.05 step after step in order to increase the size of resulting clusters, decrease their number and lower the number of alarms. When the number of alarms desired by the user is reached, then COD ends.

Algorithm Cod
Input: U_1 and U_2 the usage patterns of sites S_1 and S_2
and n the number of alarms.
Output: I the set of clusters corresponding
to malicious patterns.

1. Build M, the distance matrix between each pattern ;
2. $\forall p \in M, Neighbours_p \leftarrow$ sorted list of neighbours for p (the first usage pattern in the list of p is the closest to p).
3. $DensityList \leftarrow$ sorted list of patterns by density ;
4. $MD \leftarrow 0$;
5. $MD \leftarrow MD + 0.05$;
6. $C_1 \leftarrow Clustering(U_1, MD)$;
   $C_2 \leftarrow Clustering(U_2, MD)$;
7. $O_1 \leftarrow Outliers(C_1)$; $O_2 \leftarrow Outliers(C_2)$;
8. $I \leftarrow CommonOutliers(O_1, O_2, MD)$;
9. If $|I| \leq n$ then return I ;
10. If $MD = 1$ then return I ; // No common outlier
11. Else return to step 5 ;

End algorithm Cod

3.2 Clustering

COD Clustering algorithm is based on an agglomerative principle. The goal is to increase the volume of clusters by adding candidate objects, until the Maximum Dissimilarity (MD) is broken (*i.e.* there is one object o_i in the cluster such that the similarity between o_i and the candidate object o_c is greater than MD).
Similarity between objects. We consider each object as a sequence of characters. Our similarity is then based on the longest common subsequence (LCS), as described in definition 1.

Definition 1. *Let s_1 and s_2 be two sequences. Let $LCS(s_1, s_2)$ be the length of the longest common subsequences between s_1 and s_2. The* dissimilarity $d(s_1, s_2)$ *between s_1 and s_2 is defined as follows:* $d(s_1, s_2) = 1 - \frac{2 \times LCS(s_1, s_2)}{|s_1| + |s_2|}$

Example 1. Let us consider two parameters p_1=`intrusion` and p_2=`induction`. The LCS between p_1 and p_2 is L=`inuion`. L has length 6 and the similarity between p_1 and p_2 is $d = 1 - \frac{2 \times L}{|p_1| + |p_2|} = 33.33\%$. Which also means a similarity of 66.66% between both parameters.

Centre of clusters. When an object is inserted into a cluster we maintain the centre of this cluster, since it will be used in the CommonOutliers algorithm. The centre of a cluster C is the LCS between all the objects in C. When object o_i is added to C, its center C_c is updated. The new value of C_c is the LCS between the current value of C_c and o_i.

Algorithm Clustering
Input: U, the usage patterns
and MD, the Maximum Dissimilarity.
Output: C, the set of as large clusters as possible,
respecting MD.

1. $i \leftarrow 0$; $C \leftarrow \emptyset$;
2. $p \leftarrow$ next unclassified pattern in $DensityList$;
3. $i++$; $c_i \leftarrow p$;
4. $C \leftarrow C + c_i$;
5. $q \leftarrow$ next unclassified pattern in $Neighbours_p$;
6. $\forall o \in c_i$ If $d(o, q) > MD$ then return to step 2 ;
7. add q to c_i ;
8. $C_c \leftarrow LCS(C_c, q)$; //C_c is the center of C
9. return to step 5 ;
10. If unclassified patterns remain then return to step 2 ;
11. return C ;

End algorithm Clustering

3.3 Detecting Common Outliers

Our outlier detection principle is described in [6]. Since we want our global algorithm to require only one parameter (the number of alarms), we want to avoid introducing a similarity degree for comparing two lists of outliers. For this comparison, our algorithm uses the centre of outliers. For each pair of outliers, it calculates the similarity between centers of these outliers. If this similarity is below the current MD, then we consider those outliers as similar and add them to the alarm list.

4 Experiments

The goal of this section is to analyze our results (*i.e.* the number of outliers and true intrusions and the kind of intrusions we have detected). Our datasets come from two different research organizations; (*anonymized for submission*). We have analyzed their Web access log files from March 1 to March 31. The first log file represents 1.8 Gb of rough data. In this file, the total number of objects (parameters given to scripts) is 30,454. The second log file represents 1.2 Gb of rough data and the total number of objects is 72,381. Cod has been written in Java and C++ on a PC (2.33GHz i686) running Linux with 4Gb of

main memory. Parameters that are automatically generated by the scripts have been removed from the datasets since they cannot correspond to attacks (for instance "`publications.php?Category=Books`"). This can be done by listing all the possible generation of parameters in the scripts of a Web site.

4.1 Detection of Common Outliers

As described in Section 2, Cod proceeds by steps and slowly increases the value of MD, which stands for a tolerance value when grouping objects during the clustering process. In our experiments, MD has been increased by steps of 0.05 from 0.05 to 0.5. For each step, we report our measures in table 1. The meaning of each measure is as follows. O_1 (resp. O_2) is the number of outlying objects in site 1 (resp. site 2). $\%_1$ (resp $\%_2$) is the fraction of outlying objects on the number of objects in site 1 (resp. site 2). For instance, when MD is set to 0.3, for site 1 we have 5,607 outlying objects, which represents 18.4% of the total number of objects (*i.e.* 30,454) in site 1. COD is the number of common outliers between both sites and $\%_{FA}$ is the percentage of false alarms within the common outliers. For instance, when MD is set to 0.05, we find 101 alarms among which 5 are false (which represents 4.9%). One first observation is that outliers cannot be directly used to trigger alarms. Obviously, a number as high as 5,607 alarms to check, even for one month, is not realistic. On the other hand, the results of Cod show its ability to separate malicious behaviour from normal usage. Our false alarms correspond to normal requests that are common to both sites but rarely occur. For instance, on the references interrogation script of *anonym_lab1*, a user might request papers of "John Doe" and the request will be `publications.php?FName=John\&LName=Doe`. If another user requests papers of "John Rare" on the Web site of *anonym_lab2*), the request will be `biblio.php?FName=John\&LName=Rare` and the parameter "John" will be given as a common outlier and trigger an alarm. As we can see, $\%_{FA}$ is very low (usually we have at most 5 false alarms in our experiments for both Web sites) compared to the thousands of outliers that have been filtered by Cod. Another lesson from these experiments is that a low MD implies very small clusters and numerous outliers. These outliers are shared between both sites, among which some are false alarms due to rare but common normal usage. When MD increases, the clustering process gets more agglomerative and alarms are grouped together. Then one alarm can cover several ones of the same kind (*e.g.* the case of easter eggs explained further). At the same time, the number of outliers corresponding to normal usage decreases (since they are also grouped together). Eventually, a too large value of MD implies building clusters that do not really make sense. In this case, outliers will get larger, and the matching criteria will get too tolerant, leading to a large number of matching outliers capturing normal usage. In a streaming environment involving the real data of these experiments, one could decide to keep 70 as the number of desired alarms and watch the ratio of false alarms. If this ratio decreases, then the end-user should consider increasing the number of desired alarms.

Table 1. Results on real data

	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
O_1	13197	10860	8839	7714	6547	5607	5184	4410	3945	3532
$\%_1$	43.3%	35.6%	29%	25.3%	21.5%	18.4%	17%	14.4%	12.9%	11.6%
O_2	35983	27519	24032	20948	18152	14664	12738	11680	10179	8734
$\%_2$	49.6%	37.9%	33.1%	28.9%	25%	20.2%	17.5%	16.1%	14%	12.1%
COD	101	78	74	70	67	71	71	85	89	90
$\%_{FA}$	4.9%	5.12%	4%	2.85%	1.5%	2.8%	2.8%	10.6%	11.2%	16.6%

4.2 A Sample of Our Results

None of the attacks found in our experiments have been successful on the considered Web sites. However, our security services and our own investigations allow us to confirm the intrusion attempts that have been discovered by our method:

- **Code Injection:** a recent kind of attack aims to inject code in PHP scripts by giving a URL in the parameters. Here is a samle of such URLs detected by COD:
  - http://myweddingphotos.by.ru/images?
  - http://levispotparty.eclub.lv/images?
  - http://0xg3458.hub.io/pb.php?

  Depending on the PHP settings on the victim's Web server, the injected code allows modifying the site. These URLs are directly, automatically and massively given as parameters to scripts through batches of instructions.
- **Passwords:** another kind of (naive and basic) attack aims to retrieve the password file. This results in outliers containing parameters like `../etc/password` with a varying number of `../` at the beginning of the parameter. This is probably the most frequent attempt. It is generally not dangerous but shows the effectiveness of our method.
- **Easter Eggs:** this is not really an intrusion but if one adds the code ?=PHPE9568F36-D428-11d2-A769-00AA001ACF42 to the end of any URL that is a PHP page, he will see a (funny) picture on most servers. Also on April 1st (April Fool's Day), the picture will replace the PHP logo on any phpinfo() page. This code (as well as two other ones, grouped into the same outlier) has been detected as a common outlier by COD.

5 Conclusion

In this paper, we have proposed i) an unsupervised clustering scheme for isolating atypical behaviours, ii) a parameterless outlier detection method based on wavelets and iii) a new feature for characterizing intrusions. This new feature is based on the repetition of an intrusion attempt from one system to another. Actually, our experiments show that atypical behaviours cannot be directly used to trigger alarms since most of them correspond to normal requests. On the other hand, this very large number of outliers can be effectively filtered (reducing the

amount of atypical behaviours up to 0.21%) in order to find true intrusion attempts (or attacks) if we consider more than one site. Eventually, our method guarantees a very low ratio of false alarms, thus making unsupervised clustering for intrusion detection effective, realistic and feasible.

Acknowledgement

The authors want to thank Laurent Mirtain, the responsible for intrusion detection of Inria Sophia-Antipolis, for his assistance in identifying attacks in our access log files.

References

1. Barbara, D., Wu, N., Jajodia, S.: Detecting novel network intrusions using bayes estimators. In: 1st SIAM Conference on Data Mining (2001)
2. Bloedorn, E., Christiansen, A.D., Hill, W., Skorupka, C., Talbot, L.M.: Data mining for network intrusion detection: How to get started. Technical report, MITRE (2001)
3. Eskin, E., Arnold, A., Prerau, M., Portnoy, L., Stolfo, S.: A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. Applications of Data Mining in Computer Security (2002)
4. Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., Srivastava, J.: A comparative study of anomaly detection schemes in network intrusion detection. In: 3rd SIAM DM (2003)
5. Lee, W., Stolfo, S.J.: Data mining approaches for intrusion detection. In: 7th conference on USENIX Security Symposium (1998)
6. Marascu, A., Masseglia, F.: A multi-resolution approach for atypical behaviour mining. In: The 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2009), Bangkok, Thailand (2009)
7. Patcha, A., Park, J.-M.: An overview of anomaly detection techniques: Existing solutions and latest technological trends. Comput. Networks 51 (2007)
8. Roesch, M.: SNORT (1998)
9. Valdes, A., Skinner, K.: Probabilistic alert correlation. In: Recent Advances in Intrusion Detection, pp. 54–68 (2001)
10. Wu, N., Zhang, J.: Factor analysis based anomaly detection. In: IEEE Workshop on Information Assurance (2003)

A Multi-resolution Approach for Atypical Behaviour Mining

Alice Marascu and Florent Masseglia

INRIA Sophia Antipolis, 2004 route des lucioles - BP 93, FR-06902 Sophia Antipolis
First.Last@sophia.inria.fr

Abstract. Atypical behaviours are the basis of a valuable knowledge in domains related to security (*e.g.* fraud detection for credit card [1], cyber security [4] or safety of critical systems [6]). Atypicity generally depends on the isolation level of a (set of) records, compared to the dataset. One possible method for finding atypic records aims to perform two steps. The first step is a clustering (grouping the records by similarity) and the second step is the identification of clusters that do not correspond to a satisfying number of records. The main problem is to adjust the method and find the good level of atypicity. This issue is even more important in the domain of data streams, where a decision has to be taken in a very short time and the end-user does not want to try several settings. In this paper, we propose Mrab, a self-adjusting approach intending to automatically discover atypical behaviours (in the results of a clustering algorithm) without any parameter. We provide the formal framework of our method and our proposal is tested through a set of experiments.

1 Introduction

Atypical behaviours are the basis of a valuable knowledge in domains related to security (*e.g.* fraud detection for credit card [1], cyber security [4] or safety of critical systems [6]). Atypicity generally depends on the isolation level of a (set of) records, compared to the dataset. One possible method for finding atypic records aims to perform two steps. The first step is a clustering (grouping the records by similarity) and the second step is the identification of clusters that do not correspond to a satisfying number of records. Actually, atypical events (or outliers) might be indicative of suspicious data such as skewed or erroneous values, entry mistakes or malicious behaviours. A malicious behaviour can be detected as an outlier in datasets such as transactions in a credit card database or records of usage on a web site. To the best of our knowledge, outlier detection always relies on a parameter, given by the end-user and standing for a "degree of outlyingness" above which records are considered as atypical. For instance, in [10], a *distance-based* outlier is an object such that a *user-defined fraction* of dataset objects have a distance of more than a *user-defined minimum distance* from that object. In [5], the authors propose a nonparametric clustering process and the detection of outliers requires a *user defined value* k corresponding to the top-k desired outliers.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 899–906, 2009.
© Springer-Verlag Berlin Heidelberg 2009

In this paper we propose MRAB (Multi-Resolution detection of Atipycal Behaviours), a parameterless method intending to automatically extract outliers from a clustering result. In contrast to previous work, our goal is to find the best division of a distribution and to automatically separate values into two sets corresponding to clusters on the one hand and outliers on the other hand. Our method fits any clustering result depending on any characteristic such as distances between objects [10], objects' density [2,11] or clusters' size [7,13]. Our framework involves clustering-based outlier detection in data streams. This framework will allow us to illustrate our proposal with one of the possible characteristics observed for building a distribution of objects (*i.e.* clusters' size). The choice of data streams is motivated by the specific constraints of this domain. In a data stream environment, data are generated at a very high rate and it is not possible to perform blocking operations. In this context, requesting a parameter such as k, for top-k outliers, or p, a percentage of small clusters, should be prohibited. First, because the user doesn't have enough time to try different values of these parameters for each period of analysis on the stream. Secondly, because a permanent value may be adapted to one period of the stream but it is highly likely to be wrong on the next periods. Actually, from one batch to another, the clustering results will evolve and the data distribution will change, as well as the number or percentage of outliers. For these reasons, detecting outliers should not depend on any parameter and should be adaptive in order to keep the best accuracy all along the stream.

Section 2 gives the details of MRAB and its principle for separating outliers from clusters. Section 3 shows the advantages of MRAB through a set of experiments on real Web usage data and Section 4 gives our conclusion.

2 Parameterless Outlier Detection

Clustering is the problem of finding a partition of a data set so that similar objects are in the same part of the partition and different objects are in different parts. A data stream $S = \{S_1, ..., S_i, ..., S_n\}$ is a series of batches S_i, read in increasing order of the indices i. Each batch contains a set of objects $O = \{o_1, ..., o_m\}$. Our goal is to separate clusters in order to give the list of atypical events. This is the core of this paper, and our principle (based on a multi-resolution analysis) for this parameterless detection is presented in Section 2.

Most previous work in outlier detection requires a parameter [8,15,12,9], such as a percent of small clusters that should be considered as outliers or the top-n outliers. Generally, their key idea is to sort the clusters by size and/or tightness. We consider that our clusters will be as tight as possible, according to our clustering algorithm, and we aim to extract outliers by sorting the clusters by size. The problem is to separate "big" and "small" clusters without any *apriori* knowledge about what is big or small. Our solution is based on an analysis of the clusters' distribution, once they are sorted by size. The key idea of MRAB is to use a wavelet transform to cut down such a distribution. With a prior knowledge on the number of plateaux (we want two plateaux, the first one standing

for small groups, or outliers, and the second one standing for big groups, or clusters) we can cut the distribution in a very effective manner. Actually, each cluster having size lower than (or equal to) the first plateau will be considered as an outlier.

The wavelet transform is a tool that cuts up data or functions or operators into different frequency components, and then studies each component with a resolution matched to its scale [3]. In other words, wavelet theory represents series of values by breaking them down into many interrelated component pieces; when the pieces are scaled and translated wavelets, this breaking down process is termed wavelet decomposition or wavelet transform. Wavelet reconstructions or inverse wavelet transforms involve putting the wavelet pieces back together to retrieve the original object [14]. Mathematically, the continuous wavelet transform is defined by:

$$T^{wav} f(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(x)\psi^*(\frac{x-b}{a})dx$$

where z^* denotes the complex conjugate of z, $\psi^*(x)$ is the analyzing wavelet, a (> 0) is the scale parameter and b is the translation parameter. This transform is a linear transformation and it is co-variant under translations and dilations. This expression can be equally interpreted as a signal projection on a function family analyzing $\psi_{a,b}$ constructed from a mother function in accordance with the following equation: $\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi(\frac{t-b}{a})$. Wavelets are a family of basis functions that are localized in time and frequency and are obtained by translations and dilations from a single function $\psi(t)$, called the mother wavelet. For some very special choices of a, b, and ψ, $\psi_{a,b}$ is an orthonormal basis for $L^2(\mathbb{R})$. Any signal can be decomposed by projecting it on the corresponding wavelet basis function. To understand the mechanism of wavelet transform, we must understand the multiresolution analysis (MRA). A multiresolution analysis of the space $L^2(\mathbb{R})$ consists of a sequence of nested subspaces such as:

$... \subset V_2 \subset V_1 \subset V_0 \subset V_{-1} ... \subset V_{j+1} \subset V_j ...$

$$\overline{\bigcup_{j\in Z} V_j} = L^2(\mathbb{R})$$
$$\bigcap_{j\in Z} V_j = \{0\}$$

$\forall j \in \mathbb{Z}$ if $f(x) \in V_j \iff f(2^{-1}x) \in V_{j+1}$ (or $f(2^j x) \in V_0$)

$\forall k \in \mathbb{Z}$ if $f(x) \in V_0 \iff f(x-k) \in V_0$

There is a function $\varphi(x) \in L^2(\mathbb{R})$, called scaling function, which by dilation and translation generates an orthonormal basis of V_j. Basis functions are constructed according to the following relation : $\varphi_{j,n}(x) = 2^{-\frac{j}{2}}\varphi(2^{-j}x - n), n \in \mathbb{Z}$, and the basis is orthonormated if $\int_{-\infty}^{+\infty} \varphi(x)\varphi^*(x+n)dx = \delta(n), n \in \mathbb{Z}$. For each V_j, its orthogonal complement W_j in V_{j-1} can be defined as follows:$V_{j-1} = V_j \oplus W_j$ and $L^2(\mathbb{R}) = \bigoplus_{j\in\mathbb{Z}} W_j$. As W_j is orthogonal to V_{j-1}, then W_{j-1} is orthogonal to W_j, so $\forall j, k \neq j$ then $W_j \perp W_k$.

There is a function $\psi(x) \in \mathbb{R}$, called wavelet, which by dilations and translations generates an orthonormal basis of W_j , and so of $L^2(\mathbb{R})$. The basis functions are constructed as follows: $\psi_{j,n}(x) = 2^{-\frac{j}{2}}\psi(2^{-j}x - n), n \in \mathbb{Z}$. Therefore, $L^2(\mathbb{R})$

is decomposed into an infinite sequence of wavelet spaces, i.e.$L^2(\mathbb{R}) = \bigoplus_{j \in Z} W_j$. To summarize the wavelet decomposition: given a f_n function in V_n, f_n is decomposed into two parts, one part in V_{n-1} and the other in W_{n-1}. At next step, the part in V_{n-1} continues to be decomposed into two parts, one part in V_{n-2} and the other in W_{n-2} and so on.

A direct application of multiresolution analysis is the fast discrete wavelet transform algorithm. The idea is to iteratively smooth data and keep the details all along the way. More formal proofs about wavelets can be found in [3]. The wavelet transform provides a tool for time-frequency localization and are generally used to summarize data and to capture the trend in numerical functions. In practice, the majority of wavelets coefficients are small or insignificant, so to capture the trend only a few significant coefficients are needed. We use the Haar wavelets to illustrate our outlier detection method. Let us consider the following series of values: $[1, 1, 2, 5, 9, 10, 13, 15]$. Its Haar wavelet transform is illustrated by the following table:

Level	Approximations	Coefficients
8	1, 1, 2, 5, 9, 10, 13, 15	
4	1, 3.5, 9.5, 14	0, -1.5, -0.5, -1
2	2.25, 11.75	-1.25, -2.25
1	7	-4.75

Then, we keep only the most two significant coefficients. In our series of coefficients ($[7, -4,75, -1.25, -2.25, 0, -1.5, -0.5, -1]$) the most two significant ones are 7 and $-4,75$, meaning that the series becomes [7, $-4,75$, 0, 0, 0, 0, 0, 0]. In the following step, the inverse operation is calculated and we obtain an approximation of the original data [2.25, 2.25, 2.25, 2.25, 11.75, 11.75, 11.75, 11.75]. This gives us two plateaux corresponding to values $\{1, 1, 2, 5\}$ and $\{9, 10, 13, 15\}$. The set of outliers contains all the clusters having size smaller than the first plateau(*e.g.* 2.25). In our example, $o = \{1, 1, 2\}$ gives the sizes of outliers (*i.e.* clusters having size ≤ 2).

More generally, advantages of this method, for our problem, are illustrated in figure 1. In figure 1, the y axis stands for the size of the clusters, whereas their index in the sorted list is represented on x, and the two plateaux allow separating small and big clusters. Depending on the distribution, wavelets will give different indexes (where to cut). For instance, in our usage data at *anonymized_lab*, there is a variation between night and day in the usage of PHP scripts. This variation results into two main shapes. Figure 1 gives an illustration of two different distributions, similar to the ones we found out in our experiments. Let us consider the 10% filter on this distribution, which aims to isolate outliers corresponding to 10% of the global shape. If one uses the 10% percent filter in order to detect outliers, he will obtain a relevant outlier detection for the first distribution (corresponding to usage of scripts at 1 am). However, with the second distribution (calculated from the usages at 5 pm), this filter will give a very high value and return clusters that should not be considered as outliers. On the other hand, our wavelet based filter will adjust to the distribution variation and the

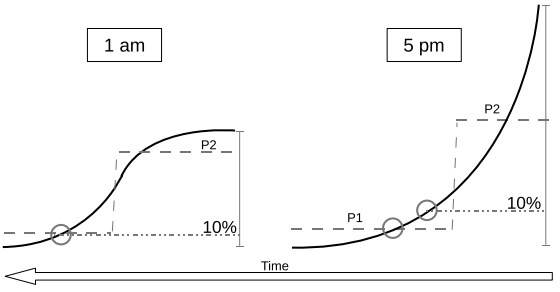


Fig. 1. A Distribution Varying With Time

threshold for outlier detection will only slightly increase, taking into account the new distribution shape.

Applying the wavelet transform on the series allows us to obtain a good data compression and, meanwhile, according to different trends, a good separation. Knowing that outliers are infrequent objects, they will always be grouped into small clusters.

3 Experiments

Choosing a good level of outlyingness in a streaming environment is highly difficult given the short time available to take a decision. In this context, an atypical event detection method which does not depend on a parameter such as k, for the top-k outliers, or a percentage p of small clusters, should be much appreciated. On the other hand, such a parameterless method also has to guarantee good results. This method should be able to provide the end-user with an accurate separation of the clusteing results into small and big clusters. It should also be able to fit any kind of distribution shape (exponential, logarithmic, linear, etc.). The most important characteristic is to be able to automatically adjust to the number of clusters and to their size from one batch to the other. Our claim is that MRAB matches all these requirements and we illustrate these features in this section.

For these experiments we used real data, coming from the Web Log usage of *anonymized_lab* from January 2006 to April 2007. The original files have a total size of 18 Gb and they correspond to a total of 11 millions navigations that have been split into batches of 8500 requests each (in average). In these experiments, we focus on 16 batches, since they are representative of the global results and they illustrate the variation of distribution. The first 8 batches have been selected among PHP request occuring between 1 and 2 am, and the 8 former have been selected among requests occuring between 3 and 4 pm.

Figure 2 shows the behaviour of filters top-k and p% on those 16 batches. The first surface (left) shows the size of clusters selected by a top-k filter. The principle of this filter is to select only the first k clusters after sorting them by

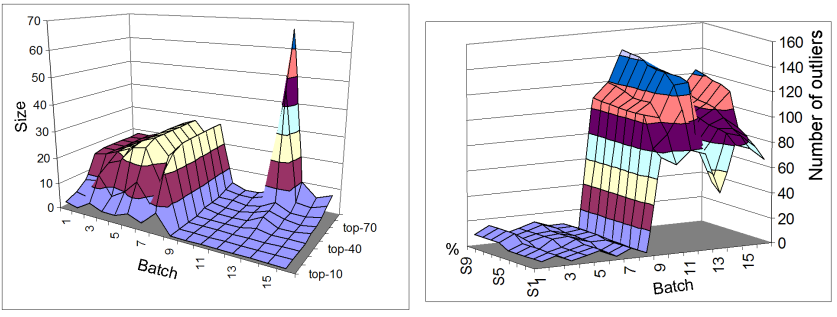


Fig. 2. size of outliers with a top-k filter and number of outliers with a p% filter

size. An obvious disadvantage of this filter is to select either too much or not enough clusters. Let us consider, for instance, batch 13 in Figure 2. With $k = 50$ the maximum outliers size is 4, whereas with $k = 90$ this size is 67 (which is the maximum size of a cluster in this batch, which contains only 84 clusters).

We have also implemented a filter based on p%, a percentage of clusters, for outlier selection. The number of outliers selected by this filter with different values of p (*i.e.* from 0.01 to 0.09) is given in figure 2 (right surface). The principle is to consider $p \in [0..1]$, a percentage given by the end-user, $d = maxVal - minVal$ the range of cluster sizes and $y = (p \times d) + minVal$. Then, the filter aims to select only clusters having size t, such that $t \leq y$. For instance, with $s = \{1, 3, 10, 11, 15, 20, 55, 100\}$ a series of sizes and $p = 0.1$ we have $d = 100 - 1 = 99$, $y = 1 + (0.1 \times 99) = 10$ and the set of outliers will be $o = \{1, 3, 10\}$. In our experiments, this filter is generally better than a top-k filter. Actually, we can notice homogeneous results from Figure 2. For instance, with batch 13 we can see a number of outliers ranging from 44 (1 %) to 78 (9 %), which corresponds to the results of top-40 to top-70.

Figure 3 gives a comparison of MRAB (applied to the same data) with top-k and percentage filtering. In the left part of Figure 3, we compare MRAB with a

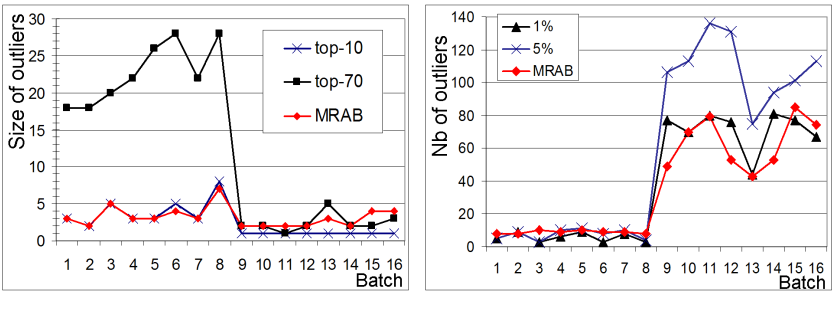


Fig. 3. Comparison with MRAB for top-k and p%

top-10 and a top-70 filter. For the first 8 batches, top-10 and MRAB give the best result. Unfortunately, for batches 9 to 16, top-10 returns too few outliers (having maximum size 1). Therefore this filter cannot be used for the whole stream. The best top-k results for batches 9 to 16 are given by a top-70. Unfortunately, its results for batches 1 to 8 are bad (values are too high, with outlier having size up to 28). Therefore, no value of k in this filter can be considered as a reference. The end-user would have to modify the value of k from one batch to another. This result is thus not acceptable and shows that top-k is unable to adjust to changes in the distribution of cluster sizes. On the other hand, thanks to its wavelet feature, MRAB is able to automatically adjust and will always select a correct maximum size to detect atypical events.

In the right part of Figure 3, we focus on two percentage filters (*i.e.* 1% and 5%) and we compare them to MRAB. Our observation is that MRAB and 1% would give similar results. For instance, with batch 7, we know that MRAB labels clusters having size less than or equal to 3 as outliers. That filtering gives a total of 9 clusters (where filter 1 % gives 8 outliers). We also can observe that most of the values given by filter 1 % on the first 8 batches are low. Filter 5 % gives better values for the first 8 batches (very similar to MRAB) but it has bad results on the next 8 batches (up to 138 outliers). This is due to the variation of the distribution, as illustrated by Figure 1. Therefore, the advantage of MRAB over the percentage filter is double:

1. MRAB does not require any parameter tuning. It adjusts automatically, whatever the distribution shape and the number of clusters. In contrast, the end-user will have to try several percentage values before finding the good range (*i.e.* the 5 % filter gives good outliers for the first batches). Furthermore, the outlier detection provided by MRAB will not degrade with a variation of distribution shape over time. Let us consider a change of usage, or a change of clustering method, resulting in a variation of the distribution shape. That new shape could vary as illustrated in Figure 1. Then, the percentage filter would have to be manually modified to fit that new distribution, whereas MRAB would keep giving the good set of outliers without manual settings.
2. MRAB gives an optimal separation between small and big values . Let us consider our previous illustration of a distribution $s = \{1, 3, 10, 11, 15, 20, 55, 100\}$. We know that on this distribution a 10% filter would give the following set of outliers : $o = \{1, 3, 10\}$. However, why not including 11 into o? Actually, 10 and 11 are very close values. On the other hand, with MRAB we have $o = \{1, 3\}$, which is obviously a natural and realistic result.

4 Conclusion

In this paper we have presented MRAB, an outlier detection method that does not require any manual tuning. Our principle is first based on a distribution of clusters according to some characteristics such as their size, tightness, density or any other characteristic. Thanks to its wavelet feature, MRAB is able to cut

down this distribution into two sets, corresponding to clusters and outliers. The advantages of MRAB are i) automatic adjustment to distribution shape variations and ii) relevant and accurate detection of outliers with very natural results. Our experiments, performed on real data, confirm this separation feature of MRAB compared to well-known outlier detection principles such as the top-k outliers or the percentage filter.

References

1. Aleskerov, E., Freisleben, B., Rao, B.: Cardwatch: A neural network based database mining system for credit card fraud detection. In: IEEE Computational Intelligence for Financial Engineering (1997)
2. Breunig, M.M., Kriegel, H.-P., Ng, R.T., Sander, J.: Lof: identifying density-based local outliers. SIGMOD Records 29(2), 93–104 (2000)
3. Daubechies, I.: Ten lectures on wavelets. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (1992)
4. Ertoz, L., Eilertson, E., Lazarevic, A., Tan, P.-N., Kumar, V., Srivastava, J., Dokas, P.: Minds - minnesota intrusion detection system. Data Mining - Next Generation Challenges and Future Directions (2004)
5. Fan, H., Zaiane, O.R., Foss, A., Wu, J.: A nonparametric outlier detection for effectively discovering top-n outliers from engineering data. In: Ng, W.-K., Kitsuregawa, M., Li, J., Chang, K. (eds.) PAKDD 2006. LNCS, vol. 3918, pp. 557–566. Springer, Heidelberg (2006)
6. Fujimaki, R., Yairi, T., Machida, K.: An approach to spacecraft anomaly detection problem using kernel feature space. In: 11th ACM SIGKDD (2005)
7. Jaing, M.F., Tseng, S.S., Su, C.M.: Two-phase clustering process for outliers detection. Pattern Recogn. Lett. 22(6-7), 691–700 (2001)
8. Jin, W., Tung, A.K.H., Han, J.: Mining top-n local outliers in large databases. In: 7th ACM SIGKDD, pp. 293–298 (2001)
9. Oldmeadow, J.J., Ravinutala, S., Leckie, C.: Adaptive clustering for network intrusion detection. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS, vol. 3056, pp. 255–259. Springer, Heidelberg (2004)
10. Knorr, E.M., Ng, R.T.: Algorithms for mining distance-based outliers in large datasets. In: 24th VLDB, pp. 392–403 (1998)
11. Papadimitriou, S., Kitagawa, H., Gibbons, P.B., Faloutsos, C.: LOCI: fast outlier detection using the local correlation integral. In: 19th International Conference on Data Engineering (2003)
12. Portnoy, L., Eskin, E., Stolfo, S.: Intrusion detection with unlabeled data using clustering. In: ACM CSS Workshop on Data Mining Applied to Security (2001)
13. Sequeira, K., Zaki, M.: Admit: anomaly-based data mining for intrusions. In: KDD 2002, pp. 386–395. ACM, New York (2002)
14. Young, R.K.: Wavelet Theory and Its Applications. Kluwer Academic Publishers Group, Dordrecht (1995)
15. Zhong, S., Khoshgoftaar, T.M., Seliya, N.: Clustering-based network intrusion detection. International Journal of Reliability, Quality and Safety Engineering 14 (2007)

Change Analysis in Spatial Data by Combining Contouring Algorithms with Supervised Density Functions

Chun Sheng Chen[1], Vadeerat Rinsurongkawong[1], Christoph F. Eick[1], and Michael D. Twa[2]

[1] Department of Computer Science, University of Houston, Houston, TX 77204-3010
{lyons19,vadeerat,ceick}@cs.uh.edu
[2] College of Optometry, University of Houston, Houston, TX 77204-6052
mtwa@optometry.uh.edu

Abstract. Detecting changes in spatial datasets is important for many fields. In this paper, we introduce a methodology for change analysis in spatial datasets that combines contouring algorithms with supervised density estimation techniques. The methodology allows users to define their own criteria for features of interest and to identify changes in those features between two datasets. Change analysis is performed by comparing interesting regions that have been derived using contour clustering. A novel clustering algorithm called DCONTOUR is introduced for this purpose that computes contour polygons that describe the boundary of a supervised density function at a given density threshold. Relationships between old and new data are analyzed relying on polygon operations. We evaluate our methodology in a case study that analyzes changes in earthquake patterns.

Keywords: Change analysis, spatial data mining, region discovery, supervised density estimation, contour clustering, interestingness comparison.

1 Introduction

Spatial datasets, containing geo-referenced data, are growing at a very high speed. Detecting changes in spatial datasets is important for many fields such as early warning systems that monitor environmental conditions or sudden disease outbreaks, epidemiology, crime monitoring, and automatic surveillance.

To address this need, this paper introduces a novel methodology and an algorithm that discover patterns of change in spatial datasets. We are interested in finding what patterns emerged between two datasets, O_{old} and O_{new}, sampled at different time frames. Change analysis centers on identifying changes concerning interesting regions with respect to O_{old} and O_{new}. Moreover, an approach to define interestingness perspectives is introduced. The approach employs supervised density functions [6] that

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 907–914, 2009.
© Springer-Verlag Berlin Heidelberg 2009

create density maps from spatial datasets. As we will explain later, regions (contiguous areas in the spatial subspace) where density functions take high (or low) values are considered interesting by this approach. Interesting regions are identified using contouring techniques.

The contributions of this paper include:

1. A novel clustering algorithm called DCONTOUR is introduced. To the best of our knowledge, DCONTOUR is the first density-based clustering algorithm that uses contour lines to determine cluster boundaries that are described as polygons. Objects that are inside a contour polygon belong to a cluster. DCONTOUR operates on the top of supervised density functions that capture what places a domain experts considers to be interesting.
2. A framework for change analysis in spatial dataset is presented that compares interesting regions that have been derived using contour clustering. It analyzes change in interestingness by comparing contour polygons.

2 Change Analysis Using Supervised Density Estimation Approach

2.1 Supervised Density Estimation

We assume that objects o in a dataset $O=\{o_1,\ldots,o_n\}$ have the form ((x, y), z) where (x, y) is the location of object o and z—denoted as z(o) is the value of the variable of interest of object o. In the following, we will introduce supervised density estimation techniques. Density estimation is called supervised because in addition to the density based on the locations of objects, we take the variable of interest z(o) into consideration when measuring density. The density estimation techniques employ influence functions that measure the influence of a point $o \in O$ with respect to another point $v \in F$; in general, a point o's influence on another point v's density decreases as the distance between o and v, denoted by d(o,v), increases. In contrast to past work in density estimation, our approach employs weighted influence functions to measure the density in datasets O: the influence of o on v is weighted by z(o) and measured as a product of z(o) and a Gaussian kernel function. In particular, the influence of object $o \in O$ on a point $v \in F$ is defined as:

$$f_{\text{inf luence}}(v,o) = z(o) * e^{-\frac{d(v,o)^2}{2*\sigma^2}} \tag{2-1}$$

If $\forall o \in O$ z(o)=1 holds, the above influence function becomes a Gaussian kernel function, commonly used for density estimation and by the density-based clustering algorithm DENCLUE [5]. The parameter σ determines how quickly the influence of o on v decreases as the distance between o and v increases. The overall influence of all data objects $o_i \in O$ for $1 \leq i \leq n$ on a point $v \in F$ is measured by the density function $\psi^O(v)$, which is defined as follows:

$$\psi^o(v) = \sum_{i=1}^{n} f_{\text{inf luence}}(v, o_i) \tag{2-2}$$

In summary, supervised density estimation does not only consider the frequency with which spatial events occur but also takes the value of the variable of interest into consideration—in general, density increases as frequency and z(o) increase.

2.2 Change Analysis through Contour Clustering

We have developed a contour clustering algorithm named DCONTOUR that combines contouring algorithms and density estimation techniques. Fig. 1 gives the pseudo-code of the algorithm.

Input: Density function ψ^o, density threshold d.
Output: Density polygons for density threshold d.
1. Subdivide the space into D grid cells.
2. Compute densities at grid intersection points by using density function ψ^o.
3. Compute contour intersection points b on grid cell edges where $\psi^o(b) = d$ using binary search and interpolation.
4. Compute contour polygons from contour intersection points b.

Fig. 1. Pseudo-code of the DCONTOUR algorithm

Step 1 creates a grid structure for the space and step 2 computes the density for each grid intersection point. Step 2 will call the density function $\mathbf{O}(D)$ times where D is the number of grid cells. In general, objects that are far away from a point contribute very little to the density of the point. Therefore, in order to speed up step 2, we implemented an "approximate" density function that only considers the influence of objects belonging to neighboring grid cells rather than all the objects in the dataset. Step 3 computes contour intersection points on grid edges. Since the density function is defined in the whole space and is nonlinear, binary search on cell-edges is used in step 3 to limit the interpolation error. Fig. 2 gives an illustration of how contour intersection points for d=4.5 are constructed. As far as the right edge of the lower left cell is concerned, because 4.5 is between 4.1 and 5.5, a contour intersection point exists on this edge; interpolating between 4.1 and 5.5, a point on this edge is sampled and its density is computed which turns out to be 4.8. Because 4.8 is larger than d, we continue the binary search by sampling a point south of this point. The binary search terminates if the density difference between a sampled point and d is less than a threshold ε. Finally, in step 4, we connect contour intersection points b found on cell edges and continue this process on its neighboring cells until a closed polygon is formed or both ends of the polyline reach the grid boundary. Step 4 uses an algorithm that was proposed by Snyder [3] to compute contours from intersection points.

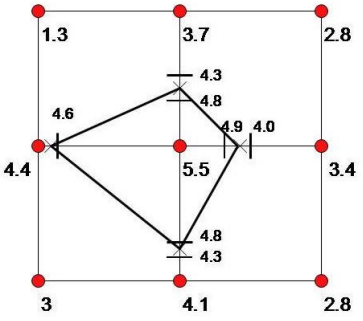


Fig. 2. Contour construction for d=4.5

Traditional contouring algorithms operate on datasets of the form ((x,y),u) where u is a measurement of an attribute of interest at the location (x,y), and use interpolation to infer values of u in locations that are not sampled. DCONTOUR, on the other hand, creates contour polygons for a given density intensity using supervised density maps as its input. A contour polygon acts as a boundary of interesting regions that are above (or below) a specific density threshold; objects surrounded by each individual polygon are defined as a cluster.

In the next two sections, a change analysis approach is introduced that applies DCONTOUR to O_{old} and O_{new}, and analyzes change with respect to the obtained contour polygons.

3 Change Analysis Predicates

This section introduces basic predicates that capture different relationships for change analysis. Given two clusterings X and X' for O_{new} and O_{old}, respectively, relationships between the regions that belong to X and X' can be analyzed. Let r be a region in X and r' be a region in X'. In this case, agreement between r and r' can be computed as follows:

- Agreement(r,r')= $|r \cap r'|/|r \cup r'|$

In general, the most similar region r' in X' with respect to r in X is the region r' for which Agreement(r,r') has the highest value. In addition to agreement, we also define predicates novelty, relative-novelty, disappearance and relative-disappearance below.

Let r, r_1, r_2,..., r_k be regions discovered at time t, and r', r_1', r_2',…, r_k' be regions that have been obtained for time t+1.

- Novelty (r') = $(r' — (r_1 \cup \ldots \cup r_k))$
- Relative-Novelty(r') = $|r' — (r_1 \cup \ldots \cup r_k)|/|r'|$
- Disappearance(r) = $(r — (r'_1 \cup \ldots \cup r'_k))$
- Relative-Disappearance(r) = $|r — (r'_1 \cup \ldots \cup r'_k)|/|r|$

Novelty measure captures regions that have not been interesting in the past. On the other hand, disappearance is used to discover regions where those characteristics are

disappearing. Relative-novelty and relative-disappearance measure percentages of novelty and disappearance. We claim that the above and similar measurements are useful to identify *what is new* in a changing environment. Moreover, the predicates we introduced so far can be used as building blocks to define more complex predicates.

It is also important to note that the above predicates are generic in the sense that they can be used to analyze changes between the old and new data based on different types of clustering. The change analysis approach that we introduced in sections 2 and 3 uses polygons as cluster models. Consequently, in our particular approach the operators '$\cap$', '$\cup$', and '—' denote polygon intersection, union and difference and |r| computes the size (area) of a polygon r. For example, agreement between two polygons r and r' is computed as the ratio of the size of the intersection between r and r' over the size the union of r and r'.

4 Demonstration

We demonstrate our methods on an earthquake dataset which is available on website of the U.S. Geological Survey Earthquake Hazards Program http://earthquake.usgs.gov/. Information recorded includes the location (longitude,

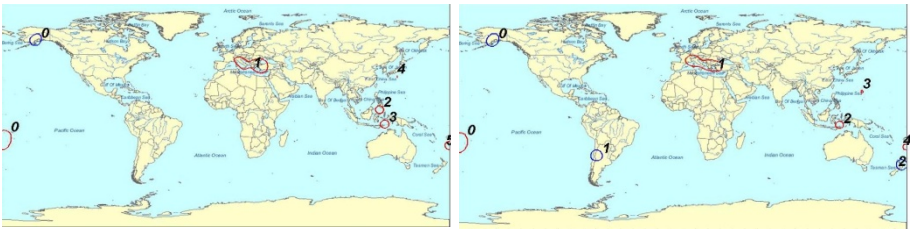


Fig. 3a. Contour polygons generated by DCONTOUR for O_{old} (left figure) and O_{new} (right figure). Blue polygons indicate areas with significant negative correlations (deep earthquakes are always less severe and shallow earthquakes tend to be strong). Red polygons are areas having positive correlations between the two variables. Blue and red polygons are numbered from 0 to n.

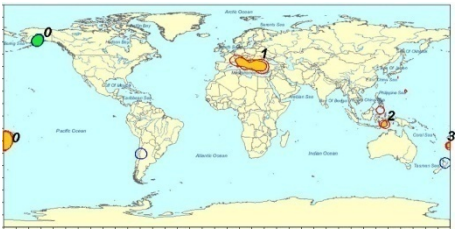


Fig. 3b. Overlap of contour polygons of O_{old} and O_{new} dataset

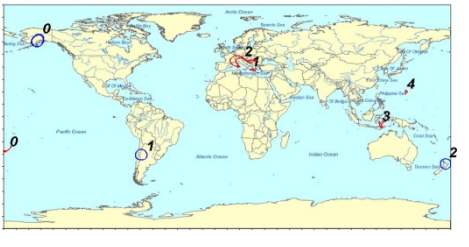


Fig. 3c. Novel polygons of dataset O_{new} with respect to dataset O_{old}

latitude), the time, the severity (Richter magnitude) and the depth (kilometers) of earthquakes. We uniformly sampled earthquakes dated from January 1986 to November 1991 as dataset O_{old} and earthquakes between December 1991 and January 1996 as dataset O_{new}. Each dataset contains 4132 earthquakes.

In this section, we discuss a case study that analyzes changes in strong positive or negative correlations between the depth of the earthquake and the severity of the earthquake. Accordingly, the variable of interest, z(o) is defined as follows:

$$z(o) = \frac{\left(severity(o) - \mu_{severity}\right)}{\sigma_{severity}} * \frac{\left(depth(o) - \mu_{depth}\right)}{\sigma_{depth}} \quad (4\text{-}1)$$

where $\mu_{severity}$ and μ_{depth} are the mean values of the severity and depth of the dataset and $\sigma_{severity}$ and σ_{depth} are the standard deviation of the earthquake severity and depth, respectively. It should be noted that that z(o) takes positive and negative values, and that the constructed density function now contains hotspots (areas high positive density) and cool spots (areas of high negative density), which identify regions of positive and negative correlation, respectively. Fig. 3.a shows the results of running DCONTOUR once with a negative density threshold and once with a positive threshold. For a better viewing of the figures, see http://www.cs.uh.edu/~ceick/kdd/CRET08.pdf.

In Fig. 3.b shows the intersection regions of the two datasets (filled by orange are positive-correlated areas and filled by green are negative-correlated areas). Using the agreement predicate, we compute that red polygon #0 in O_{old} has overlap with red polygon #0 in O_{new} of 84%. Fig. 3.c shows the novel polygons in dataset O_{new} with respect to to dataset O_{old}. For example, we only observe small changes between the red polygon #1 in O_{old} and O_{new}; the additional scope of the O_{new} polygon is the union of red polygon #1 and #2 in Fig 3.c. The relative-novelty and the size of the novel polygons are listed in Table 1. We can see that red polygon #4, blue polygon #1 and blue polygon #2 in O_{new} have the relative-novelty of 100%, indicating that these regions are new regions that only exist in dataset O_{new}. This can be verified by comparing the contour polygons for O_{new} and O_{old} in Fig. 3.a. Furthermore, we observe novel negative correlation regions of significant size (62.66 and 54.74, repetitively) in South America (blue polygon #1) and New Zealand (blue polygon #2). In summary, as we have shown in this demonstration, the relationships between two datasets can be analyzed quantitatively with the help of change predicates that operate on polygons.

Table 1. Novelty and relative-novelty for polygons in Fig. 3.c

Red polygon #	0	1	2	3	4
|Novelty|	3.6	3.06	53.3	9.00	2.43
Relative-novelty	0.04	0.01	0.25	0.31	1
Blue polygon #	0	1	2		
|Novelty|	10.3	62.66	54.74		
Relative-novelty	0.13	1	1		

5 Related Work

Our change analysis approach relies on clustering analysis. In 2006, a framework for change description that perceived changes of clusters as changes of states in a state space was proposed by Fleder et al. [6]. A framework for tracking external and internal cluster transitions in a data stream was introduced by Spiliopoulou et al [7] in the same year. In 2007, a technique for mining evolutionary behavior of interaction graphs was proposed by Asur et al. [1]. In general, these methods [1, 4, 7] can detect many types of change patterns but they require that the identity of objects must be known or objects must be characterized by nominal attributes. The advantage of our approaches is that we can detect various types of changes in data with continuous attributes and unknown object identity.

Existing contour plotting algorithms can be seen as variations of two basic approaches: level curve tracing [8] and recursive subdivision [2]. Level curve tracing algorithms scan a grid and mark grid-cell boundaries that are passed by the level curve. Contour polygons are created by connecting the marked edges. Recursive subdivision algorithms start with a coarse initial grid and recursively divide grid cells that are passed by the level curve. DCONTOUR, uses level curve tracing.

6 Summary

Developing techniques for discovering change in spatial datasets is important and providing methods to detect change for continuous attributes and for objects that are not identified apriori are advantages of the techniques we describe here. In this paper, change analysis techniques that rely on comparing clusters for the old and new data based on a set of predicates are proposed. A novel contour clustering algorithm named DCONTOUR that combines supervised density functions with contouring algorithms is introduced to automate this task.

In general, our work is a first step towards analyzing complex change patterns. The novel contributions of this paper includes: 1) using density functions in contouring algorithm; 2) change analysis is conducted by interestingness comparison; 3) degrees of change are computed relying on polygon operations; 4) a novel change analysis approach is introduced that compares clusters that are derived from supervised density functions.

References

1. Asur, S., Parthasarathy, S., Ucar, D.: An Event-based Framework for Characterizing the Evolutionary Behavior of Interaction Graphs. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2007)
2. Bruss, A.R.: Extracting Topographic Features from Elevation Data using Contour Lines. Working Paper 146, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA (1977), http://hdl.handle.net/1721.1/41980
3. Snyder, W.V.: Algorithm 531: Contour Plotting [J6]. ACM Transactions on Mathematical Software (TOMS) (ACM) 4(3), 290–294 (1978)

4. Fleder, D., Padmanabhan, B.: Cluster Evolution and Interpretation via Penalties. In: Proceedings of the 6th IEEE International Conference on Data Mining – Workshops (2006)
5. Hinneburg, A., Keim, D.A.: An Efficient Approach to Clustering in Large Multimedia Databases with Noise. In: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (1998)
6. Jiang, D., Eick, C.F., Chen, C.-S.: On Supervised Density Estimation Techniques and Their Application to Clustering. In: Proceeding of the 15th ACM International Symposium on Advances in Geographic Information Systems (2007)
7. Spiliopoulou, M., Ntoutsi, I., Theodoridis, Y., Schult, R.: Monic – Modeling and Monitoring Cluster Transitions. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2006)
8. Watson, D.: Contouring — A Guide to the Analysis and Display of Spatial Data. In: Computer Methods in the Geosciences, vol. 10. Pergamon Press, Oxford (1992)

Centroid Neural Network with Spatial Constraints

Dong-Chul Park

Dept. of Information Engineering, Myong Ji University, Korea
parkd@mju.ac.kr

Abstract. A Centroid Neural Network with spatial constraints (CNN-S) is proposed in this paper. The spatial constraints are applied to the Centroid Neural Network(CNN) algorithm to reduce noise in Magnetic Resonance(MR) images. MR image segmentation is performed to illustrate the application of the proposed algorithm. The proposed algorithm incorporates a novel approach of using the weights of attributes to evaluate the roles of the latter. Experiments and results on MR images from Internet Brain Segmentation Repository(IBSR) show that the proposed algorithm provides a superior performance over other algorithms such as Fuzzy C-Means(FCM)and Fuzzy C-Means with spatial constraints(FCM-S).

1 Introduction

Much attention has been paid to image segmentation by many researchers [1] -[4]. Image segmentation is a technique of decomposing an image into meaningful parts, or objects. Each object is labeled in a way that facilitates the description of the original image so that it can be interpreted by the system that handles the image. In general, image segmentation classifies image's pixels as belonging to one of the "objects" composing the the image. It is widely used in practice such as in object detection and image classification applications. In the field of medicine, image segmentation is often used to analyze MR image.

Traditionally, in order to cluster an image into objects, algorithms such as Self Organizing Map(SOM)[5], k-means[6], Fuzzy c-means(FCM)[7], and Centroid Neural Network(CNN)[8] are the most widely used in practice. The k-means, SOM, CNN are common approaches to assign each object into one or many classes according to their features. k-means algorithm is one of the common unsupervised learning algorithms that solve the well known clustering problem. One strength of the k-means algorithm is fast in convergence and be able to apply for a large data. Another popular approach for data clustering problems is SOM algorithm. The main idea of SOM algorithm is to find winner neuron, which is the closest to a given datum, and to update the synaptic weights of winner and its neighbors. However, performance of SOM algorithm heavily depends on

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 915–922, 2009.
© Springer-Verlag Berlin Heidelberg 2009

the initial parameter values including the learning coefficient, the sequence of data presentation, and the number of iterations. One of the efficient clustering methods is FCM. FCM is a method of clustering which allows one piece of data to belong to two or more clusters. It requires each data in the data set to belong to each of the clusters with corresponding membership value.

In order to improve the unstableness of k-means and SOM, the Centroid Neural Network (CNN)[8] algorithm introduced the concepts of the reward and the punishment to the winner and the loser neurons. To have a better local minimum than k-means and SOM, CNN starts with setting the number of groups at 2 and increases the number of groups step by step until it reaches the predetermined number of groups. In most of experiments, CNN has shown a superior performance to k-means and some conventional algorithms such as SOM, DCL[9], and SLA[10].

The SOM, k-means, FCM and CNN are used to assign each object into one or many classes. These algorithms have basically solved the problem of clustering objects well. From experiments with MR images,however, we recognized that these algorithms yield less optimal solutions when images are contaminated with noise. In practice, when analyzing an MR image, the noise can affect the doctor's diagnosis and thus this problem poses a particular challenge to researchers.

Noise usually exists in audio signals and images and is a cause of ambiguousness. A generally accepted method utilizes the intensities of neighbor values to reduce noise in audio or images. From this concept, a clustering approach called Centroid Neural Network with spatial constraints (CNN-S) is proposed in this paper. The proposed CNN-S algorithm is devised to cope with the noises in image segmentation problems. The proposed CNN-S algorithm utilizes the weights of attributes to evaluate the roles of attributes on data. The main pixel of a block is given a higher weight than neighbor pixels.

The remainder of this paper is organized as follows: Section 2 presents a brief summary of the CNN algorithm. The CNN-S with spatial constraints algorithm is proposed in Section 3. Section 4 presents experiments and results on synthetic data and a real-world application. Section 5 concludes the paper.

2 Centroid Neural Network (CNN)

The CNN algorithm [8] is an unsupervised competitive learning algorithm based on the classical k-means clustering algorithm [6]. It finds the centroids of clusters at each presentation of the data vector. The CNN first introduces definitions of the winner neuron and the loser neuron. When a data $\mathbf{x}_i$ is given to the network at the epoch (k), the winner neuron at the epoch (k) is the neuron with the minimum distance to $\mathbf{x}_i$. The loser neuron at the epoch (k) to $\mathbf{x}_i$ is the neuron that was the winner of $\mathbf{x}_i$ at the epoch (k-1) but is not the winner of $\mathbf{x}_i$ at the epoch (k). The CNN updates its weights only when the status of the output neuron for the presenting data has changed when compared to the status from the previous epoch.

When an input vector $\mathbf{x}$ is presented to the network at epoch n, the weight update equations for winner neuron j and loser neuron i in CNN can be summarized as follows:

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \frac{1}{N_j + 1}[\mathbf{x}(n) - \mathbf{w}_j(n)] \tag{1}$$

$$\mathbf{w}_i(n+1) = \mathbf{w}_i(n) - \frac{1}{N_i - 1}[\mathbf{x}(n) - \mathbf{w}_i(n)] \tag{2}$$

where $\mathbf{w}_j(n)$and $\mathbf{w}_i(n)$ represent the weight vectors of the winner neuron and the loser neuron, iteration, respectively.

The CNN has several advantages over conventional algorithms such as SOM or k-means algorithm when used for clustering and unsupervised competitive learning. The CNN requires neither a predetermined schedule for learning gain nor the total number of iterations for clustering. It always converges to suboptimal solutions while conventional algorithms such as SOM may give unstable results depending on the initial learning gains and the total number of iterations. More detailed description on the CNN can be found in [8,11].

Note that the CNN was designed for deterministic data because the distance measure used in the CNN is the quadric (Euclidean) distance.

3 Centroid Neural Network with Spatial Constraints

The Centroid Neural Network with spatial constraints is based on the Centroid Neural Network algorithm and utilizes spatial constraints to increase the influence of neighbor vectors.

Assume that N data vectors,$x(i), i = 1..N$and the number of clusters,M, are given. Each of the datum is grouped as a member of one of the M clusters. Each of group is represented by prototype vector $\mathbf{w}_j$. Cluster i has N_i members and $\mathbf{x}_i(j)$ denotes the data j in the cluster i for one instance and $\overline{\mathbf{x}}_i(j)$ is the mean vector of neighbor vectors of $\mathbf{x}_i(j)$. The penalty of neighbor pixels to the main pixel is defined by the parameter α.

The object function of CNN-S can be rewritten as follows:

$$J = \sum_{i=1}^{M}\sum_{j=1}^{N_i} u_{ij}\|\mathbf{x}_i(j) - \mathbf{w}_i\|^2 + \alpha \sum_{i=1}^{M}\sum_{j=1}^{N_i} u_{ij}\left\|\overline{\mathbf{x}_i(j)} - \mathbf{w}_i\right\|^2 \tag{3}$$

$$u_{ij} = \begin{cases} 1, & \text{if } \mathbf{x}_i(j) \in \mathbf{w}_i \\ 0, & \text{otherwise} \end{cases}$$

where $\|.\|^2$ denotes Euclidean distance with weights. $\mathbf{x}_i(j), \overline{\mathbf{x}_i(j)}, \mathbf{w}_i$,and α are defined as the input vector, the mean vector of neighbor vectors around the input, a prototype of the cluster $\mathbf{w}_i$, and the penalty of neighbors pixels, respectively.

By minimizing the object function J, we obtain the equations of the prototypes as follows:

$$\mathbf{w}_i = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N_i} u_{ij}(\mathbf{x}_i(j) + \alpha\overline{\mathbf{x}_i(j)})}{(1+\alpha)\sum_{i=1}^{M}\sum_{j=1}^{N_i} u_{ij}} \tag{4}$$

From Eq.(4), we can derive the formulas to update the winner neuron j and the loser neuron i. That is, when a data vector $\mathbf{x}$ is applied to the the network at time n, the update equations for the winner neuron j and the loser neuron i in CNN-S can be written as follows:

– for the winner neuron j,

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \frac{1}{N_j+1}\left(\frac{\mathbf{x}(n) + \alpha\overline{\mathbf{x}}(n)}{1+\alpha} - \mathbf{w}_j(n)\right) \tag{5}$$

where $\mathbf{x}(n)$ and $\overline{\mathbf{x}}(n)$ are the data vector that the cluster j has just won and the mean vector of its neighbor vectors, respectively.

– for the loser neuron i,

$$\mathbf{w}_i(n+1) = \mathbf{w}_i(n) - \frac{1}{N_i-1}\left(\frac{\mathbf{x}(n) + \alpha\overline{\mathbf{x}}(n)}{1+\alpha} - \mathbf{w}_i(n)\right) \tag{6}$$

where $\mathbf{x}(n)$ and $\overline{\mathbf{x}(n)}$ are the data vector that the cluster i has just lost and the mean vector of its neighbor vectors, respectively.

In order to avoid getting stuck at an undesirable local minimum solution, the CNN-S algorithm starts at 2 clusters and increases the number of clusters one by one until it reaches the predetermined number of groups like the CNN algorithm. The CNN-S algorithm improves upon the CNN algorithm with respect to solving the noisy image segmentation problem. The result of the CNN-S algorithm depends on the parameter α, which is a penalty of neighbor pixels.

4 Experiments and Results

In this section, we describe experimental results on both synthetic data and a real-world problem. FCM, FCM-S [12,13] and CNN-S are used for the performance evaluation purpose.

In the experiments, the original image is divided into overlapping 3x3 blocks. In the data set, a heuristic is used: larger weights for the main pixel and smaller weights for neighbor pixels. Thus, the distance of two vectors is changed to the Euclidean distance with weights.

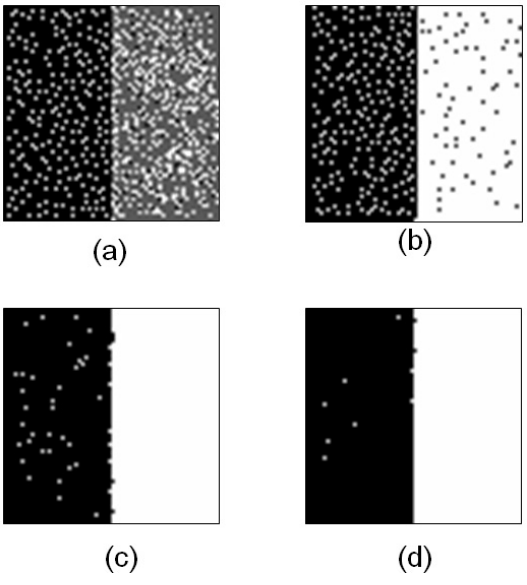


Fig. 1. (a) Original image and the results (b)by FCM, (c)by FCM-S, and (d) by CNN-S

Table 1. Comparison of accuracy of the three algorithms for Fig.1

	FCM	FCM-S	CNN-S
# of misclassified pixels	399	44	10
Accuracy(%)	43	93.71	98.57

4.1 Experiments on Synthetic Data

The synthetic data used for experiments consists of a 64x64 image. The image includes two classes with two intensity values, taken as 0 and 120. We tested the performance of the algorithms where the image data are corrupted by "salt and pepper" noise. This data set is similar to the data set described in [12]. The results are evaluated by the ratio of the number of pixels with correct classification over the total number of pixels. The performance results of the different algorithms and the segmentation images are shown in Table 1 and Fig.1, respectively.

Table 2. Accuracy vs.α values

	α values							
	0.125	0.250	0.375	0.500	0.675	0.750	0.875	1.00
FCM-S	89.57	93.71	93.43	93.14	92.86	92.57	90.71	90.57
CNN-S	98.57	98.57	98.57	96.71	90.14	90.14	90.14	90.14

Table 3. Comparison scores(%) of three algorithms for Fig.2

	Class 1	Class 2	Class 3	Total
FCM	82.88	51.8	65.11	68.17
FCM-S	89.05	64.34	80.92	80.46
CNN-S	96.73	75.46	80.10	85.75

On experiments with synthetic data, the algorithms were tested with different α values. For the overall classification results of the three different algorithms, the proposed CNN-S algorithm achieves 98.57% accuracy while the FCM and the FCM-S show 43% accuracy and 93.71% accuracy, respectively. The accuracy of each algorithm depends on the α value. The α value refers to the penalty of neighbor pixels to the main pixels. A higher α reflects greater influence of neighbor pixels. The clustering accuracies with various α values are shown in Table.2.

4.2 Experiments on Real-World Application

Experiments on MR images collected from the Internet Brain Segmentation Repository(IBSR) [14] are also performed. The test images are brain MR images

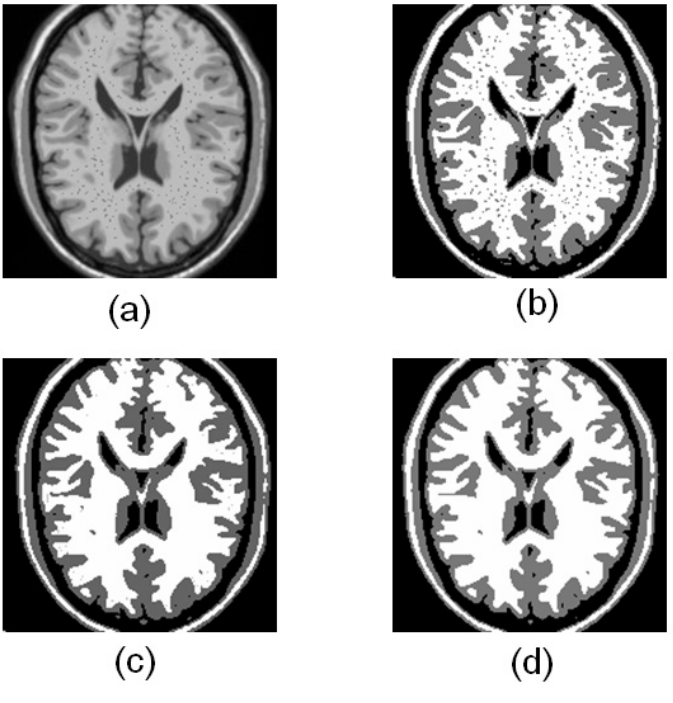


Fig. 2. Brain MR image. (a) Original image and the results (b)by FCM, (c)by FCM-S, and (d) by CNN-S.

with 3% noise. The image segmentation technique was used to cluster the MR image into 3 parts, i.e., gray matter, white matter, and cerebrospinal fluid, and comparison scores were calculated by the following ratio:

$$s_j = \frac{A_j \cap A_{ref\ j}}{A_j \cup A_{ref\ j}}$$

where A_j represents a set of pixels that belong to the j^{th} class and A_{refj} represents a set of pixels that belong to the j^{th} class in the reference segmented image. In the experiments, $\alpha = 0.8$ was used for all algorithms with spatial constraints.

Fig.2 and Table 3 present a comparison of image segmentation results on a real T1-weighted MR image. The scores in Table 3 evaluate the classification results of the algorithms of interest. As can be seen in Table 3, the proposed CNN-S algorithm outperforms FCM and FCM-S by 17.58% and 5.29%, respectively.

5 Conclusion

In this paper, the CNN-S algorithm is proposed for solving image segmentation when noise is present in images. The proposed CNN-S algorithm, based on the CNN algorithm, incorporates spatial constraints to increase the influence of neighbor pixels and reduce the noise in image. The proposed algorithm is applied to image segmentation problems where a certain noise is present in two types of images: a synthetic image and a MR image. The proposed CNN-S algorithm is compared with FCM and FCM-S, the proposed CNN-S outperforms FCM and FCM-S by 17.58% and 5.29% , respectively, in MRI experiments.

Acknowledgment

This work was supported by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korean government (MOST)(Grant No.: R01-2007-000-20330-0). This manuscript was prepared with the help of Tran Nhon and ICRL members.

References

1. Boskovitz, V., Guterman, H.: An adaptive neuro-fuzzy system for automatic Image Segmentation and Edge Dectection. IEEE Trans. Fuzzy Systems 10, 247–262 (2002)
2. Chen, S., Sandham, W., Granat, M., Sterr, A.: MRI fuzzy segmentation of brain tissue using neighborhood attraction with neural network optization. IEEE Trans. Information Technology in Biomedicine 9, 459–467 (2005)
3. Pednekar, A., Kakadiaris, I.A.: Image Segmentation Based on Fuzzy Connectedness Using Dynamic Weights. IEEE Trans. Image Processing 15, 1555–1562 (2006)
4. Yan, P., Kassim, A.A.: Medical Image Segmentation using minimal path deformable models with implicit shape priors. IEEE Trans. Information Technology in Biomedicine 10, 677–684 (2006)
5. Kohonen, T.: Self-Organization Map. Proc. IEEE 78, 1464–1480 (1990)

6. Hartigan, J.: Clustering Algorithm. Wiley, New York (1975)
7. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Fucntion Algorithms, New York (1981)
8. Park, D.C.: Centroid Neural Network for Unsupervised Competitive Learning. IEEE Trans. Neural Networks 11, 520–528 (2000)
9. Kong, S., Kosko, B.: Differential competitive learning for centroid estimation and phoneme recognition. IEEE Trans. Neural Networks 2, 118–124 (1991)
10. Tsypkin, Y.: Foundations of the Theory of Learning Systems. Academic, New York (1973)
11. Park, D.C., Woo, Y.: Weighted Centroid Neural Network for Edge Reserving Image Compression. IEEE Trans. Neural Networks 12, 1134–1146 (2001)
12. Chen, S., Zhang, D.: Robust image segmentation using FCM with spatial constraints based on new kernel-induced distance measure Systems. IEEE Trans. on Man and Cybernetics-Part B 34, 1907–1916 (2004)
13. Cai, W., Chen, S., Zhang, D.: Fast and Robust Fuzzy C-Means Clustering Algorithms Incorporating Local Information for Image Segmentation. Pattern Recognition 40(3), 825–838
14. `http://www.cma.mgh.harvard.edu/ibsr/`

Diversity in Combinations of Heterogeneous Classifiers

Kuo-Wei Hsu and Jaideep Srivastava

University of Minnesota, Minneapolis, MN, USA
{kuowei, srivastava}@cs.umn.edu

Abstract. In this paper, we introduce the use of combinations of heterogeneous classifiers to achieve better diversity. Conducting theoretical and empirical analyses of the diversity of combinations of heterogeneous classifiers, we study the relationship between heterogeneity and diversity. On the one hand, the theoretical analysis serves as a foundation for employing heterogeneous classifiers in Multi-Classifier Systems or ensembles. On the other hand, experimental results provide empirical evidence. We consider synthetic as well as real data sets, utilize classification algorithms that are essentially different, and employ various popular diversity measures for evaluation. Two interesting observations will contribute to the future design of Multi-Classifier Systems and ensemble techniques. First, the diversity among heterogeneous classifiers is higher than that among homogeneous ones, and hence using heterogeneous classifiers to construct classifier combinations would increase the diversity. Second, the heterogeneity primarily results from different classification algorithms rather than the same algorithm with different parameters.

Keywords: Multi-Classifier System, ensemble, diversity, heterogeneity.

1 Introduction

Multi-Classifier System (MCS) has gained much attention in pattern recognition, machine learning, and data mining [11, 20]. The key concept of MCS is a combination of different classifiers. It combines predictions from base classifiers into the final predictions [9] and so does an ensemble. MCS is gaining popularity because of the limits of every individual classification algorithm. Techniques proposed to create MCSs or ensembles can be broadly categorized into three categories, according to how base classifiers are constructed. The first category is to use various subsets of training data to train base classifiers, such as boosting [10, 21] and bagging (bootstrap aggregating) [6]. The second category is to use different feature sets to train base classifiers, such as random forest (RF) [7]. Finally, the third category of techniques for the MCS or ensemble construction is to employ different algorithms to build systems composed of heterogeneous classifiers. In fact, these categories represent not only ways to combine classifiers but also ways to achieve diversity among them.

It has been shown that in practice diversity is an important factor that makes them successful. Diversity between two classifiers indicates how different the predictions made by one classifier will be from those made by the other. In this paper, we present

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 923–932, 2009.
© Springer-Verlag Berlin Heidelberg 2009

theoretical and empirical analyses of classifier combination that is composed of heterogeneous classifiers. We introduce the use of heterogeneous classifiers in MCSs or ensembles, since the classifiers are expected to be uncorrelated and to behave independently from each other. We provide evidence supporting that the diversity among heterogeneous classifiers is better than that among homogeneous ones. We consider four synthetic data sets as well as six real data sets, utilize three of the most important data mining algorithms [25] with or without alternative parameter sets, and evaluate the results using ten well-known diversity measures.

To the best of our knowledge, there exist quite a few papers that theoretically examine the diversity of heterogeneous classifiers and make empirical comparisons of the diversity between homogeneous classifiers to that between heterogeneous classifiers. In our theoretical analysis, we introduce two definitions for diversity based on disagreements. These definitions distinguish themselves from others in the sense that they consider the nature of the underlying algorithm and the training data set as well. Neglecting these two factors, other definitions and measures for diversity failed in investigating effects of the use of heterogeneous classifiers in ensembles. They could not explain the phenomena presented in Section 3. Our contributions are listed below:

1). Two definitions for diversity introduced in this paper could assist researchers in understanding the diversity in MCSs or ensembles more comprehensively.

2) We present a lower bound of the probability that using heterogeneous classifiers would give better diversity when underlying algorithms are different enough.

3) We show that using heterogeneous classifiers consistently provides better diversity regardless of the diversity measures employed in experiments.

4) It is demonstrated that using alternative parameters lead to changes of diversity between homogeneous classifiers but changes are not as significant as those of diversity between heterogeneous classifiers.

The remainder of this paper is organized as follows. In Section 2 we will present the theoretical analysis for diversity between heterogeneous classifiers, while in Section 3 we will report the experiments and results. Next, we will discuss related work in Section 4. Finally, conclusions will be given in Section 5.

2 Diversity of Combinations of Heterogeneous Classifiers

Initially, we introduce notations used in this paper. We use A and S to respectively denote a symmetric classification algorithm and a data set. The i-th sample in a data set S is $z_i = (x_i, y_i)$, $1 \le i \le |S|$, while x_i is a vector of features and y_i is the class label. For convenience and clarity, $X = \{x_i\}$ while $Z = \{z_i\}$ and $Z \subseteq S$. In this paper we consider binary classification problem and hence $y_i = \{-1,+1\}$ and $Y = \{y_i\}$. Moreover, $\hat{y}_i^{(A,S)}$ is the prediction obtained from applying A on S, while I and E_S are respectively the indicator function and the expectation operator with respect to a set S. Below we define two types of diversity. The first one is the intra-algorithm diversity, which is the normalized number of disagreements caused by the nature of the algorithm when dealing with different training sets.

Definition 1. *D is an underlying distribution (which is unknown in practice) and three subsets drawn from it are denoted as S_1, S_2 and S. If S_1 and S_2 are training sets while S is a test set, then the intra-algorithm diversity of a symmetric classification*

algorithm A *is* $\mathrm{E}_S\left[\mathrm{I}(\hat{y}^{(A,S_1)} \neq \hat{y}^{(A,S_2)})\right]$, *an expectation of disagreements with respect to the test set* S.

$\hat{y}_i^{(A,S_1)} \neq \hat{y}_i^{(A,S_2)}$ indicates a disagreement in the prediction of the class label of the i-th sample. For the i-th sample, $\hat{y}_i^{(A,S_1)}$ is the prediction from the classifier created by applying A on S_1, while $\hat{y}_i^{(A,S_2)}$ is that from the classifier created by applying A on S_2. Assume $|S_1| = |S_2| = N$, $|S_1 \cap S_2| = c \cdot |S_1| = c \cdot |S_2| = c \cdot N = n$, where $0 < c < 1$ (which implies $S_1 \cap S_2 \neq \varnothing$). The constant c is used to present the proportion of common samples in both S_1 and S_2. In order not to make the above definition is too strong to be practical, we need restrictions for S, such as $|S_1 \cap S_2| = |S_1 \cap S| = |S_2 \cap S| = c \cdot N = n$. Thus, a simple setting is $S = S_1 \cap S_2$.

Definition 2. *A symmetric classification algorithm* A *is* (α, β)*-stable with respect to the intra-algorithm diversity if* $\Pr\left[\mathrm{E}_{S_1 \cap S_2}[\mathrm{I}(\hat{y}^{(A,S_1)} \neq \hat{y}^{(A,S_2)})] \leq \alpha\right] \geq \beta$ *where* $0 \leq \alpha, \beta \leq 1$.

The (α, β)-stability for the intra-algorithm diversity can be interpreted as follows: α is the upper bound for the normalized number of disagreements under a probability at least β. The second type of diversity is inter-algorithm diversity. It is defined as the normalized number of disagreements due to diverse natures of individual algorithms.

Definition 3. D *is an underlying distribution (which is unknown in practice) and two subsets drawn from it are denoted as* S *and* S'. *If* S *is a training set and* S' *is a test set, the inter-algorithm diversity of two symmetric classification algorithms* A_1 *and* A_2 *is* $\mathrm{E}_{S'}[\mathrm{I}(\hat{y}^{(A_1,S)} \neq \hat{y}^{(A_2,S)})]$ *where* $S' \subseteq S \sim D$ *and* $S' \neq \varnothing$.

$\hat{y}_i^{(A_1,S)} \neq \hat{y}_i^{(A_2,S)}$ indicates a disagreement, with respect to the class label of the i-th sample in S', between the prediction given by a classifier from applying A_1 on S and that given by a classifier from applying A_2 on S. In other words, we apply two algorithms on the whole data set S to build two classifiers, and the diversity between them in this context is defined as the disagreements between these two classifiers for an arbitrary non-empty subset S' of S. What is more, we employ the inter-algorithm diversity to define the differentiability for two classification algorithms.

Definition 4. Two s*ymmetric classification algorithms* A_1 *and* A_2 *are* (δ, γ)*-differentiable if* $\Pr\left[\mathrm{E}_{S'}[\mathrm{I}(\hat{y}^{(A_1,S)} \neq \hat{y}^{(A_2,S)})] \geq \delta\right] \geq \gamma$, *where* $0 \leq \delta, \gamma \leq 1$.

A_1 and A_2 are (δ, γ)-differentiable if δ is the lower bound for the number of disagreements under a probability at least γ. This definition is used to set an assumption that A_1 and A_2 are really different. Afterwards, we consider the combinations of homogeneous and heterogeneous algorithms. We denote a combination of classifiers as $A_a + A_b$ where subscripts a and b indicate the underlying algorithms. Diversities of $A_1 + A_1$ and $A_2 + A_2$ are $\mathrm{E}_{S_1 \cap S_2}\left[\mathrm{I}(\hat{y}^{(A_1,S_1)} \neq \hat{y}^{(A_1,S_2)})\right]$ and $\mathrm{E}_{S_1 \cap S_2}[\mathrm{I}(\hat{y}^{(A_2,S_1)} \neq \hat{y}^{(A_2,S_2)})]$, respectively. The diversity between heterogeneous classifiers can be written as $\mathrm{E}_{S_1 \cap S_2}[\mathrm{I}(\hat{y}^{(A_1,S_1)} \neq \hat{y}^{(A_2,S_2)})]$.

Proposition. *If symmetric classification algorithms* A_1 *and* A_2 *are respectively* (α_1, β_1)*-stable and* (α_2, β_2)*-stable with respect to intra-algorithm diversity, while* A_1 *and* A_2 *are* (δ_2, γ_2)*-differentiable, then the following holds:*

$$\Pr\left[\left(\mathrm{E}_{S_1\cap S_2}[\mathrm{I}(\hat{y}^{(A_1,S_1)} \neq \hat{y}^{(A_2,S_2)})] - \mathrm{E}_{S_1\cap S_2}[\mathrm{I}(\hat{y}^{(A_1,S_1)} \neq \hat{y}^{(A_1,S_2)})]\right) \geq \delta_2 - 2\cdot\alpha_1\right] \geq \gamma_2 \cdot \beta_1 \quad (1)$$

If $\delta_2 > 2\cdot\alpha_1$, then the diversity of $A_1 + A_2$ will be larger than the diversity of $A_1 + A_1$ with a probability of at least $\gamma_2 \cdot \beta_1$.

Eq. (1) shows the difference between the diversity from using heterogeneous algorithms and that from using homogeneous ones. First of all, let us focus on the first term $\mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_2,S_2)})$, the sum of $\mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_2,S_2)} \wedge \hat{y}_i^{(A_1,S_1)} = \hat{y}_i^{(A_1,S_2)})$ and $\mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_2,S_2)} \wedge \hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)})$ since one of them is true. Notice that we consider only binary classification problems. Furthermore, $\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}$ when $\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_2,S_2)}$ and $\hat{y}_i^{(A_1,S_1)} = \hat{y}_i^{(A_1,S_2)}$, so that $\mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_2,S_2)} \wedge \hat{y}_i^{(A_1,S_1)} = \hat{y}_i^{(A_1,S_2)})$ is equal to $\mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}) \cdot \mathrm{I}(\hat{y}_i^{(A_1,S_1)} = \hat{y}_i^{(A_1,S_2)})$; furthermore, $\hat{y}_i^{(A_1,S_2)} = \hat{y}_i^{(A_2,S_2)}$ when $\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_2,S_2)}$ and $\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)}$, so that $\mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_2,S_2)} \wedge \hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)})$ is equal to $\mathrm{I}(\hat{y}_i^{(A_1,S_2)} = \hat{y}_i^{(A_2,S_2)}) \cdot \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)})$. So, $\mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_2,S_2)})$ gives Eq. (2).

$$\mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}) \cdot \mathrm{I}(\hat{y}_i^{(A_1,S_1)} = \hat{y}_i^{(A_1,S_2)}) + \mathrm{I}(\hat{y}_i^{(A_1,S_2)} = \hat{y}_i^{(A_2,S_2)}) \cdot \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)}) \quad (2)$$

Next, $\mathrm{E}_{S_1\cap S_2}[\mathrm{I}(\hat{y}^{(A_1,S_1)} \neq \hat{y}^{(A_2,S_2)})] - \mathrm{E}_{S_1\cap S_2}[\mathrm{I}(\hat{y}^{(A_1,S_1)} \neq \hat{y}^{(A_1,S_2)})]$

$$= \frac{1}{n}\sum_{i=1}^{n} \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_2,S_2)}) - \frac{1}{n}\sum_{i=1}^{n} \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)})$$

$$= \frac{1}{n}\sum_{i=1}^{n} \left\{ \begin{array}{l} \mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}) \cdot \mathrm{I}(\hat{y}_i^{(A_1,S_1)} = \hat{y}_i^{(A_1,S_2)}) \\ + \mathrm{I}(\hat{y}_i^{(A_1,S_2)} = \hat{y}_i^{(A_2,S_2)}) \cdot \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)}) - \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)}) \end{array} \right\} \text{ from Eq. (2)}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \left\{\mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)})\right\} + \frac{1}{n}\sum_{i=1}^{n} \left\{-2\cdot\mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}) \cdot \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)})\right\} \quad (3)$$

Under the assumption that A_1 and A_2 are (δ_2, γ_2)-differentiable the probability that the first term in Eq. (3) will be larger than or equal to δ_2 is at least γ_2. Considering the second term in Eq. (3) and $\mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}) = \{0,1\}$, we have

$$\sum_{i=1}^{n} \mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}) \cdot \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)}) \leq \sum_{i=1}^{n} 1 \cdot \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)})$$

We assume A_1 is (α_1, β_1)-stable for intra-algorithm diversity and get the probability that the second term will be larger than or equal to $-2\alpha_1$ is at least β_1.

$$\frac{1}{n}\sum_{i=1}^{n} \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_2,S_2)}) - \frac{1}{n}\sum_{i=1}^{n} \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)})$$

$$= \frac{1}{n}\sum_{i=1}^{n} \mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}) - \frac{2}{n}\sum_{i=1}^{n} \mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}) \cdot \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)})$$

$$\geq \frac{1}{n}\sum_{i=1}^{n} \mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}) - \frac{2}{n}\sum_{i=1}^{n} \mathrm{I}(\hat{y}_i^{(A_1,S_1)} \neq \hat{y}_i^{(A_1,S_2)}) \text{ since } \mathrm{I}(\hat{y}_i^{(A_1,S_2)} \neq \hat{y}_i^{(A_2,S_2)}) = \{0,1\}$$

$\geq \delta_2 - 2\cdot\alpha_1$ with probability at least $\gamma_2 \cdot \beta_1$, since they are independent events.

Therefore, the diversity given by heterogeneous classifiers is larger than that given by homogeneous classifiers by at least $\delta_2 - 2 \cdot \alpha_1$ with a probability at least $\gamma_2 \cdot \beta_1$. In order to make the above inequality interesting and useful, we need $\delta_2 \geq 2 \cdot \alpha_1$. That is, we need to make sure that the inter-algorithm diversity is at least twice larger than the intra-algorithm diversity of A_1. In summary, according to the above analysis, if we want to obtain better diversity by using heterogeneous classifiers, we should focus on finding an algorithm that is as different from the first algorithm as possible.

3 Experiments and Results

In this paper, we perform experiments with synthetic and real data sets. We exploit two data generators, RDG1 and RandomRBF built in Weka [23], to individually generate two synthetic data sets. We also use six UCI benchmark data sets as real data sets: *breast-w*, *credit-a*, *credit-g*, *diabetes*, *kr-vs-kp*, and *sick* [3, 12].

Diversity measures are proposed to assist in MCS or ensemble design [17]. In [14], authors give a comprehensive summary of diversity measures. Here, we employ ten popular measures to establish the quantitative determination of diversity between classifiers. Diversity measures considered here are summarized below: Q-statistic (*Q*), correlation coefficient (ρ), disagreement measure (*DIS*), double-fault measure (*DF*), entropy (*E*), Kohavi-Wolpert variance (*KW*), interrater agreement (κ), measure of difficulty (θ), generalized diversity (*GD*), and coincident failure diversity (*CFD*).

Furthermore, we consider three disparate algorithms: C4.5 decision tree [19] (named J48 in Weka), Naïve Bayes [13] (NB for short), and nearest neighbor [1] (named IBk in

Table 1. Summary of ten popular diversity measures [16]

Definition	Notations
$Q = (N^{11}N^{00} - N^{01}N^{10}) \cdot (N^{11}N^{00} + N^{01}N^{10})$	N is the number of samples. N^{11}, N^{00}, N^{10}, N^{01} present the number of samples for which both, none, only the1st, only the 2nd classifier(s) made correct prediction, respectively.
$\rho = \frac{N^{11}N^{00} - N^{01}N^{10}}{\sqrt{(N^{11}+N^{10}) \cdot (N^{01}+N^{00}) \cdot (N^{11}+N^{01}) \cdot (N^{10}+N^{00})}}$	
$DIS = (N^{01}+N^{10}) \cdot (N^{11}+N^{10}+N^{01}+N^{00})^{-1}$	
$DF = N^{00} \cdot (N^{11}+N^{10}+N^{01}+N^{00})^{-1}$	
$E = 1/N \sum_{j=1}^{N} \min(l_j, 2-l_j)$	l_j is the number of classifiers that make correct prediction for a sample j. $\overline{p}$ is the average accuracy of all classifiers.
$KW = 1/4N \sum_{j=1}^{N} l_j \cdot (2-l_j)$	
$\kappa = 1-(0.5 \cdot \sum_{j=1}^{N} l_j \cdot (2-l_j)) \cdot (N \cdot \overline{p} \cdot (1-\overline{p}))^{-1}$	
$\theta = \mathrm{var}(X)$ where random variable X denotes the portion of classifiers that make correct prediction for a random sample; That is, $X = \{0, 0.5, 1\}$	
$GD = 1-(P_2(0.5 \cdot P_1 + P_2))^{-1}$	P_i denotes the probability that i randomly selected classifiers will make incorrect prediction
$CFD = P_1 \cdot (1-P_0)^{-1}$ (or 0 if $P_0 = 1$)	

Weka). We select any two of them to create a pair of heterogeneous classifiers so that, for each data set, we set up six experimental sets in each of which we compare diversity of the combination of homogeneous classifiers with that of heterogeneous ones.

As for experiments, we adopt the following procedure to perform experiments, given an input dataset D and algorithms A_1 and A_2: First of all, we randomly draw samples from D without replacement and generate two training datasets. For synthetic datasets, the ratio of a training dataset to the whole dataset (D) is 0.1; for real datasets, it is 0.5. Next, we use one training datasets to create the first classifiers based on an algorithm A_1 , which could be J48, NB, or IBk. We denote this dataset as C_1 . Afterwards, we use the other training dataset to create the second classifier, C_2 , based on the same algorithm A_1 . Next, we create the third classifier, C_3 , by using the second training dataset and another algorithm A_2 (where $A_1 \neq A_2$). Following that, we draw samples from D with replacement and produce ten testing datasets. Then, for each testing dataset, we collect predicted class labels given by C_1 , C_2 , and C_3 as well. Next, we calculate the diversity between C_1 and C_2 (i.e., homogeneous classifiers) and also the diversity between C_1 and C_3 (i.e., heterogeneous classifiers) in ten diversity measures. Finally, we average the diversity values over ten testing datasets.

In the following, we present the results obtained by using the above three classification algorithms with default parameters first and then we present the results obtained by using the algorithms with various parameter sets.

Applying the above procedure to synthetic and real data sets, we collect the diversity values obtained from heterogeneous and homogeneous classifiers. Table A1 and Table A2 show the results for synthetic data sets and real data sets, respectively. For both Table A1 and Table A2, the first column exhibits a group of six experimental sets with respect to data sets. The second column shows the algorithms used in an experimental set. For instance, J48+J48 presents a pair of homogeneous classifiers, while J48+NB presents a pair of heterogeneous classifiers where J48 is the first employed algorithm and NB (i.e., Naïve Bayes) is the second one. For convenience, symbols ↓ and ↑ mean that respectively a higher value and a lower value will give a better diversity with respect to some measure. The results clearly present that using heterogeneous classifiers (shaded rows) leads to better diversity regardless of the diversity measures used in experiments.

According to the above theoretical analysis, the heterogeneity primarily comes from using different algorithms instead of using the same algorithm with different parameters. Here, we provide empirical support for this argument. However, it is impractical and unnecessary to study all possible combinations of parameters. Thus, we consider five quite different parameter sets, as listed in below, to increase the variability of classifiers that are from the same algorithm. Such a selection, or parameter tuning, is a common exercise in data mining.

1) J48: unpruned tree; NB: kernel density estimator; IBK: 3-nearest neighbor
2) J48: minimum 5 instances per leaf; NB: supervised discretization; IBk: 5-nearest neighbor
3) J48: 5-fold reduced error pruning; NB: kernel density estimator; IBk: 5-nearest neighbor, weighted by the inverse of distance
4) J48: confidence threshold 0.2 for pruning, minimum 5 instances per leaf; NB: supervised discretization; IBk: 5-nearest neighbors, weighted by 1-distance
5) J48: unpruned tree, binary splits, minimum 5 instances per leaf; NB: supervised discretization; IBk: 5-nearest neighbors, hold-one-out evaluation for training, minimizing mean squared error

We applied algorithms with these parameter sets on all data sets we mentioned earlier. However, due to the limitation of space, we do not report all results; rather, we present here the results for a synthetic data set and a real data set. Table A3 and Table A4 give the results of applying such pairs of homogeneous classifiers to a synthetic data set and the real data set *diabetes*, respectively. The first column shows parameter sets and the second column shows experiments for homogeneous and heterogeneous classifiers. The superscript asterisk means that, the pair of homogeneous classifiers is constructed with different parameters while one of them (the second one) comes with alternative parameters corresponding. The results presented in Tables A3 and A4 are not as optimistic as those reported in Tables A1 and A2. For the combination of homogeneous classifiers, using different parameters indeed gives better diversity. Changing parameters does not mean changing the nature of an algorithm but the way the algorithm searches the hypothesis space, if we interpret it in the classical language of machine learning. However, the diversity among classifiers that are based on the same algorithm but come with different parameters would not be good enough to differentiate them. From these results, in the setting considered here, the primary source of heterogeneity is the mix of different algorithms rather than the use of the same algorithm with different parameters. Nevertheless, the conclusion by no means indicates that employing different parameters has no effect on the diversity. It will be interesting to study the theoretical relationship between diversity and this factor.

4 Related Work

The study of diversity in ensemble has gained increasing attention, even though in theory there is no strong connection between diversity and the overall accuracy [5, 8, 14, 15, 16, 18, 20, 23].In [18] authors indicate that, in general, diversity compensates for errors made by individual classifiers. However, diversity itself is not a strong predictor of the overall accuracy of an ensemble [17]. In [15] authors discuss the relationship between diversity and accuracy, while in [23] it is indicated that an effective ensemble requires each of individual classifiers to offer high accuracy and to generate diverse errors. Additionally, in [22], authors demonstrate that boosting requires stronger diversity than does bagging while bagging does not depend only on diversity, and they argue that diversity depends on the size of training data set. Moreover, in [2] authors consider using different feature sets argue that using different feature sets is the only way to achieve diversity in a system of homogeneous classifiers. However, the argument is not necessarily true because using homogeneous classifiers with different parameters would lead to the change of diversity, as we can see in tables. Furthermore, we connect heterogeneity to diversity without considering the effects of using different feature sets. In [4] authors study the combination of heterogeneous classifiers with the focus on some combination methods. Nevertheless, neither a theoretical analysis nor an empirical investigation of the source of heterogeneity is performed in the paper.

5 Conclusions

This paper studied theoretically and empirically the relationship between heterogeneity and diversity. We performed a rich set of experiments to provide empirical evidence.

To evaluate the results, we considered four synthetic data sets as well as six real benchmark data sets, utilized three classification algorithms without and with five different parameter sets, and employed ten popular diversity measures. Consequently, we built a foundation for the use of heterogeneous classifiers in MCSs or ensembles. This is particularly essential because there are quite a few papers theoretically examining the diversity of classifier combinations and, at the same time, empirically comparing the diversity of the combination of homogeneous classifiers with that of heterogeneous classifiers. Two important observations in this paper will make substantial contributions to the future MCS or ensemble design. First, the diversity among heterogeneous classifiers is higher than that among homogeneous ones. Second, the heterogeneity mainly results from using different classification algorithms instead of using the same algorithm with different parameters. Future work includes the study of the relationship between heterogeneity and accuracy.

References

1. Aha, D., Kibler, D.: Instance-based learning algorithms. Machine Learning 6(1), 37–66 (1991)
2. Alkoot, F.M., Kittler, J.: Multiple expert system design by combined feature selection and probability level fusion. In: Proc. of the 3rd International Conference on Information Fusion, vol. 2, pp. THC5/9–THC516 (2000)
3. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html
4. Bahler, D., Navarro, L.: Methods for Combining Heterogeneous Sets of Classifiers. In: The 17th National Conference on Artificial Intelligence, Workshop on New Research Problems for Machine Learning (2000)
5. Banfield, R.E., Hall, L.O., Bowyer, K.W., Kegelmeyer, W.P.: A New Ensemble Diversity Measure Applied to Thinning Ensembles. In: International Workshop on Multiple Classifier Systems, pp. 306–316 (2003)
6. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
7. Breiman, L.: Random Forests. Machine Learning 45(1), 5–32 (2001)
8. Brown, G., Wyatt, J., Harris, R., Yao, X.: Diversity creation methods: a survey and categorization. Information Fusion 6(1), 5–20 (2005)
9. Dietterich, T.G.: Ensemble Methods in Machine Learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000)
10. Freund, Y., Schapire, R.E.: Experiments with a New Boosting Algorithm. In: Proc. of the 13th International Conference on Machine Learning, pp. 148–156 (1996)
11. Ghosh, J.: Multiclassifier Systems: Back to the Future. In: Roli, F., Kittler, J. (eds.) MCS 2002. LNCS, vol. 2364, pp. 1–15. Springer, Heidelberg (2002)
12. Hettich, S., Bay, S.D.: The UCI KDD Archive. University of California, Department of Information and Computer Science, Irvine, CA (1999), http://kdd.ics.uci.edu
13. John, G.H., Langley, P.: Estimating Continuous Distributions in Bayesian Classifiers. In: The 11th Conference on Uncertainty in Artificial Intelligence, pp. 338–345 (1995)
14. Kuncheva, L.I., Whitaker, C.J.: Ten measures of diversity in classifier ensembles: limits for two classifiers. In: A DERA/IEE Workshop on Intelligent Sensor Processing, pp. 10/1–10/10 (2001)

15. Kuncheva, L.I., Skurichina, M., Duin, R.P.W.: An experimental study on diversity for bagging and boosting with linear classifiers. Information Fusion 3(4), 245–258 (2002)
16. Kuncheva, L.I., Whitaker, C.J.: Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy. Machine Learning 51(2), 181–207 (2003)
17. Kuncheva, L.I.: That elusive diversity in classifier ensembles. In: Proc. of Iberian Conference on Pattern Recognition and Image Analysis, pp. 1126–1138 (2003)
18. Opitz, D., Maclin, R.: Popular ensemble methods: An empirical study. Journal of AI Research 11, 169–198 (1999)
19. Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo (1993)
20. Ranawana, R.: Multi-Classifier Systems - Review and a Roadmap for Developers. International Journal of Hybrid Intelligent Systems 3(1), 35–61 (2006)
21. Schapire, R.E.: The boosting approach to machine learning: An overview. In: MSRI Workshop on Nonlinear Estimation and Classification (2002)
22. Skurichina, M., Kuncheva, L., Duin, R.P.: Bagging and Boosting for the Nearest Mean Classifier: Effects of Sample Size on Diversity and Accuracy. In: Roli, F., Kittler, J. (eds.) MCS 2002. LNCS, vol. 2364, pp. 62–71. Springer, Heidelberg (2002)
23. Valentini, G., Masulli, F.: Ensembles of Learning Machines. In: Marinaro, M., Tagliaferri, R. (eds.) WIRN 2002. LNCS, vol. 2486, pp. 3–22. Springer, Heidelberg (2002)
24. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
25. Wu, X., Kumar, V., Quinlan, J.R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.-H., Steinbach, M., Hand, D.J., Steinberg, D.: Top 10 Algorithms in Data Mining. Knowledge and Information Systems 14(1), 1–37 (2008)

Appendix

Table A1. Average diversity values obtained from combinations of heterogeneous classifiers (shaded rows) and those of homogeneous classifiers on four synthetic data sets

Exp.	Algo.	$Q\downarrow$	$\rho\downarrow$	$DIS\uparrow$	$DF\downarrow$	$E\uparrow$	$KW\uparrow$	$\kappa\downarrow$	$\theta\downarrow$	$GD\uparrow$	$CFD\uparrow$
Set 1	J48+J48	1	1	0	0.225	0	0	1	0.1175	0	0
	J48+NB	0.38	0.1375	0.3375	0.1125	0.3375	0.085	0.325	0.05	0.64	0.775
Set 2	J48+J48	1	1	0	0.225	0	0	1	0.1175	0	0
	J48+IBk	0.705	0.2575	0.2	0.0625	0.2	0.05	0.595	0.0975	0.6625	0.7925
Set 3	NB+NB	1	1	0	0.3425	0	0	1	0.085	0	0
	NB+J48	0.38	0.1375	0.3375	0.1125	0.3375	0.085	0.325	0.05	0.64	0.775
Set 4	NB+NB	1	1	0	0.3425	0	0	1	0.085	0	0
	NB+IBk	0.32	0.0875	0.3375	0.0525	0.3375	0.085	0.32	0.065	0.79	0.88
Set 5	IBk+IBk	1	1	0	0.1025	0	0	1	0.1625	0	0
	IBk+J48	0.705	0.2575	0.2	0.0625	0.2	0.05	0.595	0.0975	0.6625	0.7925
Set 6	IBk+IBk	1	1	0	0.1025	0	0	1	0.1625	0	0
	IBk+NB	0.32	0.0875	0.3375	0.0525	0.3375	0.085	0.32	0.065	0.79	0.88

Table A2. Average diversity values obtained from combinations of heterogeneous classifiers (shaded rows) and those of homogeneous classifiers on six real data sets.

Exp.	Algo.	$Q\downarrow$	$\rho\downarrow$	$DIS\uparrow$	$DF\downarrow$	$E\uparrow$	$KW\uparrow$	$\kappa\downarrow$	$\theta\downarrow$	$GD\uparrow$	$CFD\uparrow$
Set 1	J48+J48	1	1	0	0.095	0	0	1	0.1717	0	0
	J48+NB	0.8133	0.32	0.1367	0.0533	0.1367	0.0333	0.73	0.1217	0.6483	0.7633
Set 2	J48+J48	1	1	0	0.095	0	0	1	0.1717	0	0
	J48+IBk	0.8583	0.3117	0.0983	0.0333	0.0983	0.0233	0.8033	0.1483	0.6633	0.79
Set 3	NB+NB	1	1	0	0.15	0	0	1	0.1417	0	0
	NB+J48	0.8133	0.32	0.1367	0.0533	0.1367	0.0333	0.73	0.1217	0.6483	0.7633
Set 4	NB+NB	1	1	0	0.15	0	0	1	0.1417	0	0
	NB+IBk	0.665	0.2	0.1517	0.035	0.1517	0.0367	0.6983	0.1233	0.75	0.8517
Set 5	IBk+IBk	1	1	0	0.0683	0	0	1	0.18	0	0
	IBk+J48	0.8583	0.3117	0.0983	0.0333	0.0983	0.0233	0.8033	0.1483	0.6633	0.79
Set 6	IBk+IBk	1	1	0	0.0683	0	0	1	0.18	0	0
	IBk+NB	0.665	0.2	0.1517	0.035	0.1517	0.0367	0.6983	0.1233	0.75	0.8517

Table A3. Average diversity values from combinations of homogeneous (HO*) and heterogeneous (HE) classifiers (HE) on a synthetic data set

Para. sets	Exp.	$Q\downarrow$	$\rho\downarrow$	$DIS\uparrow$	$DF\downarrow$	$E\uparrow$	$KW\uparrow$	$\kappa\downarrow$	$\theta\downarrow$	$GD\uparrow$	$CFD\uparrow$
1	HO*	1	0.8567	0.0733	0.3267	0.0733	0.02	0.8533	0.06	0.13	0.1867
	HE	0.34	0.16	0.4	0.1267	0.4	0.1	0.2	0.0333	0.6033	0.7467
2	HO*	0.9733	0.7367	0.13	0.31	0.13	0.03	0.74	0.0433	0.2133	0.3133
	HE	0.34	0.16	0.4	0.1267	0.4	0.1	0.2	0.0333	0.6033	0.7467
3	HO*	0.9267	0.8067	0.0833	0.29	0.0833	0.02	0.8333	0.0667	0.13	0.1967
	HE	0.34	0.16	0.4	0.1267	0.4	0.1	0.2	0.0333	0.6033	0.7467
4	HO*	0.9267	0.8067	0.0833	0.29	0.0833	0.02	0.8333	0.0667	0.13	0.1967
	HE	0.34	0.16	0.4	0.1267	0.4	0.1	0.2	0.0333	0.6033	0.7467
5	HO*	0.9767	0.89	0.05	0.3067	0.05	0.0133	0.9	0.0767	0.0733	0.12
	HE	0.36	0.1667	0.3883	0.125	0.3883	0.0967	0.2233	0.0333	0.6017	0.745

Table A4. Average diversity values from combinations of homogeneous (HO*) and heterogeneous (HE) classifiers (HE) on the real data set *diabetes*

Para. sets	Exp.	$Q\downarrow$	$\rho\downarrow$	$DIS\uparrow$	$DF\downarrow$	$E\uparrow$	$KW\uparrow$	$\kappa\downarrow$	$\theta\downarrow$	$GD\uparrow$	$CFD\uparrow$
1	HO*	0.97	0.79	0.0733	0.1833	0.0733	0.02	0.8533	0.09	0.1767	0.27
	HE	0.7533	0.3833	0.2067	0.1067	0.2067	0.0467	0.5967	0.0667	0.51	0.67
2	HO*	0.9067	0.5867	0.1467	0.15	0.1467	0.0367	0.71	0.07	0.3333	0.4867
	HE	0.7533	0.3833	0.2067	0.1067	0.2067	0.0467	0.5967	0.0667	0.51	0.67
3	HO*	0.8767	0.6033	0.1267	0.1433	0.1267	0.0333	0.74	0.0833	0.3167	0.4633
	HE	0.7533	0.3833	0.2067	0.1067	0.2067	0.0467	0.5967	0.0667	0.51	0.67
4	HO*	0.9067	0.5867	0.1467	0.15	0.1467	0.0367	0.71	0.07	0.3333	0.4867
	HE	0.7533	0.3833	0.2067	0.1067	0.2067	0.0467	0.5967	0.0667	0.51	0.67
5	HO*	0.9367	0.63	0.13	0.1567	0.13	0.0333	0.74	0.0733	0.3033	0.4567
	HE	0.7533	0.3833	0.2067	0.1067	0.2067	0.0467	0.5967	0.0667	0.51	0.67

Growth Analysis of Neighbor Network for Evaluation of Damage Progress

Ken-ichi Fukui[1], Kazuhisa Sato[2], Junichiro Mizusaki[2], Kazumi Saito[3], Masahiro Kimura[4], and Masayuki Numao[1]

[1] The Institute of Scientific and Industrial Research, Osaka University, Japan
[2] Institute of Multidisciplinary Research for Advanced Materials, Tohoku University, Japan
[3] School of Administration and Informatics, University of Shizuoka, Japan
[4] Department of Electronics and Informatics, Ryukoku University, Japan

Abstract. We constructed two types of neighbor networks, i.e., TOP k and k-MNN (Mutually Nearest Neighbor) networks, on observed AE (Acoustic Emission) data produced by damages in SOFC (Solid Oxide Fuel Cell). Afterwards, we analyzed growth properties of the neighbor networks for evaluation of damage progress. The results show that the power index of degree dynamically changes as damage progress phase changes. Also we found the decrease of cluster coefficient and shrinking effective diameter in k-MNN network reflect the occurrence of various combination of damages.

Keywords: complex network, temporal analysis, degree, cluster coefficient, effective diameter.

1 Introduction

Complex network can be observed all over the place in the world, e.g., WWW, citation networks, protein reaction network among others. It is reported that these complex networks often have the properties of scale free and/or small world[1] which means these networks are not a simple random or regular network. Analyzing network properties is one of the effective way to understand a structure and relationships among objects. Moreover, since most of the networks in the real world change over time, it is important to investigate not only a snapshot but also growth of the network, e.g., Leskovec et. al. studied the growth of citation network and reported densification power law[2].

Meanwhile, this paper focuses on physical damage on a fuel cell. Fuel cell is regarded as a highly efficient power generation system as well as low-pollution. However, a crucial issue in putting SOFC (Solid Oxide Fuel Cell) into practical use is the establishment of a technique for evaluating the deterioration of SOFC in the operating environment[3]. Since SOFC operate in a harsh environment (i.e., high temperature, oxidation-reduction), this causes decrement of the

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 933–940, 2009.
© Springer-Verlag Berlin Heidelberg 2009

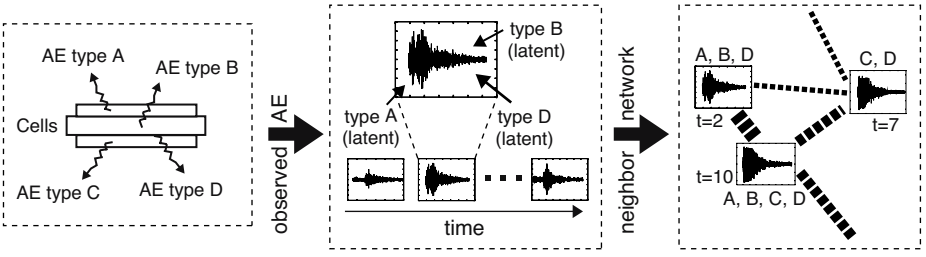


Fig. 1. Simplified image of a neighbor network on AE waves

reaction area by chemical/physical degradation, consequently the cell loses its performance. We (two of the co-authors) have previously developed a technique to measure the mechanical damage of SOFC using the Acoustic Emission (AE) method[4]. AE is an elastic wave (i.e., vibration, sound wave including ultrasonic wave) produced by expansion/contraction/crack/friction of materials[5]. The final goal of our research is to establish a technique to estimate damage type and clarify fracture mechanism (pattern).

We have conducted an initial experiment using Sequence-based SOM (Self-Organizing Map)[7], which we modified the original SOM[6], assuming a group of similar AE waves represents certain type of damage. However, several AE waves are observed overlapped in one wave. It is difficult to investigate patterns of latent (overlapped) AE types by cluster analysis since it does not represent relationship between objects. Therefore this paper employed neighbor network, TOP k and k-MNN (Mutually Neighbor Network) are employed, which reflects the shared latent AE types implicitly (Fig.1). Network structure can be patterns of shared latent AE types, and also if occurrence pattern of latent AE types change as damage progresses, it should affect growth of the network. Hence, this paper studied the network growth in terms of typical network characteristics: power index of degree, cluster coefficient, and effective diameter. We confirmed from the results that the power index of degree dynamically changes as state of materials change. Also the decrease of cluster coefficient and shrinking effective diameter in k-MNN network reflect the occurrence of various damage combinations as lowering the temperature.

2 Neighbor Network of AE Waves

2.1 Similarity Measure

In advance of network construction, similarity between extracted two AE waves is defined as follows. Let the feature vector of the t^{th} AE wave be $\mathbf{x}_t = (\hat{\mathbf{x}}_t, \bar{\mathbf{x}}_t)$, where $\hat{\mathbf{x}}$ is a time-domain feature vector and $\bar{\mathbf{x}}$ is a spectrum-domain feature vector. Here, the time-domain consist of the following 7 features[8]: duration, maximum amplitude, root mean square, crest factor, variance, skewness, and kurtosis. Also spectrum-domain features consist of K PC (Principal Component) score of the power spectrum obtained by PCA. All features are normalized to be

each average is equal to zero and the standard deviation is equal to one. Then similarity measure $S(\mathbf{x}_i, \mathbf{x}_j)$ is given by:

$$S(\mathbf{x}_i, \mathbf{x}_j) = \gamma * f_s(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j) + (1-\gamma) * f_s(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j), \tag{1}$$

where $f_s(\mathbf{a}, \mathbf{b}) = \exp\left(-\|\mathbf{a}-\mathbf{b}\|/\sigma^2\right)$ is a Gaussian function that maps Euclidean distance of two objects into $[0, 1]$. Also $\gamma \in (0, 1]$ is a balancing parameter between time- and spectrum-domain features, and σ is a scaling parameter, $\sigma = 1.0$ is used in this work.

2.2 TOP k Network

$N \times N$ adjacency matrix of TOP k network $A = (a_{i,j})$ is given by:

$$a_{i,j} = \begin{cases} 1 & SR_{all}(\mathbf{x}_i, \mathbf{x}_j) \leq k \quad (i < j) \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where the function $SR_{all}(\mathbf{x}_i, \mathbf{x}_j)$ returns the similarity ranking between $\mathbf{x}_i$ and $\mathbf{x}_j$ by eq.(1) among $\forall l, m\, \{(\mathbf{x}_l, \mathbf{x}_m); l < m\}$. That is, TOP k network has links if the ranking of similarity between the nodes (AE waves) is within top k among all the combination. Since this network is an undirected graph, only the part $i < j$ is used.

2.3 k-MNN Network

$N \times N$ adjacency matrix of k-Mutually Nearest Neighbor (MNN) network $B = (b_{i,j})$ is given by:

$$b_{i,j} = \begin{cases} 1 & SR_{ind}(\mathbf{x}_i, \mathbf{x}_j) \leq k \text{ and } SR_{ind}(\mathbf{x}_j, \mathbf{x}_i) \leq k \quad (i < j) \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where the function $SR_{ind}(\mathbf{x}_i, \mathbf{x}_j)$ returns the similarity ranking of $\mathbf{x}_j$ from $\mathbf{x}_i$ by eq.(1) among $\forall l\, \{(\mathbf{x}_i, \mathbf{x}_l); i \neq l\}$. That is, k-MNN network has links only if $\mathbf{x}_j$ is within k nearest neighbor of $\mathbf{x}_i$ and vice versa.

2.4 Configuration of Growth Network

A network at time $t_a \in \{1, \cdots, T\}$ is constructed by using $\{\mathbf{x}_t : t = 1, \cdots, t_a\}$. Though the configuration is backward removing process, which removes a node and related links at time t, once the network is configured the growth process can be replayed forward.

3 Network Statistics

Power Index of Degree. A network whose degree distribution follows power law is called a scale-free network. We calculated a power index by the least square fitting. If a power index is large, there are a large number of AE waves that have a small number of latent AE types and a small number of AE waves that have a large number of latent AE types.

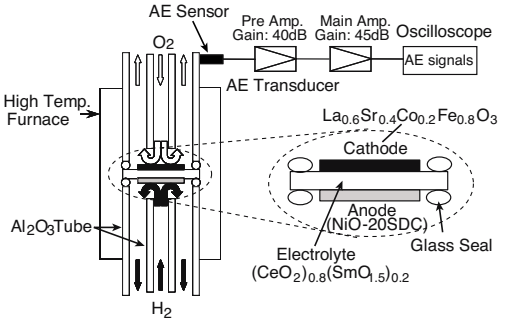


Fig. 2. The schematic layout of the SOFC's performance test apparatus

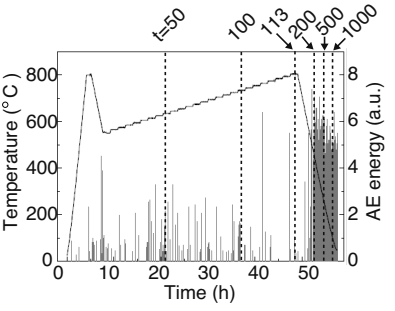


Fig. 3. Controlled temperature change and detected AE energy

Cluster Coefficient. We employed the definition of cluster coefficient based on the number of triangles within the whole network at time t [9]. The cluster coefficient indicates degree of tight coupling, in other words degree of grouping of similar AE waves.

Effective Diameter. We employed the effective diameter[2] that is the minimum distance **d** that more than ratio of β node pairs are connected at most a distance **d** ($\beta = 0.9$ is used in this work). If AE waves that have various combination of latent AE types are produced, the effective diameter will be shrunk because it makes shortcut between clusters.

4 Growth Analysis of the AE Networks

4.1 Damage Evaluation Test Settings

The schematic layout of the SOFC's performance test apparatus is shown in Fig.2. The test section was initially heated up to 800 °C in order to melt the soda glass ring (Fig.3). AE measurement was done by using a wide-band piezoelectric transducer[1]. The AE transducer was attached to the outer Al_2O_3 tube away from the heated section. The sampling rate is 10^{-6} sec, thus the maximum frequency is 500 KHz. The bar graph in Fig.3 indicates AE energy, which corresponds to the occurrence frequency of AE signals at that time.

Through 60 hours of running SOFC, 1429 AE waves were extracted by the burst extraction method[10], where the maximum burst level was set to five. From these AE waves, we prepared TOP 5000, 10000 and 15-, 30-MNN networks, where the number of links in 15- and 30-MNN networks were 5347 and 10818, respectively.

4.2 Network Overview

Overview of the TOP 5000 network and the 15-MNN network are shown in Fig.4 and Fig.5, where only the maximum connection components are depicted.

[1] PAC UT-1000, URL: http://www.pacndt.com

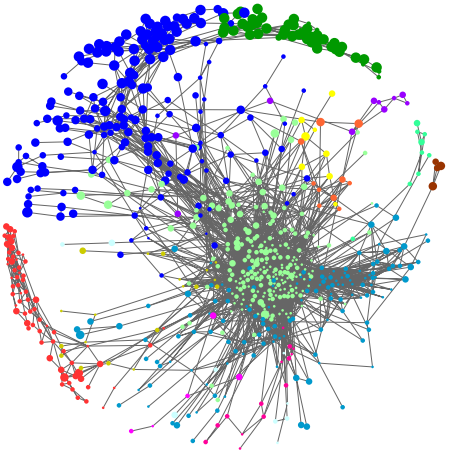


Fig. 4. Overview of TOP 5000 network

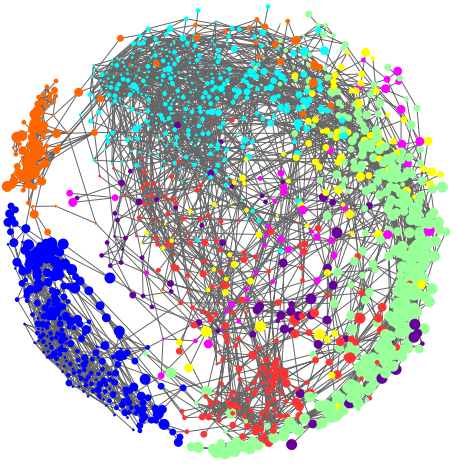


Fig. 5. Overview of 15-MNN network

The networks are visualized by using Cytoscape[2] and the layout is done by the weighted spring embedded. Note that similarity values are used as the weights, however, the values are used only for visualization. Size of a node indicates the maximum burst level of correspond AE wave that represents damage scale. Moreover, the networks are split into groups by means of Fast Modularity Community Structure Inference Algorithm[3][11]. The balancing parameter in the similarity measure by eq.(1) was set to $\gamma = 0.4$ which gives the highest modularity (in average of four networks) among the candidates $\{0.1, 0.2, \cdots, 0.9\}$.

TOP 5000 network has a core in the center and there are mainly three groups around the core, while 15-MNN network has mainly five groups and the center seems almost random network.These groups are expected to be the same damage type, while diverse-linked AE waves can be considered as complex waves that contains various latent AE types. Moreover, the nodes between the groups are expected to be AE waves that share latent AE types of both groups.

4.3 Change in Degree Distribution

Fig.6 and Fig.7 show change in power index of degree over time. Corresponding actual time and temperature are indicated in Fig.3. The networks whose configuration is the same (TOP k or k-MNN) show the same trend even though k changed. However, TOP k and k-MNN networks show greatly different trends. The power index in both networks change up and down rapidly till about $t = 400$, afterwards it gradually increases in TOP k network, while it gradually decreases in k-MNN network. In particular, k-MNN network does not follow power law in the last part and around $t = 100$. This result is reasonable because k-MNN network is limited to have k links per node even though possible nodes are increasing.

[2] http://www.cytoscape.org/

[3] http://www.cs.unm.edu/ aaron/research/fastmodularity.htm

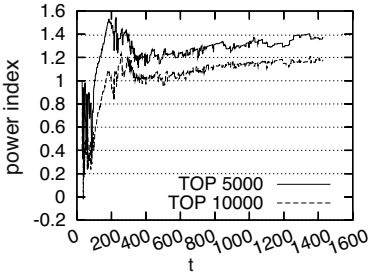


Fig. 6. Change in power index of degree on TOP k network

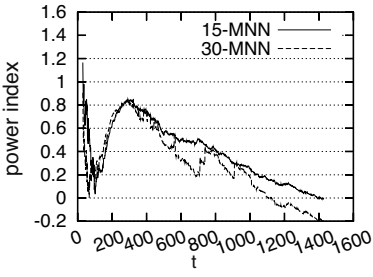


Fig. 7. Change in power index of degree on k-MNN network

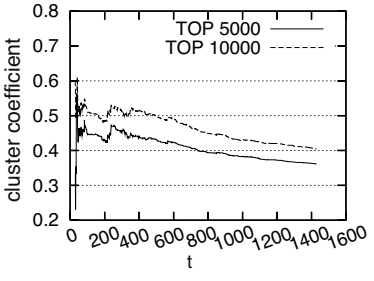


Fig. 8. Change in cluster coefficient on TOP k network

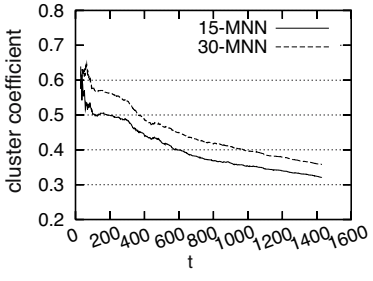


Fig. 9. Change in cluster coefficient on k-MNN network

An interesting part from perspective of damage progress is that dynamical changing points, $t = 50 - 100$, $200 - 300$, can be considered as phase transition points in the damage progress. Actually, heating period is till $t = 113$ from Fig.3 that damage types different from lowering period may occur. While about $t = 200 - 300$, we have already confirmed by the previous work that is sequential cluster analysis using Sequence-based SOM[7], very large AE waves overlapping a lot of latent AE types, which may be high degree nodes, were produced in this period. This is probably a crack progress because of the initial small cracks or unevenness of the material, which is not serious damage. The result conforms with the previous work.

4.4 Change in Cluster Coefficient

Fig.8 and Fig.9 show change in cluster coefficient over time. Both networks, TOP k and k-MNN, show similar trends, it gradually decreases. Although the number of links are almost the same between TOP k and k-MNN networks, the gradient in k-MNN network is higher than that of TOP k. This is because that k-MNN network grew from clustered areas and then diverse-linked area grew, which is the center area in Fig.5 and does not appear in TOP k network. We also confirmed this growth process by time-series visualization (though the detail is spared because of short of space). This growth process could not be found by the previous work[7].

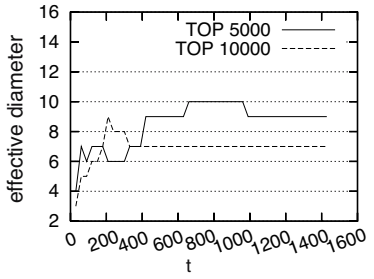


Fig. 10. Change in effective diameter on TOP k network

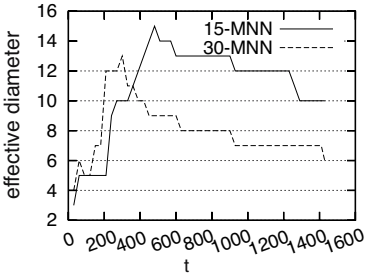


Fig. 11. Change in effective diameter on k-MNN network

4.5 Change in Effective Diameter

Lastly, Fig.10 and Fig.11 show change in effective diameter over time. The diameter increased and nearly steadied in TOP k network. While in k-MNN network, it rapidly increased till about $t = 300(k = 15)$ and $500(k = 30)$ as clustered areas grew. Since these areas are not highly connected each other, it becomes long diameter. Afterwards, as diverse-linked area grew the diameter is getting shorter since shortcuts within the network increase. This shrinking diameter may be by the same reason that we discussed about cluster coefficient.

We can speculate from the result about damage progress as follows. Since glass seal does not solidify till about 400°C(t=300, turning point of the diameter), almost single type of damage occur even containing several latent AE types. However, as solidifying the glass seal and compressing the electrodes and the electrolyte (after 400°C), chaining various types of damages may occur such as vertical cracks in electrodes, electrolyte, glass seal, delamination between electrodes and electrolyte. This may produce various combination of AE types, which leads growth of the diverse-linked area.

5 Conclusion

We analyzed the growth of neighbor networks, namely TOP k and k-Mutually Neighbor networks, on observed AE waves that is damage progress in Solid Oxide Fuel Cell. The results show that the power index of degree dynamically changes when the occurrence pattern of AE dynamically changed, which was a reasonable result. Also decrease of cluster coefficient and shrinking effective diameter in k-MNN network reflect occurrence of diverse-linked waves which is various combination of damages. This is initial experiments trying to see the effectiveness of network analysis for AE waves, other network analysis, e.g., correlation among neighbors, should be studied.

Acknowledgement

This research is supported by the project of Materials Science & Technology Research Center for Industrial Creation (MSTeC) in Japan.

References

1. Watts, D.J.: Small Worlds: The Dynamics of Networks Between Order and Randomness. Princeton Univ. Press, Princeton (2004)
2. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: Proc. of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 177–187 (2005)
3. Krishnamurthy, R., Sheldon, B.W.: Stresses due to oxygen potential gradients in non-stoichiometric oxides. Journal of Acta Materialia 52, 1807–1822 (2004)
4. Sato, K., Omura, H., Hashida, T., Yashiro, K., Kawada, T., Mizusaki, J., Yugami, H.: Tracking the onset of damage mechanism in ceria-based solid oxide fuel cells under simulated operating conditions. Journal of Testing and Evaluation 34(3), 246–250 (2006)
5. Spanner, J.C.: Acoustic Emissions: Techniques and Applications. Amer Society for Nondestructive (1974)
6. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (1995)
7. Fukui, K., Sato, K., Mizusaki, J., Saito, K., Numao, M.: Combining burst extraction method and sequence-based som for evaluation of fracture dynamics in solid oxide fuel cell. In: Proc. of 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), pp. 193–196 (2007)
8. Scheffer, C., Heyns, P.S.: Wear monitoring in turning operations using vibration and strain measurements. Mechanical Systems and Signal Processing 15(6), 1185–1202 (2001)
9. Newman, M.E.J.: The structure and function of complex network. SIAM Review 45(2), 167–256 (2003)
10. Kleinberg, J.: Bursty and hierarchical structure in streams. In: Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1–25 (2002)
11. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Physical Review E 69(2), 026113 (2004)

A Parallel Algorithm for Finding Related Pages in the Web by Using Segmented Link Structures*

Xiaoyan Shen, Junliang Chen, Xiangwu Meng, Yujie Zhang, and Chuanchang Liu

State key Laboratory of Networking and Switching Technology Beijing University of Posts and Telecommunications, Beijing, China
mynameissxy@yahoo.com.cn,
{chjl,mengxw,zhangyj,lcc3265}@bupt.edu.cn

Abstract. In this paper, a simple but powerful algorithm: block co-citation algorithm is proposed to automatically find related pages for a given web page, by using HTML segmentation technologies and parallel hyperlink structure analysis. First, all hyperlinks in a web page are segmented into several blocks according to the HTML structure and text style information. Second, for each page, the similarity between every two hyperlinks in the same block of the page is computed according to several information, then the total similarity from one page to the other is obtained after all web pages are processed. For a given page u, the pages which have the highest total similarity to u are selected as the related pages of u. At last, the block co-citation algorithm is implemented in parallel to analyze a corpus of 37482913 pages sampled from a commercial search engine and demonstrates its feasibility and efficiency.

Keywords: Related pages, Co-citation algorithm, HTML Segmentation, Parallel, Scalable.

1 Introduction

Finding related pages is not an easy job. Many studies have been conducted to evaluate the similarity of two pages efficiently and accurately, and the current algorithms for finding related pages are divided into three categories.

The first category is content-based algorithms which compute page similarities completely based on page contents [1-2]. However, these algorithms ignore the available hyperlink data and are subject to infection by spam pages. Another problem of content-based methods is that they generally require large storage and long computing time, so these methods are not easily applied to large corpus.

The second category is link-based algorithms, which use the connectivity information to evaluate the similarity between web pages [3-7]. Though link-based algorithms are in general more resistant against spamming, they ignore the differences of hyperlinks within a page and treat a web page as a single node. However, web pages can be

* This research was sponsored by National Natural Science Foundation of China (No. 60432010), National 973 project of China(No. 2007CB307100).

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 941–948, 2009.
© Springer-Verlag Berlin Heidelberg 2009

subdivided into several smaller sections and each of which may contain some useful semantic information with the same topic. We refer to such sections as blocks. The relationship between links in the same block may stronger than the relationship between links in different blocks.

The third category is anchor-based algorithms which use the similarity between texts near anchors, denoted as the anchor window, to find related pages [8-9]. Using anchor window, which often has a topic summary of the pointed target document, can avoid the expensive time and space cost in the content analysis process.

In this paper, we propose a block co-citation algorithm to find related pages. We revise the co-citation algorithm by only considering link pairs in the same block. We also introduce the parallel implementation of our algorithm in detail, which can handle large number of pages efficiently.

This paper is organized as follows: In Section 2, we introduce the details of how to find related pages and the parallel implementation of the algorithm by using MapReduce model. In Section 3, we measure the required time of our algorithm based on MapReduce model and use manual evaluation strategies to compare the experiment results of proposed algorithm with traditional co-citation algorithm. Finally, we conclude with discussion and future work in Section 4.

2 Algorithm Details

The most simple yet powerful non-vision based methods was proposed in [10]. A list of partitioning tags was used to identify the boundary between HTML sections. Though their method is clear and performing well, it can not be applied directly by us. The granularity of blocks is too small to get enough co-citation link pairs.

For the reasons above mentioned, we revise the segmenting algorithm in [10] to better suit our situation. In this paper, a block of a page is defined as a list of links in the page. The indexes of links in the block represent their relative order in original page. The size of block means the number of links contained in the block. In contrast, the initial blocks obtained from [10] with text information is called as a section here.

In order to avoid generating many trivial blocks, we should merge several neighboring sections according to some heuristic rules based on the observations of a few thousand web pages from various sources and formats. At last, all links in each section are extracted to construct a block.

For arbitrary two neighbor sections A and B, the following rules are used to decide whether two sections can be merged. Each of the rules is used, if and only if previous rules are not satisfied.

A. If the ratio of the number of anchor words to the number of total words in an section is greater than a threshold, and the number of links in the block is larger than a threshold (we set it to 2), we call the type of the section is a link section, otherwise, is a non-link section. When both type of A and B is link section, and the background color attribute of the CSS style in them is the same, they are merged. Normally,

B. A and B can not be merged, if one of the sections has boldfaced texts.

C. A and B can not be merged, if the font, font size and font color of the text in them are different. We combine these factors into a formula to quantify this difference.

Based on our observation, two sections with total different font style are almost belonged to different topics.

D. A and B can not be merged, if the number of words in both two blocks is larger than 10.

E. When the type of A and B is not the same, but they can be merged into a section with type is identical to the type of original one which has more words than the other, A and B are merged.

F. If A and B can not comply with each of the above rules, they can not be merged.

In order to validate the proposed segmenting strategies, 56 pages sampled randomly from the experimental corpus. The links in each page are segmented into blocks manually according to the topic. For each page, let BA i be the set of blocks generated using the algorithm, then the total number of blocks is expressed as:

$$NB = \sum_i \left| BA_i \right| \tag{1}$$

For ease of analysis, define BM_i as the set of blocks generated manually.

E-precision is the ratio of the number of blocks the same with the blocks generated manually, to NB.

$$E-precision = \frac{\sum_i \left| \{ b \mid b \in BA_i \cap \exists b' (b' \in BM_i \cap \forall l (\in b \leftrightarrow l \in b')) \} \right|}{NB}, \tag{2}$$

where l is a link in a block.

The E-precison of original algorithm and our algorithm are 9.6% and 29.7%, respectively. It appears that original algorithm generates more blocks.

Before describing how to generate related pages, the following terms are defined.

Definition 1. Two pages p_1 and p_2 are block co-cited once if they are both pointed by two hyperlinks in the same block b of the same page p. If more than one block in p contains both p_1 and p_2, only the first is valid avoiding p devoting too high similarity to the pair. The related pair generated from page p is denoted as $<p_1, p_2>_p$, and the similarity of $<p_1, p_2>_p$ is denoted as: $S_p(<p_1, p_2>_p)$, computed from the first block.

Definition 2. The top sub-domain name of page p is defined as the domain name directly under the leftmost top-level domain, denoted as TSD(p). For example, TSD(www.yahoo.cn) is “yahoo.cn”, since “cn” is the country code top-level domain governed by China Internet Network Information Center.

The flow of finding related pages from takes all blocks as input and includes three key steps:

(1) Filtering hyperlinks. So as to avoid meaningless related pairs generated by some insignificant links, all hyperlinks in a block are processed by the following filter strategies. Only the links remained after filter procedures generate valid related pairs. The strategies are: Discard blocks which contain more than S (block size threshold) links; Discard all links pointed to a non HTML file; Replace all dynamic links with their domain name; The URL normalization technologies in [11] are used to transform all links into its canonical form; Each of the links, which

has the same URL format as some link ahead, will be deleted; All links with no anchor are discarded; Do not consider the links in a block which have the same top sub-domain name as its parent page.

(2) Generating related pairs. In this step, the similarity between every two different links remained in a block is computed, and then related pairs are generated with their similarity in current page and other useful information. Furthermore, after filtering the links in blocks, we only generate a related pair between two links with different top sub-domain name, because we do not want to bring many related pairs within the same top sub-domain name.

The similarity is computed as follow:

$$S_p(< p_1, p_2 >_p) = P_p(< p_1, p_2 >_p) * (1 + \frac{|words(A_{p,p1}) \bigcap words(A_{p,p2})|}{|words(A_{p,p1}) \bigcup words(A_{p,p1})|}), \quad (3)$$

where $A_{b,p1}$ ($A_{b,p2}$) is the anchor texts of p_1 (p_2) in block b, $words(s)$ is the words set of the anchor texts s, expecting all stop words with highest IDF.

$$P_b(< p_1, p_2 >_b) = \begin{cases} 1 & |i-j| \le \lambda \\ e^{\frac{-(|i-j|-\lambda)}{2}} & |i-j| > \lambda \end{cases}, \quad (4)$$

where i, j are the index of p_1 and p_2 in b respectively, near neighbor threshold λ is the threshold to differentiate near neighbors from far neighbors.

(3) Merging related pairs.

For illustration clearly, we first define the following concepts.

Definition 3. For a given top sub-domain name d, arbitrary two pages p_1 and p_2, the domain similarity from p_1 to p_2 is denoted as: $DS_d(p_1 \rightarrow p_2)$ and vice versa.

$DS_d(p_1 \rightarrow p_2)$ is computed by summing up all $S_p(< p_1, p_2 >_p)$ generated from page p who belong to d. In order to avoid template links making many noises in our results, for an anchor text of p_2 belong to d, only the top r (if has) $S_p(< p_1, p_2 >_p)$ will be added to $DS_d(p_1 \rightarrow p_2)$, and so as $DS_d(p_2 \rightarrow p_1)$. In order to limit some malicious top sub-domain names provide too many similarities between two pages, we limit the maximum value of $DS_d(p_1 \rightarrow p_2)$ or $DS_d(p_2 \rightarrow p_1)$ to N, which is called domain similarity threshold. Finally,

$$DS_d(p_1 \rightarrow p_2) = \min(N, \sum_j \sum_{top\ r} S_p(< p_1, p_2 >_p)) \quad (5)$$

Definition 4. For arbitrary two pages p_1 and p_2, the total similarity from p1 to p2 is calculated as:

$$TS(p_1 \to p_2) = \frac{1}{1+\ln(in_{p2})} \sum_{d \in D(p1,p2)} DS_d(p_1 \to p_2) \tag{6}$$

where in_{p2} is the number of blocks which contain page p_2.

Now we discuss the parallel implementation of the steps mentioned in section 3, which is inspired by MapReduce [12].

Figure 1 shows the MapReduce flow of find related pages from the blocks of pages.

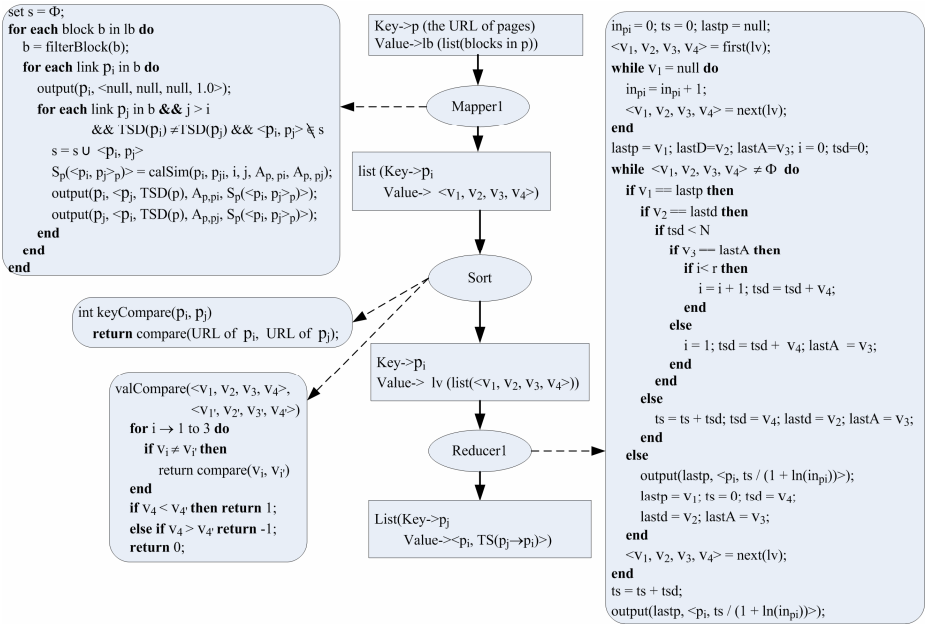


Fig. 1. MapReduce flow of finding related pages from blocks

Mapper1 operation finishes the task of filtering links and generating related pairs. The filterBlock routine filters all links in blocks as above mentioned. The calSim routine calculates the similarity of each related pairs according to (4). For a given key p_i outputted by Mapper1, the corresponding value is a quaternion, namely, $< p_j, TSD(p), A_{p,\,p_i}, S_p(< p_i, p_j >_p) >$.Besides these items, we also output an auxiliary value $< null, null, null, 0 >$ for p_i.

In Sort1 operation, the comparison method of the URL keys simply compares URLs by alphabetic order, null string is the smallest string in alphabetic order. The method to compare the output quad values of mapper1 is the sequential comparison, that is, if the first components of two values are not equal, we will output the comparison result of them, otherwise, we will compare the second components, and so on. It is worth noting that we compare the 4th component reversely, so the quad value with higher similarity is processed early in Reducer1. After the sort process is

finished, all values for the same key will be grouped together. In these values, the auxiliary quaternion value must are ahead of all normal values, the normal values following auxiliary value are also grouped by their four components in order.

In Reducer1 operation, the number of auxiliary quaternion corresponding to the key page p_i is counted to compute in_{p_i} firstly. Then for each page p_j related to p_i, the $TS(p_j \rightarrow p_i)$ is computed simply by summing all similarities with the limitation showed in (10). At last p with its total similarity to pm1 is outputted as a key-value in Reducer1.

The Mapper2 and Reducer2 operation finish generating the related pages set for each page p_j, and pick at most M pages with highest total similarity to p_j as the results. We also limit the minimum value of the total similarity to T.

3 Experiment and Evaluation

The values of the algorithm parameters mentioned in Section 2 are determined based on statistical results or from experience. The choice of Block size threshold S, Near neighbor threshold λ, Anchor repetition frequency r, domain similarity threshold N, Number of related pages M and Total similarity threshold T are 80, 8, 9, 10, 15, 4.

In order to implement the algorithm, Hadoop (http://lucene.apache.org/hadoop/) is selected as the MapReduce platform, CWT200G (http://www.cwirf.org) is used as the pages corpus.

The experiment is conducted on 8 HP servers each with 4GB memory and two dual 1.8 GHz AMD processors. All 37482913 pages in CWT200G are processed and approximately 270MB related pages database are generated. The database is divided into 64 partitions according to the hash code of the URL of pages and is stored on disk. We set the mapper and reducer numbers to 64 based on the suggestions in [12]. The time required to find related pages on 8 servers are measured by using the input set with different size from 5M to 35M pages. It is observed in Figure 2 that the total running time is linear with the number of pages. The reason is that the I/O time cost is significant in our parallel implementation.

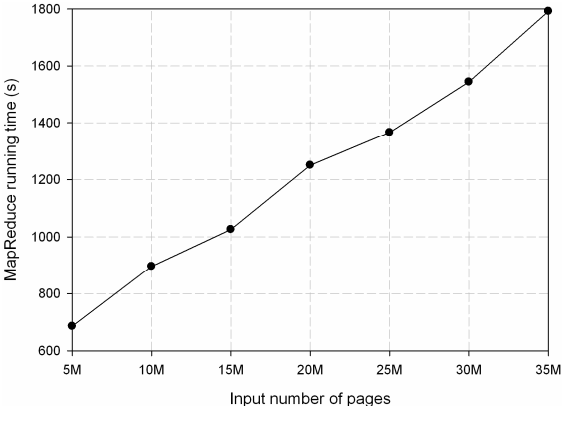


Fig. 2. Running time performance for input pages from 5M to 35M

We also implement tradition co-citation algorithm by using Hadoop. In order to evaluate precision of the proposed block co-citation algorithm and traditional co-citation algorithm, a user feedback study is performed. We carry out a blind evaluation of the results of two algorithms on 28 different pages.

For each test page, we combine and shuffle the top 15 related page results of both two algorithms before presenting them to our human evaluators. Each of the 7 evaluators, who are students in our lab, is instructed to mark all the results based on the following criteria: Score 0 stands for page could not be accessed or is totally unrelated to the test page; Score 1 stands for page belongs to a similar but not identical topic as the test page; Score 2 stands for page belongs to the same topic or type as the test page.

The performance of retrieving related pages is investigated through average R-score. First, the average score of top R results for every test page is calculated, and then the sum of the average score is divided by the number of test pages, namely:

$$\overline{S_R} = (\sum_{j=1}^{N}(\sum_{i=1}^{R}\overline{s}_{ji} / R)) / N, \tag{7}$$

where N is the number of test pages, $\overline{s}_{ji}$ is the average score given by students for the i-th relate page result of the j-th test page.

Figure 3 illustrates the average R-score comparison of the two algorithms. It is shown that the average R-score of block co-citation algorithm is better than that of traditional co-citation algorithm over all the R values.

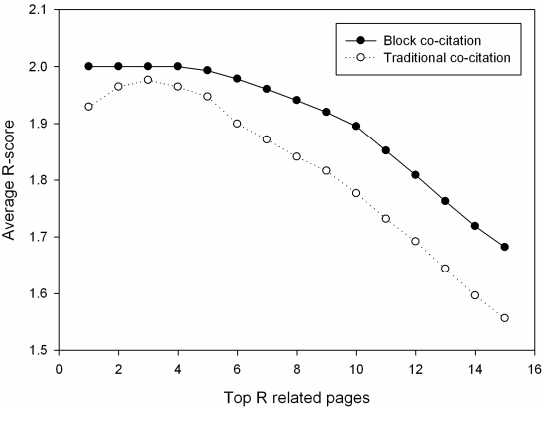


Fig. 3. Comparison of average R-score

4 Conclusions

In this paper, the block co-citation algorithm based on HTML segmentation technologies is introduced to find related pages for a given page, and demonstrate one implementation of this method using MapReduce framework to achieve scalability and

parallelization. By limiting the vote times of links with the same anchor text in the same top sub-domain name and the value of domain similarity, the influence of template links is reduced to a reasonable level. It also prevents the results from being distorted by malicious hyperlinks in malicious domains. Experiments conducted on a large corpus suggest that block co-citation algorithm outperforms traditional co-citation algorithm in the scenarios we tested. Given all that, block co-citation algorithm may be a practical choice for commercial search engines to provide related pages service.

References

1. Loia, V., Senatore, S., Sessa, M.I.: Discovering related web pages through fuzzy-context reasoning. In: The 2002 IEEE International Conference on Plasma Science, pp. 100–105 (2002)
2. Fan, W.-B., et al.: Recognition of the topic-oriented Web page relations based on ontology. Journal of South China University of Technology (Natural Science) 32(suppl.), 31–47 (2004)
3. Dean, J., Henzinger, M.R.: Finding related pages in the World Wide Web. Computer Networks 11(11), 1467–1479 (1999)
4. Tsuyoshi, M.: Finding Related Web Pages Based on Connectivity Information from a Search Engine. In: Proceedings of the 10th International World Wide Web Conference, pp. 18–19 (2001)
5. Hou, J., Zhang, Y.: Effectively finding relevant web pages from linkage information. IEEE Transactions on Knowledge and Data Engineering 11(4), 940–950 (2003)
6. Ollivier, Y., Senellart, P.: Finding Related Pages Using Green Measures: An Illustration with Wikipedia. In: The 22nd National Conference on Artificial Intelligence (AAAI 2007). pp. 1427–1433 (2007)
7. Fogaras, D., Racz, B.: Practical Algorithms and Lower Bounds for Similarity Search in Massive Graphs. IEEE Transactions on Knowledge and Data Engineering 19(5), 585–598 (2007)
8. Chakrabarti, S., et al.: Automatic Resource Compilation by Analyzing Hyperlink Structure and Associated Text. In: The 7th International Conference on World Wide Web, pp. 65–74 (1998)
9. Chakrabarti, S., Dom, B., Indyk, P.: Enhanced Hypertext Categorization Using Hyperlinks. In: 1998 ACM SIGMOD international conference on Management of data. pp. 307–318 (1998)
10. Debnath, S., et al.: Automatic identification of informative sections of Web pages. IEEE Transactions on Knowledge and Data Engineering 17(9), 1233–1246 (2005)
11. Lee, S.H., Kim, S.J., Hong, S.H.: On URL normalization. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganá, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3481, pp. 1076–1085. Springer, Heidelberg (2005)
12. Dean, J., Ghemawat, J.: MapReduce Simplified Data Processing on Large Clusters. In: The Proceedings of the 6th Symp. on Operating Systems Design and Implementation, pp. 137–149 (2004)

Boosting Biomedical Information Retrieval Performance through Citation Graph: An Empirical Study

Xiaoshi Yin[1,3], Xiangji Huang[1], Qinmin Hu[2], and Zhoujun Li[3]

[1]School of Information Technology, York University,
Toronto, Ontario, M3J 1P3, Canada
{xyin,jhuang}@yorku.ca
[2]Computer Science Department, York University,
Toronto, Ontario, M3J 1P3, Canada
vhu@cse.yorku.ca
[3]School of Computer Science and Engineering, Beihang University,
Beijing, 100083, China
lizj@buaa.edu.cn

Abstract. This paper presents an empirical study of the combination of content-based information retrieval results with linkage-based document importance scores to improve retrieval performance on TREC biomedical literature datasets. In our study, content-based information comes from the state-of-the-art probability model based Okapi information retrieval system. On the other hand, linkage-based information comes from a citation graph generated from *REFERENCES* sections of a biomedical literature dataset. Three well-known linkage-based ranking algorithms (PageRank, HITS and InDegree) are applied on the citation graph to calculate document importance scores. We use TREC 2007 Genomics dataset for evaluation, which contains 162,259 biomedical literatures. Our approach achieves the best document-based MAP among all results that have been reported so far. Our major findings can be summarized as follows. First, without hyperlinks, linkage information extracted from *REFERENCES* sections can be used to improve the effectiveness of biomedical information retrieval. Second, performance of the integrated system is sensitive to linkage-based ranking algorithms, and a simpler algorithm, InDegree, is more suitable for biomedical literature retrieval.

Keywords: biomedical information retrieval, ranking, link analysis.

1 Introduction

Current biomedical research is characterized by immense volumes of data, accompanied by a tremendous increase in the number of biomedical related publications. In order to provide biologists a more effective and more accurate way to access biomedical literature datasets, this paper explores an integrated approach to bring linkage information into biomedical retrieval. We utilize information in

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 949–956, 2009.
© Springer-Verlag Berlin Heidelberg 2009

REFERENCES sections to generate a citation graph, which can be seen as a network defined by content relevant or content similarity. Three linkage-based ranking algorithms, InDegree, PageRank and HITS, are applied to the citation graph to calculate linkage-based ranking scores for documents in the dataset. On the other hand, content-based topic-relevant ranking scores come from the probability model based Okapi IR system, which ranks documents according to their topic-relevant probabilities. Finally, two different combination methods are used to integrate linkage-based information with content-based information. In our experiments, significant performance improvements can be obtained by using the integrated approach. It is worth noting that the approach we introduced achieves the best document-based MAP among all results that have been reported so far.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 introduces the integrated approach which combines content-based IR results from a probability model based IR system with linkage-based ranking scores from a citation graph to improve biomedical literature retrieval performance. Our experimental results followed by analysis and discussion are presented in Section 4. Conclusions of this paper are presented in Section 5.

2 Related Work

The idea of using link analysis to improve the effectiveness of a information retrieval system was proposed in 1998 [1][2]. A recent research by Borodin et al. [3] was based on 34 queries and 200 webpages retrieved by the Google search engine for each query. Linkage-based ranking scores calculated by link analysis ranking algorithms were combined with Google's retrieval results. Surprisingly, some relative simpler link analysis ranking algorithms outperformed more sophisticated algorithms [3]. Najork et al. [4] evaluated the effectiveness of link analysis ranking algorithms based on a webpage dataset containing 463 million webpages and 28,043 queries. Their experimental results show that utilize linkage information can improve ranking results from a content-based IR system, and HITS algorithm is about as effective as webpage InDegree algorithm. Another research in biomedical literature retrieval which is more similar to our work was done by Lin [5]. This research focused on searching on biomedical literature abstracts, in the context of the PubMed search engine. In this research, PageRank and HITS were used to calculate linkage-based ranking scores on the abstract network provided by the PubMed search engine. The traditional linear combination method was used to combine the linkage-based ranking scores with retrieval results from Terrier information retrieval platform on TREC 2005 Genomics dataset.

Instead of using hyperlinks or content similarity links provided by a third part search engine, linkage information we used are extracted from a citation graph, which is generated from *REFERENCES* sections in TREC 2007 Genomics dataset. We find that a simpler link analysis algorithm is more suitable for biomedical literature retrieval than well-known PageRank and HITS.

3 An Integrated Approach

The integrated information retrieval system we introduced consists of two components, which are content-based information retrieval component and linkage-based ranking score generator.

The content-based IR component is the state-of-the-art probability model based Okapi IR system [6][7], which retrieves topic-relevant documents from the dataset. Its output is a ranked list of 1000 most relevant documents for each topic according to their relevance scores. The linkage-based ranking score generator aims to calculate importance scores for retrieved documents by applying a linkage ranking algorithm to the citation graph. Normalized scores from two components are combined to calculate new ranking scores for retrieved documents.

3.1 Content-Based Approach

We use Okapi Basic Search System as the content-based IR component. Okapi is an information retrieval system based on the probability model of Robertson and Sparck Jones [8]. Retrieved documents from Okapi are ranked in the order of their probabilities of relevance to the query. Search term is assigned weight based on its within-document term frequency and query term frequency. The weighting function used is BM25 [6].

$$w = \frac{(k_1+1)*tf}{K+tf} * \log \frac{(r+0.5)/(R-r+0.5)}{(n-r+0.5)/(N-n-R+r+0.5)} * \frac{(k_3+1)*qft}{k_3+qft} \oplus k_2 * nq * \frac{(avdl-dl)}{(avdl+dl)} \quad (1)$$

where N is the number of indexed documents in the collection, n is the number of documents containing a special term, R is the number of documents known to be relevant to a special topic, r is the number of relevant documents containing the term, tf is within-document term frequency, qtf is within-query term frequency, dl is the length of the document, $avdl$ is the average document length, nq is the number of query terms, the k_is are tuning constants (which depend on the database and possibly on the nature of the queries and are empirically determined), K equals to $k_1 * ((1-b) + b * dl/avdl)$ and $\oplus$ indicates that its following component is added only once per document.

In our experiments, the values of k_2 and k_3 in BM25 function are set to be 0 and 8 respectively. k_1 and b are set to two groups of values. More details will be shown in Section 4.

3.2 Linkage-Based Approach

Citation Graph Construction. For 36 official topics in TREC 2007 Genomics track, Okapi provides 36 ranked documents lists. For each topic, after removing duplicate documents in corresponding list, we get a document set called *Root Set*. Then the Root Set is augmented by documents that cite documents in Root Set and cited by documents in Root Set, to obtain a larger *Base Set*.

Given a Base Set, a citation graph can be constructed. A node is created for every document in Base Set, and a directed edge is placed between two nodes if there is a citation relationship between two documents.

Link Analysis ranking algorithms. We utilize three different algorithms to calculate linkage ranking scores on the citation graph, they are InDegree, PageRank and HITS.

InDegree. is a simple node importance ranking algorithm, which rank documents according to their *popularities* [3]. The popularity of a document is defined as the number of documents cited it. InDegree is a simple algorithm which can be viewed as a predecessor of all link analysis ranking algorithms.

PageRank. extended InDegree algorithm, which considers links between nodes with different weights, proposed by Brin and Page in 1998. The PageRank algorithm performs a random walk on the graph that simulates the behavior of a "random surfer". The surfer starts from some nodes chosen according to a distribution D. Page et al. define the PageRank of a webpage A as:

$$PR(A) = (1-d)D(i) + d * \sum_i PR(T_i)/C(T_i) \tag{2}$$

where $PR(T_i)$ denotes the PageRank of document T_i which cites document A in our experiments. $C(T_i)$ means the number of references in document T_i. And $d \in [0,1]$ is a damping factor which is usually set to 0.85 [9].

HITS. introduced by Kleinberg [10], is a more refined algorithm for calculating the importance of webpages. The importance of a webpage is decided by two attributes, namely *hub* attribute and *authority* attribute. The hub attribute captures the quality of the page as a pointer to useful resources, while the authority attribute captures the quality of the page as a resource itself. HITS computes two vectors of scores, which defined by Kleinberg (1999) as:

$$\vec{a'} = A^T * \vec{h} \tag{3}$$

$$\vec{h'} = A * \vec{a} \tag{4}$$

where A in our experiments is the adjacency matrix of the citation graph, $A_{ij} = 1$ if document i refers to document j, otherwise, $A_{ij} = 0$; $\vec{a}$ is authority vector where ith element denotes the authority score of document i; while vector $\vec{h}$ contains the hub scores of documents.

3.3 Integration

Many previous research studies used the traditional linear combination method to combine content-based information with linkage-based information [3][5]. The traditional linear combination is defined as:

$$Score_{combine} = \lambda * NScore_{content} + (1-\lambda) * NScore_{linkage} \tag{5}$$

where $Score_{combine}$ denotes combination score; $NScore_{content}$ denotes normalized content-based score; $NScore_{linkage}$ denotes normalized linkage-based score; and $\lambda \in [0,1]$ is the combination parameter which denotes the weight of content-based score in combination. In our experiments, score normalization method we used is 0-1 normalization.

Some research has demonstrated that link analysis ranking algorithms may promote documents which are irrelevant to topic [3]. So we introduce a modified linear combination method, which takes a document's topic-relevant rank into account. Formally, we define the modified linear combination method as:

$$Score'_{combine} = \lambda * NScore_{content} + (1 - \lambda) * (NScore_{linkage}/rank_{content}) \quad (6)$$

where $rank_{content}$ is the rank of a document in content-based IR result. From equation (6) we can see that a document's linkage-based score will be suppressed if it is believed to be irrelevant to the query, since the larger the document's content-based rank is the more irrelevant it is believed.

4 Experimental Results

4.1 Dataset and Evaluation Measures

The dataset used in our experiments is TREC 2007 Genomics dataset, which is a full-text biomedical corpus consisting of 162,259 documents from 49 genomics-related journals. Documents in this dataset are in HTML format, and can be identified by their PMIDs [11]. 36 topics from TREC 2007 Genomics track are used as queries.

In our experiments, we use document-based Mean Average Precision (MAP) as the evaluation measure, since the linkage-based ranking scores are calculated based on a document-level citation graph. Moreover, document-based MAP is the most widely recognized measure in information retrieval. The definition of document-based MAP can be found in [11].

4.2 Experimental Results and Analyses

In this section, "Okapi" means the content-based Okapi IR system, "InDegree", "PageRank" and "HITS" denote three integrated IR systems using InDegree, PageRank and HITS as their linkage-based ranking algorithm respectively.

Influence of different ranking algorithms. Figures 1 shows evaluation results on document-based MAP with different λ values. We can see that PageRank

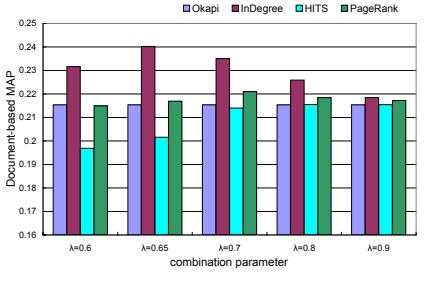
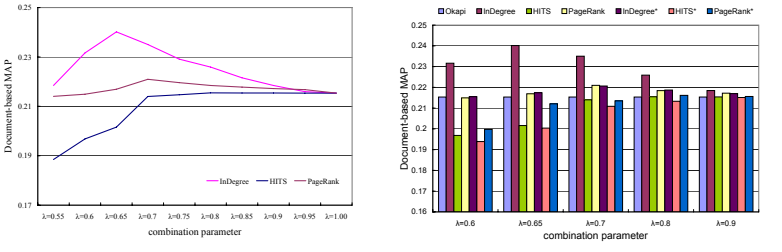


Fig. 1. Performances on different algorithms

outperforms HITS over all λ values. This makes sense, since a node (document) in a citation graph can be a good authority when many documents cite it, but it usually is not a good hub at the same time, because of the time span limitation of Genomics dataset. However, the philosophy of PageRank algorithm is different, which only concerns about authority. Thus, PageRank is more suitable for our experiments. InDegree algorithm, because of its simplicity, we expected to set it as a bar to other algorithms. Surprisingly, InDegree outperforms all other algorithms over every λ value. This may lies in that the structure of biomedical literature citation graph is different from the WWW hyperlink graph. The connectivity of a citation graph is not depends on some key articles [12], which results in a simpler algorithm, InDegree, suits for biomedical literature retrieval more. The time complexity of InDegree algorithm is much less than PageRank and HITS. Its time complexity increases linearly with the growing of the graph.



(a) Influence of combination parameter (b) Influence of combination methods

Fig. 2. Influence of combination parameter and methods

Influence of different combination methods. Two different linear combination methods are used in our experiments. Figure 2(a) shows document-based MAP values of the integrated IR system using the modified linear combination method with different λ values. The comparison of the effect of combination methods is shown in Figure 2(b). InDegree*, PageRank* and HITS* mean the integrated systems using the traditional linear combination method and applying corresponding link analysis algorithm.

Tuning involves an exploration of λ from 0 in steps of 0.02. From Figure 2(a) we can see that InDegree and PageRank can bring the maximum positive influences to the content-based IR system when the weight assigned to linkage component is 0.35 and 0.3 respectively. HITS is more likely to promote topic-irrelevant documents, only when the weight assigned to linkage-based ranking scores from HITS is less than 0.2, it can slightly improve the content-based retrieval result. Figure 2(b) shows that the modified linear combination method can achieve a better document-based MAP than the traditional linear combination method by suppressing linkage-based ranking scores of the documents which are believed to be less relevant.

Performance on different baseline runs. In order to find out wheather our integrated approach can also make performance improvements on other

content-based IR runs, we combine linkage-based ranking scores with five different content-based IR runs. We built different indices and set different BM25 parameters to get different content-based IR results. In Table 1, $Okapi_1$ and $Okapi_2$ are the content-based Okapi runs which set BM25 parameters $k_1 = 0.5$ and $b = 1.3$, using word-based index and sentence-based index respectively. $Okapi_3$, $Okapi_4$ and $Okapi_5$ are content-based Okapi runs which all set BM25 parameters $k_1 = 2$ and $b = 0.4$, using paragraph-based index, sentence-based index and word-based index respectively. These parameter settings have been tested on many datasets in different domains, which results in good performances [6][7].

Note that $Okapi_4$ achieves the best document-based MAP among results that have been reported so far. Although $Okapi_3$ is not as good as the best run reported in TREC 2007 Genomics Track in terms of document-based MAP, it outperforms the best run reported in TREC 2007 Genomics Track on document-based MAP [11] after integrating linkage information into it.

Table 1. Performances on different IR runs

Run	DocumentMAP	improvement	PassageMAP	improvement
$Okapi_1$	0.2153		0.0946	
$Integrate_1$	0.2401	**+11.5%**	0.1001	+5.8%
$Okapi_2$	0.2329		0.0899	
$Integrate_2$	0.2456	+5.4%	0.1033	**+14.8%**
$Okapi_3$	0.3277		0.0865	
$Integrate_3$	0.3489	+6.4%	0.0910	+5.2%
$Okapi_4$	0.3502		0.0899	
$Integrate_4$	**0.3691**	+5.3%	0.0945	+5.1%
$Okapi_5$	0.2084		0.0758	
$Integrate_5$	0.2298	+10.2%	0.0834	+10.0%

Table 1 shows document-based MAP and Passage-based MAP [11] when using InDegree algorithm in the integrated approach. We can see that among three link analysis ranking algorithms, InDegree algorithm still performs best. our integrated approach can make improvements on different IR runs, no matter whether the baseline run is strong or weak. For the IR baseline run that achieves the best document-based MAP in TREC 2007 [11], the integrated approach still makes 5.3% improvement on document-based MAP and 5.1% improvement on Passage-based MAP. Overall, the integrated approach achieves up to 11.5% improvement on Doucment-based MAP and 14.8% improvement on Passage-based MAP.

5 Conclusions

In this paper, we propose an integrated approach to improve the effectiveness of biomedical literature retrieval, which combines content-based topic-relevant ranking scores with linkage-based ranking scores. The linkage-based ranking scores are calculated on a citation graph generated from *REFERENCES* section of documents in TREC 2007 Genomics dataset. On the other hand, content-based topic-relevant ranking scores are provided by the probability model based

Okapi IR system. Based on the experimental results, we find that integrating linkage-based ranking scores calculated from the citation graph into content-based ranking scores can improve performances of biomedical literature retrieval. Moreover, the integrated approach makes improvements on all randomly chosen IR baselines in our experiments. The performance of an integrated IR system can be affected by link analysis ranking algorithms and combination methods. We also find that a simpler linkage-based ranking algorithm, InDegree, is more suitable for biomedical literature retrieval than some sophisticated algorithms. The integrated approach can be extended to other information retrieval domains easily, as long as citation relationships among documents can be obtained.

Acknowledgments

This research is supported in part by the research grant from Natural Science & Engineering Research Council (NSERC) of Canada, the Early Research Award/Premier's Research Excellence Award, and partly supported by China Scholarship Council.

References

1. Marchiori, M.: The quest for correct information on the web: Hyper search engines. In: Proc. of the 6th International World Wide Web Conference, pp. 1225–1235 (1997)
2. Carriere, J., Kazman, R.: Webquery: Searching and visualizing the web through connectivity. In: Proc. of the 6th International World Wide Web Conference, pp. 1257–1267 (1997)
3. Borodin, A., Roberts, G.O., Rosenthal, J.S., Tsaparas, P.: Link analysis ranking algorithms, theory, and experiments. ACM Tran. on Internet Technologies 5, 231–297 (2005)
4. Najork, M., Zaragoza, H., Taylor, M.: Hits on the web: How does it compare? In: Proc. of the 30th ACM SIGIR, pp. 471–478 (2007)
5. Lin, J.: PageRank without Hyperlinks: Reranking with Related Document Networks. Technical Report LAMP-TR-146/HCIL-2008-01 (January 2008)
6. Beaulieu, M., Gatford, M., Huang, X., Robertson, S.E., Walker, S., Williams, P.: Okapi at TREC-5. In: Proc. of of TREC-5, pp. 143–166 (1997)
7. Huang, X., Peng, F., Schuurmans, D., Cercone, N., Robertson, S.: Applying machine learning to text segmentation for information retrieval. Information Retrieval Journal 6(4), 333–362 (2003)
8. Robertson, S.E., Sparck, J.K.: Relevance weighting of search terms. Journal of the American Society for Information Science 27, 129–146 (1976)
9. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems 30, 107–117 (1998)
10. Kleinberg, J.: Authoritative sources in a hyperlinked environment. Journal of ACM (JASM) 46, 604–632 (1999)
11. Hersh, W., Cohen, A., Ruslen, L., Roberts, P.: Trec 2007 genomics track overview. In: Proc. of TREC (2007)
12. An, Y., Janssen, J., Milios, E.: Characterizing and mining the citation graph of the computer science literature. Knowledge and Information Systems 6(6), 664–678 (2004)

Similarity-Based Feature Selection for Learning from Examples with Continuous Values

Yun Li, Su-Jun Hu, Wen-Jie Yang, Guo-Zi Sun,
Fang-Wu Yao, and Geng Yang

College of Computer
Nanjing University of Posts and Telecommunications
66 Xinmofan Rd, 210003, Nanjing, P.R. China
{liyun,husj,wjyang,sun,fwyao,yangg}@njupt.edu.cn

Abstract. In many real world problems, such as machine learning and data mining, feature selection is often used to choose a small subset of features which is sufficient to predict the target labels well. In this paper, we will propose a feature selection algorithm based on similarity and extension matrix. Extension matrix is an important theory in learning from examples and it is originally designed to deal with discrete feature values. However, in the paper it is extended to cope with continuous values and designed as search strategy. The evaluation criterion for feature selection is based on the similarity between classes, which is obtained from the similarity between examples in different classes using min-max learning rule. The algorithm is proved in theory and shown its higher performance than two other classic general algorithms over several real-world benchmark data sets and facial image sets with different poses and expressions for gender classification.

Keywords: Feature Selection, Similarity, Extension Matrix.

1 Introduction

Feature selection is one of key problems in machine learning and data mining, which not only reduces training and inference time but also obtains better accuracy of the predictor. In general, feature selection including two key points: the proper evaluation criterion and the efficient search algorithm. According to the criterion, algorithms can be categorized into two classes: the filter model and the wrapper model [1,2]. In the wrapper model the selection method tries to directly optimize the performance of a specific predictor (algorithm). The main drawback of this method is its computational deficiency. In the filter model the selection is done as a preprocessing, without trying to optimize the performance of any specific predictor directly. This is usually achieved through an (ad-hoc) evaluation function using a search method in order to select a set that maximizes this function. Different methods apply a variety of search heuristics. For the search strategies, see [1,3] for a list of these techniques. However, in this paper, we focus on learning from examples, and present the feature selection evaluation criterion

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 957–964, 2009.
© Springer-Verlag Berlin Heidelberg 2009

and search strategy based on the model and theory of learning from examples. Finding the optimal set of features is intractable, and many problems related to feature selection have been shown to NP-hard [4]. As a result, we are forced to find heuristic methods that represent a compromise between solutions quality and time. The proposed method belongs to filter model.

2 Feature Selection Algorithm Analysis

2.1 Similarity Measure

As we have known, if the similarity between classes with respect to a selected feature subset is low, then the distance between classes is large and the classification performance of the selected feature subset is high, because we try to find the features that can separate the two classes as far as possible. Therefore, we first pay attention to the similarity measures between classes. Suppose the feature set is $F = [F_1, F_2, \cdots, F_n]$, then every example e is represented by $e = (v_1, v_2, \cdots, v_n)$, where v_i is the value of feature $F_i (i = 1, 2, \cdots, n)$ and it is continuous.

Definition 1. *Similarity between examples.* According to the model of learning from examples, suppose the data set has two classes, we name them as Positive class P and Negative class N. Then the similarity between a positive example $e^+ = (v_1^+, v_2^+, \cdots, v_n^+) \in P$ and a negative example $e^- = (v_1^-, v_2^-, \cdots, v_n^-) \in N$ with respect to F is defined as:

$$Sim(e^+, e^-|F) = \min_{i=1}^{n}(sim(v_i^+, v_i^-)) \tag{1}$$

where $sim(v_i^+, v_i^-)$ denotes the similarity between a positive example and a negative example with respect to feature F_i, and one solution is defined as follows:

$$sim(v_i^+, v_i^-) = 1 - |(v_i^+ - v_i^-)/(max_i - min_i)| \tag{2}$$

max_i and min_i is the maximum and minimum values of i'th feature respectively.

Noting that there exists many solutions to calculate the $sim(v_i^+, v_i^-)$, we can not guarantee that our selected similarity measure have the best performance for feature selection problem on specific data set. However, the following theoretic analysis will show our proposed feature selection method is not sensitive to the choice of similarity measure for $sim(v_i^+, v_i^-)$.

Definition 2. *Similarity between a positive example and N. with respect to F* is defined as follows, which is the maximum value of similarity between a positive example and all negative examples.

$$Sim(e^+, N|F) = \max_{e^- \in N}(Sim(e^+, e^-|F)) \tag{3}$$

Definition 3. Similarly, the *similarity between classes.* can be defined as the maximum value of similarity between all positive examples and N.

$$Sim(P, N|F) = \max_{e^+ \in P}(Sim(e^+, N|F)) \tag{4}$$

For given examples e^+, e^-, a feature set F, and a similarity measure sim, one can consider the similarity degree between e^+ and e^- with respect to F (e^+, e^- with respect to F are denoted by $e^+|F$ and $e^-|F$) is equal to 1 if and only if $e^+|F = e^-|F$, i.e., the similarity between e^+ and e^- on every feature F_i in F is equal to 1 and denoted by $\bigcap_{F_i \in F}(sim(e^+, e^-|F_i)) = 1$, and the similarity is equal to 0 if and only if $e^+|F \neq e^-|F$, i.e., $\bigcap_{F_i \in F}(sim(e^+, e^-|F_i)) = 0$. This consequent implies the overlapping degree between positive class P and negative class N(denoted by $OV(P, N|F)$) is equal to 0 if and only if the intersection of P and N with respect to F is empty, i.e., $(P \bigcap N|F) = \Phi$ and $OV(P, N|F) > 0$ if and only if $(P \bigcap N|F) \neq \Phi$ where Φ denotes empty set, and $OV(P, N|F)$ is defined as $\bigcup_{e^+ \in P} \bigcup_{e^- \in N}(\bigcap_{F_i \in F}(sim(e^+, e^-|F_i)))$ [5]. The $OV(P, N|F)$ can be considered as the similarity between P and N, and it shows whether the intersection of two classes is empty. It is clear that $Sim(e^+, e^-|F)$ taking value [0,1] with a larger value denoting a higher degree of similarity, and then min-max learning rule, i.e., $\bigcup = max$ and $\bigcap = min$, is adopted [6]. So $OV(P, N|F)$ is changed to Definition 3 above and regarded as maximal degree of similarity between two sets.

2.2 Evaluation Criterion

The optimal feature selection generally attempts to select the minimally sized subset of features without sacrificing or even get higher classification accuracy than the full feature set [7]. And just as pointed out in [3], generally, the objective function for feature selection consists of two terms that compete with each other: (1)goodness-of-fit (to be maximized), and (2) the number of features (to be minimized). Goodness-of-fit is generally denoted by classification performance and here it is inversely represented by the similarity between classes with respect to selected subset, i.e., the smaller similarity, the higher goodness-of-fit. Suppose the inter-class similarity between P and N with respect to feature set F is $R = Sim(P, N|F)$. The optimal feature subset Fs should be satisfied:

- $Fs \subset F$ and $Sim(P, N|Fs) = min_{F' \subset F} Sim(P, N|F') \leq \beta$, $\beta \geq R$, F' denotes a feature subset.
- The number of features in Fs is minimum.

2.3 Search Strategy

Extension matrix is an important theory for learning from examples and first developed by [8]. In the past, it was always used to deal with discrete feature values, such as in [5,9]. Here, we will extend it to handle continuous values and utilize it as the search strategy for feature selection.

Set positive class $P = \{e_1^+, e_2^+, \cdots, e_l^+\}$, where $e_p^+ = (v_{p1}^+, v_{p2}^+, \cdots, v_{pn}^+)$, $p = 1, 2, \cdots, l$, l is the number of positive examples and v_{pi}^+ denotes the value of feature F_i ($i = 1, 2, \cdots, n$) for the p'th positive example. Set negative class $N = \{e_1^-, e_2^-, \cdots, e_g^-\}$, where negative examples $e_q^- = (v_{q1}^-, v_{q2}^-, \cdots, v_{qn}^-)$, $q = 1, 2, \cdots, g$, g is the number of negative examples and v_{qi}^- denotes the value of

feature F_i $(i = 1, 2, \cdots, n)$ for the q'th negative example. The extension matrix of P against N is defined as Eq.(5)

$$\begin{aligned} EM(e_p^+, e_q^-) &= [em_1, em_2, \cdots, em_n], em_i = sim(v_{pi}^+, v_{qi}^-), i = 1, 2, \cdots, n \\ EM(e_p^+, N) &= [R_1, R_2, \cdots, R_g]^T, R_q = EM(e_p^+, e_q^-), q = 1, 2, \cdots, g \\ EM(P, N) &= [M_1, M_2, \cdots, M_l]^T, M_p = EM(e_p^+, N), p = 1, 2, \cdots, l \end{aligned} \quad (5)$$

Noting that each feature is corresponding to a column in extension matrix $EM(P, N)$, and they have one-to-one relationship. In other words, feature F_i is corresponding to the i'th column in the extension matrix $EM(P, N)$

2.4 Algorithm Design

Definition 4. *β consistency*, given a threshold β, if $Sim(P, N|F^{'}) \leq \beta$, then P and N is β consistent with respect to feature subset $F^{'}$, and $F^{'}$ is also called as a consistent feature subset with respect to P and N.

Definition 5. *β element*, threshold β, if the element value in extension matrix is not more than β, then the element is called a β element.

Definition 6. A *path* of an extension matrix refers to a connection of its β elements, which is obtained by selecting one β element from each row of the extension matrix $EM(P, N)$.

Lemma 1. Let $F^{'}$ be a feature subset, P and N are β consistent with respect to $F^{'}$ if and only if there exists at least one β element in the columns, which are corresponding to the features in $F^{'}$, of each row of extension matrix $EM(P, N)$.

Proof. According to Definition 4, if P and N are β consistent with respect to $F^{'}$, then $Sim(P, N|F^{'}) \leq \beta$. According to Definition 2 and 3, the similarity between any positive example and any negative example is not more than β, i.e., $Sim(e^+, e^-|F^{'}) \leq \beta$. From Definition 1, there exists at least one feature $F_i \in F^{'}$ such that $sim(v_i^+, v_i^-) \leq \beta$, namely the element em_i in the extension matrix $EM(e^+, e^-)$ is not more than β. Based on Definition 2 and 3, we know the searching involves all positive and negative examples, then all rows of extension matrix $EM(P, N)$ are traversed according to the definition of extension matrix. So there exists at least one β element in the columns, which correspond to the features in $F^{'}$, of each row of $EM(P, N)$.

Conversely, if there exists a β element in each row of extension matrix $EM(P, N)$, it implies that there exists at least a feature $F_i \in F$ $(i \in [1, n])$ such that $sim(v_i^+, v_i^-) \leq \beta$ for each row. According to Definition 1, the similarity between a positive example and a negative example with respect to F_i is not more than β. All these features F_i gotten from β element in each row are combined to obtain the feature subset $F^{'}$, which satisfies $Sim(P, N|F^{'}) \leq \beta$, then P and N is β consistent with respect to $F^{'}$. This completes the proof.

Theorem 1. Let β be a given threshold and $EM(P, N)$ be the extension matrix of P against N. Then finding an optimal feature subset is equivalent of searching

for a *path* in the $EM(P, N)$, which involves the minimum number of columns (features).

Proof. According to evaluation criterion introduced in section 2.2 and Definition 4, one knows that Fs is an optimal feature subset if and only if: P and N is β consistent with respect to Fs, and The cardinality of Fs reaches a minimum.

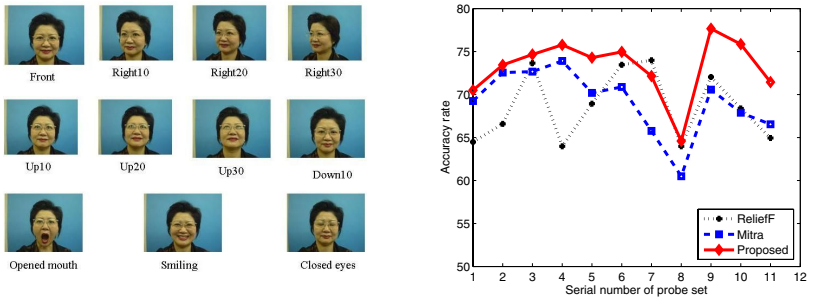
According to Lemma 1, if there exists at least one β element in the columns, which are corresponding to feature in Fs, of each row of $EM(P, N)$, then P and N is β consistent with respect to Fs. Therefore this *path* can be obtained by selecting one β element from each row of $EM(P, N)$. Each column, which is involved in the process of selecting β element, corresponds to a feature. Hence, the number of involved columns is the number of selected features. Furthermore, if the *path* is made to involve minimum columns, then it satisfies the cardinality of feature subset reaching a minimum. This completes the proof.

According to Theorem 1, the optimal feature subset selection problem can be transformed into a search for a *path* that involves the minimum number of columns in $EM(P, N)$. If we use the greedy algorithm, then it equals to select the columns with maximal number of β element.The proposed heuristic algorithm for feature selection is described in Algorithm 1. Let τ be the threshold for the number of remaining β elements, F_{sup} be the selected feature subset in one iteration, Fs be the final result, $|P|$ denotes the number of native classes in Positive class P and F is the full feature set.

Algorithm 1. Feature selection algorithm for learning from examples

Step 1. Initialization: $Fs \leftarrow \Phi$
Step 2. Positive class P and Negative class N are specified, extension matrix $EM(P, N)$ is constructed, $\beta \geq Sim(P, N|F)$ is calculated, and let $F_{sup} \leftarrow \Phi$;
Step 3. Find i'th column with maximal number of β elements, which is corresponding to feature F_i, $F_{sup} = F_{sup} \cup F_i$.
(If there is more than one column with the maximal number of β elements, select one column such that the sum of its β element values is minimum);
Step 4. Remove the rows that include an β element in the selected i'th column of $EM(P, N)$, and then get updated extension matrix $EM(P, N)$.
Step 5. **If** $EM(P, N) = \Phi$ or the number of β elements in $EM(P, N)$ is less than τ
$Fs = Fs \cup F_{sup}$
Else
Go to *step 3*
End
Step 6. **If** $|P| \geq 2$
Go to *step 2*
Else
output Fs
End

Noting that, in the implementation of the proposed heuristic feature selection algorithm for learning from examples, the extension matrix $EM(P, N)$ does not have to be really generated in memory. The time consumption exists in the



(a) (b)

Fig. 1. Gender classification: (a)Examples from face probe sets; (b) The accuracy rates on probe sets

number of β elements and removal rows need to be aggregated. Now, suppose the number of the classes is c, the number of training examples is S and the number of features is n, and if the number of examples in each native class is equal, then the total number of rows in extension matrix for all iterations is $((c-1)/2c)S^2$, and the average time complexity of the algorithm is approximate $\theta(n((c-1)/2c)S^2) \approx \theta(nS^2)$.

3 Experiments and Results

In this section, we will compare the proposed method with other two classic feature selection methods on classification performance and dimensionality reduction via benchmark data sets and facial image for gender classification. The selected methods are ReliefF [10,11] and the newly presented algorithm in [12] , we call it as Mitra's. These algorithms are well-known in machine learning.

Four bench-mark real-world data sets, DIAB, WDBC, Wine and Sonar, are downloaded from UCI Machine Learning Repository [13], and high-dimensional facial data sets for gender classification are used. The gallery sets used for training include 100 male examples and 100 female examples, which have the same vector dimension of 1584 with the gabor filter. The probe sets used for testing include 11 kinds of facial data with various facial views and expressions, such as front, down 10 degrees, smiling, closed eyes, opened mouth, right 10 degrees, right 20 degrees, right 30 degrees, up 10 degrees, up 20 degrees and up 30 degree. The number of testing samples in these probe sets are 1278, 820, 805, 815, 805, 814, 815, 805, 819, 816 and 816 respectively, and these probe sets are numbered as 1-11. Some examples are shown in Figure 1(a).

We will validate the performance of feature selection algorithm using K-NN classifiers with different K values and 5-fold cross-validation is used in the experiment. For gender classification, traditional Support Vector Machines (SVM)[14] is adopted, and the parameter C is set to 1. All the algorithms are implemented using Matlab compiler, the termination terms for the proposed algorithm is the number of remaining β elements less than the 1% and 1/100000 of original

number of β elements over benchmark data sets and facial data sets respectively. The termination term is dependent on the characteristics of data set, which can be determined by cross validation.

The experimental results are shown in Table 1 for benchmark data sets, which are corresponding to K=1, 3 and 5 for K-NN classifier. For gender classification, the detailed accuracy rates on eleven probe sets are displayed in Figure 1(b) and the dimensionality reduction rates for The proposed algorithm, Mitra's and ReliefF are 96.65, 85.23 and 93.94, respectively. From the experimental results, the following four observations are obtained.

Table 1. Experimental results for K-NN classifier with K=1, 3, 5 for benchmark data sets, Mean: mean accuracy, DR: Dimensionality Reduction

Data Sets	Algorithms	K=1		K=3		K=5	
		Mean%/Var.	DR%	Mean%/Var.	DR%	Mean%/Var.	DR%
DIAB	Mitra's (k=4)	**66.17**/0.22	50.00	68.21/0.20	50.00	71.14/0.20	50.00
	ReliefF	59.92/0.24	**75.00**	67.68/0.24	**75.00**	70.89/0.20	**75.00**
	The proposed	65.76/ 0.22	**75.00**	**71.82**/ 0.20	**75.00**	**72.08**/ 0.20	**75.00**
WDBC	Mitra's (k=20)	83.31/0.13	66.67	86.71/0.11	66.67	86.14/0.12	66.67
	ReliefF	84.51/0.13	**90.00**	89.17/0.10	90.00	88.65/0.10	**90.00**
	The proposed	**89.93**/0.09	**90.00**	**91.18**/0.08	**91.33**	**90.84**/0.08	**90.00**
Wine	Mitra's (k=7)	68.86/0.21	53.85	66.20/0.21	53.85	62.52/0.22	**53.85**
	ReliefF	72.86/0.20	53.85	69.65/0.20	53.85	76.70/0.18	**53.85**
	The proposed	**75.89**/0.18	**56.92**	**70.18**/0.18	**60.00**	**77.44**/0.18	**53.85**
Sonar	Mitra's (k=45)	62.14/0.22	91.33	60.54/0.22	91.67	55.60/0.24	91.67
	ReliefF	64.14/0.23	90.00	**66.83**/0.22	90.00	**71.92**/0.19	90.00
	The proposed	**70.78**/0.20	**92.67**	65.96/0.21	**92.33**	69.77/0.20	**92.67**

- The proposed evaluation criterion and search strategy is efficient for feature selection in learning from examples with continuous values.
- The proposed algorithm always can get better trade-off between the classification accuracy and dimensionality reduction, and in most cases it can get highest classification accuracy or highest dimensionality reduction without sacrificing dimensionality reduction or classification performance for different K values in K-NN classifier.
- For gender classification, in most cases, the proposed method obtains highest accuracy than ReliefF and Mitra's with highest dimensionality reduction.
- The proposed algorithm can obtain higher performance for different classifiers, such as K-NN classifier and SVM.

4 Conclusions

The paper presents an algorithm to perform feature selection for learning from examples with continuous values, where the evaluation criterion is both the overlapping (similarity) between classesand the cardinality of optimal feature subset

are minimum. The proposed similarity measure is reasonable and has low time complexity. And extension matrix is extended to deal with continuous value and adopted as search strategy. The algorithm is proved by theory and it has been shown to get better trade-off between classification accuracy and dimensionality reduction for different classifiers.

Acknowledgment

We gratefully thank OMRON Cooperation for supplying facial images. This research was partially supported by the Natural Science Fund for Colleges and Universities in JiangSu Province via the grants 08KJB520007, Scientific Research Foundation of Nanjing University of Posts and Telecommunications via the grants NY207137, National Natural Science Foundation of China via 40801149, JiangSu Province Bureau of Surveying and Mapping via JSCHKY200810.

References

1. Liu, H., Yu, L.: Toward Integrating Feature Selection Algorithms for Classification and Clustering. IEEE Trans. KDE 17(4), 491–502 (2005)
2. Kohavi, R., John, G.: Wrapper for Feature Subset Selection. Artificial Intelligence 97, 234–273 (1997)
3. Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. Journal of Machine Learning Research 3, 1158–1172 (2003)
4. Blum, A.L., Rivest, R.L.: Training a 3-Node Neural Networks is NP-Complete. Neural Network 5, 117–127 (1992)
5. Tsang, E.C.C., Yesung, D.S., Wang, X.Y.: OFFSS: Optimal Fuzzy-Valued Feature Subset Selection. IEEE Trans. on Fuzzy Systems 11(2), 202–213 (2003)
6. Chen, Y.X., Wang, J.Z.: A Region-based Fuzzy Feature Matching Approach for Content-Based Image Retrieval. IEEE Trans. PAMI 24(9), 1252–1267 (2002)
7. Molina, L.C., Belanche, L.A., Nebot: Feature Selection Algorithm: a Survey and Experimental Evaluation. In: Proc. of IEEE ICDM, pp. 306–313 (2002)
8. Hong, J.R.: AE1: an Extension Matrix Approximate Method for the General Covering Problem. International Journal of Computer & Information Science 12, 421–437 (1985)
9. Michalski, R.S., Mozetic, I., Hong, J.R.: The Multi-Purpose Incremental Learning System AQ_{15} and Its Testing Application to Three Medical Domains. In: Proc. of AAAI (August 1986)
10. Kononerko, I.: Estimating Attributes: Analysis and Extension of RELIEF. In: Bergadano, F., De Raedt, L. (eds.) ECML 1994. LNCS, vol. 784, pp. 171–182. Springer, Heidelberg (1994)
11. Sikonja, M.K., Kononenko, L.: An Adaptation of Relief for Attribute Estimation in Regression. In: Proc. of ICML, pp. 296–304 (1997)
12. Mitra, P., Murthy, C.A., Pal, S.K.: Unsupervised Feature Selection Using Feature Similarity. IEEE Trans. PAMI 24(3), 301–312 (2002)
13. Merz, C.J., Murphy, P.M.: UCI repository of machine learning database (1996), `http://www.ics.uci.edu/mlearn/MLRepository.html`
14. Chang, C.C., Lin, C.J.: LIBSVM: a Library for Support Vector Machines (2002), `http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.ps.gz`

Application-Independent Feature Construction from Noisy Samples*

Dominique Gay[1], Nazha Selmaoui[1], and Jean-François Boulicaut[2]

[1] University of New Caledonia, ERIM EA3791, PPME EA3325,
BP R4, 98851 Nouméa, New Caledonia
{dominique.gay, nazha.selmaoui}@univ-nc.nc
[2] INSA-Lyon, LIRIS CNRS UMR5205
69621 Villeurbanne Cedex, France
jean-francois.boulicaut@insa-lyon.fr

Abstract. When training classifiers, presence of noise can severely harm their performance. In this paper, we focus on "non-class" attribute noise and we consider how a frequent fault-tolerant (FFT) pattern mining task can be used to support noise-tolerant classification. Our method is based on an application independent strategy for feature construction based on the so-called δ-free patterns. Our experiments on noisy training data shows accuracy improvement when using the computed features instead of the original ones.

1 Introduction

Real-world data may contain noise. When dealing with classification tasks, it has been shown that presence of noise in the data may have a negative impact on the performance of classifiers learned from the data [1]. One may differentiate two main types of noise: class noise when noise affects the class label, and attribute noise when noise affects all attributes but the class label. Many solutions have been proposed to tackle class noise, e.g., by noise elimination or noise correction (see [1] for a survey) and more recently by instance weighting [2]. On the other hand, others approaches aim at solving the problem of attribute noise by noise identification and modeling [3,4], and noise cleansing [1,5]. In this paper, we focus on Boolean attribute noise problem and we address the following important question: *how to learn accurate predictive models from attribute-noisy data sets?*

Instead of removing noisy instances or correcting noisy values, we propose a method to cope with attribute noise without changing or removing any attributes values in the training data. Our approach combines two recent advances in fault-tolerant itemset mining and feature construction. The goal of *fault-tolerant itemset* mining [6] is to support the discovery of relevant frequent itemsets in noisy binary data (see, e.g., [7] for a recent survey). Among others, an extension to (frequent) closed set mining towards fault-tolerance has been studied in [8] that enables a bounded number (δ) of errors per item/attribute. It is based on the

* This work is partly funded by French ANR contract MDCO-2007 Bingo2.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 965–972, 2009.
© Springer-Verlag Berlin Heidelberg 2009

so-called δ-free sets, i.e., the approximate condensed representation of frequent itemsets introduced in [9]. Using such patterns is the core of our approach to robust *feature construction*. Following the proposals from, for instance, [10,11,12], we consider that attribute sets may be more relevant than single attributes for class discrimination. Then, pattern types based on the so-called closedness properties enable to avoid redundant features in an application-independent setting. In this paper, we investigate further the use of δ-freeness (and thus δ-closedness) when considering feature construction from noisy training data. From that perspective, it extends our previous work [11] which focussed on noise-free samples only. Our proposal can be summarized as follows. First, we mine non-redundant fault-tolerant patterns based on (δ)-free itemsets. Then, we process these patterns to compute new features that will enable to encode a new training set. Finally, classical classification algorithms can be applied. This proposal is yet another contribution to pattern-based classification. In pioneering studies, association rules were considered [13]. Since then, emerging patterns [14] (see [15] for a survey and [16] for a noise tolerant pattern-based approach), and more recently condensed representations of frequent itemsets have been studied [10,11,12]. The paper is organized as follows. Section 2 provides the needed definitions before the description of our proposal in Section 3. Section 4 provides an experimental validation on selected `UCI` (http://archive.ics.uci.edu/ml/) data. Section 5 concludes.

2 Preliminaries

Let us consider a binary database $r = \{\mathcal{T}, \mathcal{I}, \mathcal{R}\}$, where $\mathcal{T}$ is a set of transactions (or objects) described by a set $\mathcal{I}$ of Boolean items (or attributes) and $\mathcal{R} : \mathcal{T} \times \mathcal{I} \mapsto \{0, 1\}$. When $\mathcal{R}(t, i) = 1$, we say that transaction t contains item i. An itemset $I \in \mathcal{I}$ is a set of items. The frequency of itemset $I \in \mathcal{I}$ is $freq(I, r) = |Objects(I, r)|$ where $Objects(I, r) = \{t \in \mathcal{T} \mid \forall i \in \mathcal{I} \quad \mathcal{R}(t, i) = 1\}$. I is said to be γ-frequent if $freq(I, r) \geq \gamma$.

Definition 1 (association rule, δ-strong rule, δ-free itemset). *An* association rule π *on* r *is an expression* $I \Rightarrow J$, *where* $I \subseteq \mathcal{I}$ *and* $J \subseteq \mathcal{I} \setminus I$. *The* frequency *of the rule* π *is* $freq(I \cup J, r)$ *and its* confidence *is* $conf(\pi, r) = freq(I \cup J, r)/freq(I, r)$. *Let* δ *be an integer. A* δ-strong rule *is an association rule of the form* $I \Rightarrow^{\delta} J$ *which is violated in at most* δ *objects, and where* $I \subseteq \mathcal{I}$ *and* $J \subseteq \mathcal{I} \setminus I$. *An itemset* $I \subseteq \mathcal{I}$ *is a* δ-free itemset *iff there is no* δ*-strong rule which holds between its proper subsets. When* $\delta = 0$, δ *is omitted, and we talk about* strong rules, *and* free itemsets.

First introduced in [9], δ-free itemsets and δ-strong rules have been designed as an approximate condensed representation for frequency queries. δ-freeness is a generalization of the key pattern concept [17] (case $\delta = 0$) and it can also be discussed in terms of equivalence classes.

Definition 2 (δ-closure, equivalence class). *Let* δ *be an integer. The* δ-closure *of an itemset* I *on* r *is* $cl_\delta : \mathcal{P}(\mathcal{I}) \rightarrow \mathcal{P}(\mathcal{I})$ *s.t.* $cl_\delta(I, r) = \{i \in \mathcal{I} \mid$

$freq(I,r) - freq(I \cup \{i\},r) \leq \delta\}$. *When* $\delta = 0$, $cl_0(I,r) = \{i \in \mathcal{I} \mid freq(I,r) = freq(I \cup \{i\},r)\}$ *and it corresponds to the well-known closure operator. We can also group itemsets by* δ-closure equivalence classes *(δ-CECs): two δ-free itemsets I and J are said δ-equivalent ($I \sim_{cl_\delta} J$) if $cl_\delta(I,r) = cl_\delta(J,r)$.*

Once again, when $\delta = 0$, we get the formalization of closure equivalence classes from [17]. We can also derive δ-strong rules from δ-CECs (i.e., from δ-free itemsets and their δ-closures). Indeed, we have a δ-strong association rule between a δ-free itemset and each element of its δ-closure. In [7,8], δ-free itemsets and their associated δ-closures are combined to define the so-called δ-bi-sets.

Definition 3 (frequent δ-bi-set). *A bi-set (T,I) such that $T \subseteq \mathcal{T}$ and $I \subseteq \mathcal{I}$ is a frequent δ-bi-set iff $I = I_1 \cup I_2$, I_1 is a γ-frequent δ-free itemset, $cl_\delta(I_1,r) = I$ and $Objects(I_1,r) = T$.*

δ-bi-sets appear as an extension of the so-called formal concepts or associated closed itemsets. Indeed, they appear as examples of maximal combinatorial rectangles of 1 values having at most δ 0 per column [8]. We now consider how we use δ-CECs to derive robust features.

3 Deriving Features from δ-closure Equivalence Classes

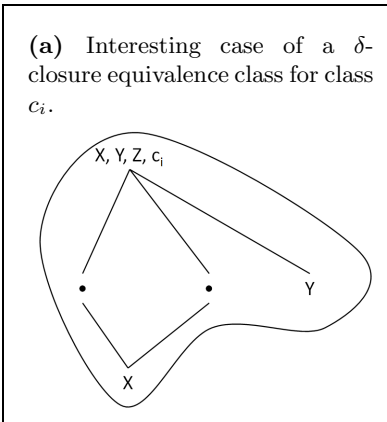
In order to manage classification tasks, we are interested in relevant δ-strong association rules contained in δ-CECs. Figure 1(a) shows a typical case of an interesting δ-CEC: δ-free itemsets X and Y do not contain a class attribute and their (equal) δ-closure (X,Y,Z,c_i) contains a class attribute c_i. Indeed, we may derive two potentially interesting δ-strong rules : $X \rightarrow c_i$ and $Y \rightarrow c_i$. According to the formalization from [18], $\pi : X \rightarrow c_i$ is a δ-strong characterization rule (δ-SCR) if c_i is a class attribute and body X is minimal. X is minimal if there is no other frequent rule $\pi' : Y \rightarrow c_i$ s.t. $Y \subseteq X$ and $conf(\pi',r) \geq 1 - \frac{\delta}{\gamma}$. Moreover, when $\delta \in [0; \lfloor \gamma/2 \rfloor[$, the set of δ-SCRs does not contain included or equal body conflicts.

Defining δ-SCR only based on the confidence measure is not sufficient for prediction. Therefore, we propose to exploit a Growth rate measure Gr which has been already proved useful in such a context [14]. The Growth rate of $\pi : X \rightarrow c_i$ is defined as a ratio of relative frequencies as follows:

$$Gr(\pi,r) = \frac{freq_r(X, r_{c_i})}{freq_r(X, r \setminus r_{c_i})}$$

where r_{c_i} is the database restricted to objects of Class c_i. In [19], Gr is set in the general framework of the so-called δ-dependent measures. Such measures depend on the rule antecedent frequency (γ) and the rule number of exceptions (δ) following two principles: *(i)*, when γ is fixed, $Gr(\pi,r)$ increases with $freq(\pi,r)$ and *(ii)*, when δ is fixed, $Gr(\pi,r)$ increases with γ.

This leads us to lower bounds for several interestingness measures (including Growth rate) w.r.t. γ and δ values (see [19] for details). In Figure 1(b), contingency table for $\pi : X \rightarrow c_i$ a δ-strong rule concluding on a class attribute c_i



(a) Interesting case of a δ-closure equivalence class for class c_i.

(b) Contingency table forδ-strong rule $X \rightarrow^{\delta} c_i$.

$X \rightarrow c_i$	c_i	$\bar{c_i}$	Σ
X	$\gamma - \delta$	δ	γ
$\bar{X}$	.	.	.
Σ	$\lvert r_{c_i} \rvert$	$\lvert r \setminus r_{c_i} \rvert$	$\lvert r \rvert$

Fig. 1. Information contained in δ-closure equivalence classes

shows that, by construction, we have a lower bound $(\gamma - \delta)$ for $freq(X, r_{c_i})$, an upper bound δ for $freq(X, r \setminus r_{c_i})$ and other deductible bounds for dotted cells. Moreover, we can deduce a lower bound for Gr and $conf$ measure. Indeed,

$$Gr(\pi, r) \geq \frac{\gamma - \delta}{\delta} \cdot \frac{|r \setminus r_{c_i}|}{|r_{c_i}|} \quad \text{and} \quad conf(\pi, r) \geq 1 - \delta/\gamma$$

Then, with few deduction steps, we get:

$$Gr(\pi, r) \geq 1 \quad \Longrightarrow \quad \delta < \gamma \cdot \frac{|r \setminus r_{c_j}|}{|r|} \tag{1}$$

$$conf(\pi, r) \geq 1/2 \quad \Longrightarrow \quad \delta < \gamma/2 \tag{2}$$

where c_j is the majority class. Thus, for a given frequency threshold γ, γ-frequent δ-free itemsets whose δ-closures contain a class attribute s.t. δ satisfies Equations (1) and (2) are emerging patterns. In the rest of the paper, γ and δ values are constrained w.r.t. Equations (1) and (2).

Our feature construction process may now be summarized with Algorithm 1. Procedure `FeaturesExtraction` (Line 1) mines all γ-frequent δ-free itemsets I' that are bodies of δ-strong characterization rules in r. This step is performed efficiently using straightforward extension of `AC-like`[1]. `AC-like` is an implementation of a levelwise algorithm presented in [9] that benefits from anti-monotonic properties of δ-freeness and γ-frequency to compute γ-frequent δ-free itemsets. Moreover, since we are interested in minimal itemsets whose δ-closures contain a class attribute, there is no need to check supersets of itemsets selected in a precedent level.Then, each I' becomes a new descriptor for r', and (Line 1) the value of I' for a transaction t is the proportion of items in I' that are verified by t in r. *Items* is the dual operator for *Objects*. Now, $\mathcal{R}' \mapsto [0; 1]$ and

[1] http://liris.cnrs.fr/∼turing

Algorithm 1: Building new data set with noise-tolerant features

input : A binary database $r = \{\mathcal{T}, \mathcal{I}, \mathcal{R}\}$,
two integers γ and δ for frequency and errors thresholds
output: A numeric database $r' = \{\mathcal{T}, \mathcal{I}', \mathcal{R}'\}$ made of noise-tolerant features as descriptors

begin
  $\mathcal{I}' \longleftarrow$ `FeaturesExtraction`(r, γ, δ);
  for $t \in \mathcal{T}$ **do**
    for $I' \in \mathcal{I}'$ **do**
      $\mathcal{R}'(t, I') \longleftarrow \frac{|I' \cap \text{Items}(t,r)|}{|I'|}$;
  $r' \longleftarrow \{\mathcal{T}, \mathcal{I}', \mathcal{R}'\}$;
end

$\mathcal{R}'(t, I') \in \{0, \frac{1}{p}, \ldots, \frac{p-1}{p}, 1\}$ with $p = |I'|$). Finally, r' is the new database made of noise-tolerant features, ready for a classification learning step.

4 Experimental Validation

To assess the noise-tolerance of our process, we ran our feature construction process (`NTFC`) on several `UCI` data sets and several noisy versions of them. We want to learn accurate predictive models despite of noisy samples. Therefore, in our experiments we deal with attribute-noisy training sets and clean test sets. We add random noise at different levels only on attributes and only in training sets. For a data set d and a x% noise level ($x \in \{10, 20, 30, 40, 50\}$), each attribute value has x% chances to be changed (within its range values) in each transaction of the training set[2]. When dealing with continuous attributes, we first discretize the training data: each attribute is split into several intervals using entropy-based splitting method [20]. Then, we add x% noise and perform a simple binarization. Finally, we run `C4.5` decision tree [21] and Naive Bayes classifier (`NB`) on each noisy version of the data set and on each data set enhanced by new descriptors generated with Algorithm 1. All pre-processing steps (adding noise, discretization, binarization) and accuracy results for `NB` and `C4.5` classifiers are obtained with 10-folds stratified cross-validation – using `Weka` platform [22].

A strategy for setting δ: Automatically setting frequency threshold is still an open question (see [23,24] for preliminary results in that direction). Given a frequency threshold γ, how can we determine relevant δ values? Evolution of δ-dependent interestingness measure (such as growth rate) w.r.t. δ is well known. Decreasing δ implies higher values for Gr but such interesting patterns could be rare, especially in noisy data sets. When increasing δ, extracted patterns tend to match with noisy patterns in training data, but higher δ values tend to be less relevant (with low Gr values). In Figure 2, we plotted training accuracy results w.r.t. δ values for `tic-tac-toe` data set. As expected, accuracy on training data increases with δ until stabilization (or slowing down). δ values around stabilization are interesting since lower values lead to less accurate models and

[2] It does not mean that x% of values have been modified.

Table 1. Accuracy results : original v.s. NTFC processed data

Dataset	C4.5	NTFC & C4.5 (Avg)	Max	NB	NTFC & NB (Avg)	Max
breast-w (f:20-40/5)	95.57	**95.62**	**95.85**	97.28	*96.59*	*96.85*
10%	95.14	*94.59*	**95.14**	96.99	**97.22**	**97.56**
20%	93.28	*92.31*	*92.99*	96.99	**97.05**	**97.14**
30%	89.56	**91.00**	**91.85**	96.85	**97.14**	**97.28**
40%	87.70	**87.73**	**88.41**	96.42	*94.99*	**97.28**
50%	87.56	*86.30*	**87.56**	96.28	**96.51**	**97.21**
colic (f:25-50/5)	85.04	*84.31*	**85.85**	79.90	**82.73**	**85.31**
10%	83.14	*82.95*	**85.29**	78.81	**82.59**	**84.77**
20%	82.04	*80.88*	**83.12**	78.81	**81.28**	**85.31**
30%	80.95	*80.78*	**82.85**	79.09	**81.37**	**83.96**
40%	80.93	**81.72**	**84.21**	78.00	**82.32**	**84.22**
50%	81.53	**82.04**	**83.66**	73.92	**80.56**	**82.31**
heart-c (f:25-45/5)	80.47	**82.48**	**84.13**	81.79	**83.20**	**84.12**
10%	78.83	**80.17**	**81.18**	82.78	**83.39**	**84.78**
20%	75.85	**79.46**	**82.18**	83.12	**83.26**	**84.46**
30%	77.19	**78.40**	**80.21**	84.45	*83.93*	**84.82**
40%	77.50	*76.40*	**78.52**	84.45	*82.63*	**84.48**
50%	72.28	**75.53**	**77.52**	79.52	**79.93**	**88.20**
heart-h (f:10-22/2)	75.55	**79.60**	**81.64**	84.07	*82.50*	*83.03*
10%	77.62	**78.83**	**80.56**	83.73	*80.70*	*81.31*
20%	76.60	**78.24**	**80.29**	83.39	*82.30*	*83.00*
30%	75.56	**79.08**	**80.00**	84.05	*83.10*	**84.34**
40%	74.17	**79.19**	**80.96**	83.37	*80.51*	*82*
50%	73.83	**79.93**	**81.68**	79.95	**80.17**	**82.34**
heart-s (f:22-32/2)	81.85	**83.33**	**86.67**	81.48	**82.10**	**84.45**
10%	80.74	**81.17**	**84.45**	81.48	*81.17*	**85.92**
20%	78.52	**79.75**	**80.74**	82.59	*81.60*	**84.07**
30%	74.07	**74.07**	**76.30**	81.11	**81.92**	**84.07**
40%	72.96	*70.15*	**73.33**	81.85	*71.70*	*72.59*
50%	59.26	**64.07**	**65.55**	55.55	**61.18**	**65.18**
iris (f:7-30)	93.33	**93.33**	**93.33**	92.67	**94.00**	**94.00**
10%	92.67	**93.00**	**94.67**	92.67	**92.67**	**94.00**
20%	90.00	*89.22*	**93.33**	92.67	**93.11**	**94.67**
30%	90.00	*86.50*	*89.33*	92.67	*91.33*	*92.00*
40%	86.00	*84.67*	*84.67*	92.67	*88.00*	*88.33*
50%	84.00	*77.33*	*77.33*	94.00	*77.33*	*77.33*
tic-tac-toe (f:25-40/5)	93.21	**99.40**	**100.00**	68.47	**79.72**	**81.73**
10%	92.06	**97.65**	**99.17**	68.57	**75.99**	**78.81**
20%	88.95	**96.26**	**97.18**	70.14	**73.81**	**74.42**
30%	82.78	**93.00**	**95.62**	71.92	**73.56**	**74.53**
40%	77.98	**89.43**	**91.55**	74.11	*73.56*	**74.83**
50%	71.39	**83.82**	**85.60**	70.57	**71.16**	**71.39**
wine (f:9-18)	91.08	**91.53**	**93.76**	98.89	*92.92*	*93.86*
10%	91.63	**91.87**	**94.38**	96.6	*93.24*	*94.41*
20%	91.01	**91.74**	**92.71**	96.04	*94.42*	*95*
30%	84.37	**88.15**	**91.04**	94.38	*91.73*	**94.97**
40%	82.02	**88.28**	**88.82**	93.27	*91.42*	**93.79**
50%	82.12	**85.11**	**87.58**	88.10	**89.29**	**90.42**

higher values bring nothing more. Let δ_{opt} denote these values. These interesting values depends on the amount of noise in data. Since in real case, noise level is not known a priori, a reasonable way to reach these δ_{opt} values is (1) to increase δ starting from 0, and (2) to check evolution of training data accuracy for stabilization.

Accuracy results: Accuracy results comparison are shown in Tab. 1. For each data set, we used different frequency thresholds indicated in parenthesis: (f:x-y/z) means that absolute frequency thresholds vary from x to y with step z (when regular). NTFC accuracy results are reported in two columns: *Avg* for average accuracy overall tested γ values for δ_{opt} values and *Max* for maximal accuracy overall γ and δ values.

First, we remark that there often exists a combination of γ and δ values (col. Max) for which NTFC & C4.5 (44/48) and NTFC & NB (36/48) achieve better accuracy results than on original data. Second, even if less impressive, our simple

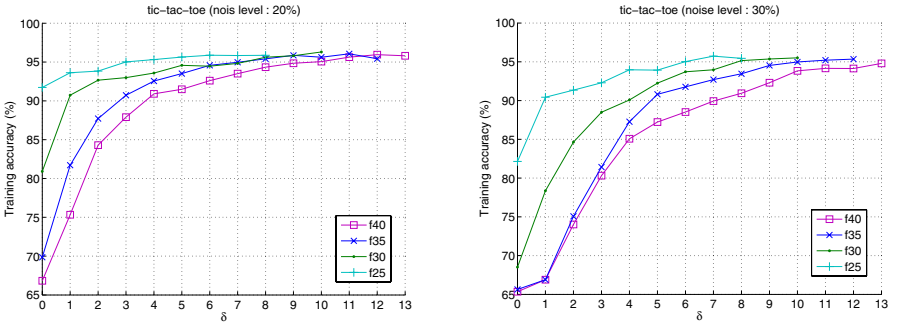


Fig. 2. Evolution of `NTFC & C4.5` accuracy on training database w.r.t. δ for various frequency thresholds and noise levels on `tic-tac-toe` data set

strategy for setting δ (col. Avg) also allows us to catch better accuracy results on noisy data sets enhanced by `NTFC` (36/48 for `C4.5` and 27/48 for `NB`).

5 Conclusion

In this paper, we have proposed an original process (`NTFC`) combining recent advances in fault-tolerant pattern mining and feature construction to handle classification tasks in attribute-noisy environments. We enhance data description with new features based on relevant fault-tolerant patterns, say γ-frequent δ-free itemsets and their δ-closures. Experimental results show that classification tools such that `NB` and `C4.5` achieve higher accuracy results with an enhanced description of data (i.e., using noise-tolerant features) than with original data. δ values have shown to be related to the amount of noise in the data. Finally, we have given a strategy to automatically set δ values that are relevant w.r.t. the noise level in data samples – thus supporting a parameter tuning dedicated to our process.

References

1. Zhu, X., Wu, X.: Class noise vs. attribute noise : A quantitative study. Artificial Intelligence Revue 22, 177–210 (2004)
2. Rebbapragada, U., Brodley, C.E.: Class noise mitigation through instance weighting. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS, vol. 4701, pp. 708–715. Springer, Heidelberg (2007)
3. Kubica, J., Moore, A.W.: Probabilistic noise identification and data cleaning. In: Proceedings ICDM 2003, pp. 131–138. IEEE Computer Society, Los Alamitos (2003)
4. Zhang, Y., Wu, X.: Noise modeling with associative corruption rules. In: Proceedings ICDM 2007, pp. 733–738. IEEE Computer Society, Los Alamitos (2007)

5. Yang, Y., Wu, X., Zhu, X.: Dealing with predictive-but-unpredictable attributes in noisy data sources. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS, vol. 3202, pp. 471–483. Springer, Heidelberg (2004)
6. Yang, C., Fayyad, U.M., Bradley, P.S.: Efficient discovery of error-tolerant frequent itemsets in high dimensions. In: Proceedings KDD 2001, pp. 194–203. ACM Press, New York (2001)
7. Besson, J., Pensa, R.G., Robardet, C., Boulicaut, J.F.: Constraint-based mining of fault-tolerant patterns from boolean data. In: Bonchi, F., Boulicaut, J.-F. (eds.) KDID 2005. LNCS, vol. 3933, pp. 55–71. Springer, Heidelberg (2006)
8. Pensa, R.G., Robardet, C., Boulicaut, J.F.: Supporting bi-cluster interpretation in 0/1 data by means of local patterns. Intelligent Data Analysis 10, 457–472 (2006)
9. Boulicaut, J.F., Bykowski, A., Rigotti, C.: Approximation of frequency queries by means of free-sets. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS, vol. 1910, pp. 75–85. Springer, Heidelberg (2000)
10. Selmaoui, N., Leschi, C., Gay, D., Boulicaut, J.F.: Feature construction and delta-free sets in 0/1 samples. In: Todorovski, L., Lavrač, N., Jantke, K.P. (eds.) DS 2006. LNCS, vol. 4265, pp. 363–367. Springer, Heidelberg (2006)
11. Gay, D., Selmaoui, N., Boulicaut, J.F.: Feature construction based on closedness properties is not that simple. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS, vol. 5012, pp. 112–123. Springer, Heidelberg (2008)
12. Cheng, H., Yan, X., Han, J., Hsu, C.W.: Discriminative frequent pattern analysis for effective classification. In: Proceedings ICDE 2007, pp. 716–725. IEEE Computer Society, Los Alamitos (2007)
13. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: Proceedings KDD 1998, pp. 80–86. AAAI Press, Menlo Park (1998)
14. Dong, G., Li, J.: Efficient mining of emerging patterns: discovering trends and differences. In: Proceedings KDD 1999, pp. 43–52. ACM Press, New York (1999)
15. Ramamohanarao, K., Fan, H.: Patterns based classifiers. World Wide Web 10, 71–83 (2007)
16. Fan, H., Ramamohanarao, K.: Noise tolerant classification by chi emerging patterns. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS, vol. 3056, pp. 201–206. Springer, Heidelberg (2004)
17. Bastide, Y., Taouil, R., Pasquier, N., Stumme, G., Lakhal, L.: Mining frequent patterns with counting inference. SIGKDD Explorations 2, 66–75 (2000)
18. Crémilleux, B., Boulicaut, J.F.: Simplest rules characterizing classes generated by delta-free sets. In: Proceedings ES 2002, pp. 33–46. Springer, Heidelberg (2002)
19. Hébert, C., Crémilleux, B.: Optimized rule mining through a unified framework for interestingness measures. In: Tjoa, A.M., Trujillo, J. (eds.) DaWaK 2006. LNCS, vol. 4081, pp. 238–247. Springer, Heidelberg (2006)
20. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continous-valued attributes for classification learning. In: Proceedings IJCAI 1993, pp. 1022–1027. Morgan Kaufmann, San Francisco (1993)
21. Quinlan, J.R.: C4.5: Programs for machine learning. Morgan Kaufmann, San Francisco (1993)
22. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
23. Cerf, L., Gay, D., Selmaoui, N., Boulicaut, J.F.: A parameter free associative classifier. In: Song, I.-Y., Eder, J., Nguyen, T.M. (eds.) DaWaK 2008. LNCS, vol. 5182, pp. 293–304. Springer, Heidelberg (2008)
24. Zhang, S., Wu, X., Zhang, C., Lu, J.: Computing the minimum-support for mining frequent patterns. Knowledge and Information Systems 15, 233–257 (2008)

Estimating Optimal Feature Subsets Using Mutual Information Feature Selector and Rough Sets

Sombut Foitong[1], Pornthep Rojanavasu[1], Boonwat Attachoo[1], and Ouen Pinngern[2]

[1] Department of Computer Engineering, Faculty of Engineering,
King Mongkut's Institute of Technology Ladkrabang, Bangkok, 10200, Thailand
[2] Department of Computer Science, Faculty of Science, Ramkhamhaeng University,
Bangkok 10240, Thailand
s8060058@kmitl.ac.th, s8060022@kmitl.ac.th,
boonwat@kmitl.ac.th, ouen@ru.ac,th

Abstract. Mutual Information (MI) is a good selector of relevance between input and output feature and have been used as a measure for ranking features in several feature selection methods. Theses methods cannot estimate optimal feature subsets by themselves, but depend on user defined performance. In this paper, we propose estimation of optimal feature subsets by using rough sets to determine candidate feature subset which receives from MI feature selector. The experiment shows that we can correct nonlinear problems and problems in situation of two or more combined features are dominant features, maintain an improve classification accuracy.

Keywords: Mutual Information, Rough sets, Nonlinear Problems, MLP, C4.5.

1 Introduction

Mutual information (MI) can be considered as advanced statistics to rank the salient features [1 – 3]. The main advantages of MI are the robustness to noise and transformation. The effective has been verified in [1], [4] and [5]. Batiti's mutual information feature selector [1]s (MIFS) use mutual information between inputs and outputs for classification problems. He demonstrated that MI can be very useful in feature selection problems and the MIFS can be used in any classifying systems for its simplicity whatever the learning algorithm may be. Kwak [2] verified that MIFS does not work in nonlinear problems. He proposed MIFS – U to improve MIFS for solving nonlinear problems. He applied Taguchi method [7] to MIFS – U in situation two or more combined features which are dominant features. However, this method spent a lot of times for training and testing MLP by several possible features subsets. The main disadvantage of MIFS and MIFS – U is that they are unable to estimate the appropriate feature subsets. The classification accuracy is applied to evaluate feature subsets, various machine learning algorithms may be used as selection criteria. However, these machine learning algorithms take a lot of computational time for each training process. Moreover, the MIFS – U can be mistaken when dealing with highly nonlinear problem.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 973–980, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Rough set theory [8] has been used successfully as a selection tool to discover data dependencies and has been shown to reduce the number of attributes contained in a dataset by purely structural methods. Given a dataset with discretized attribute values, by using rough sets it is possible to find a subset (termed a *reduct*) of the original attributes that are the most informative; all other attributes can be removed from the dataset with minimal information loss.

This paper illustrates the limitation of the MIFS and the MIFS – U using simple examples and proposes the usage of dependency of attribute from rough sets theory [8] which can breakdown this limitation and improve performance. The rest of this paper is organized as follow. Section 2 provides the basic information theory and rough sets theory including concepts of entropy, mutual information, and dependency of attribute. Section 3 shows detail of the MIFS and the MIFS – U and describe our proposed method. The proposed method is applied to several problems to show their effectiveness in Section 4. Conclusions and future works are discussed in Section 5.

2 Preliminaries

2.1 Entropy and Mutual Information

Let X, Y be discrete random variables having Ω and Ψ alphabets respectively. If the probability distribution of X is $p(x) = \Pr\{X = x\}$, $x \in \Omega$ then the entropy of X can be denoted as $H(X)$. For two discrete random variables X and Y with their joint probability distribution $p(x, y)$, the mutual information is:

$$I(X;Y) = \sum_{x\in\Omega}\sum_{y\in\Psi} p(x,y)\log\frac{p(x,y)}{p(x)p(y)}. \tag{1}$$

For continuous random variables, the entropy and mutual information are defined as

$$\begin{aligned} H(X) &= \int p(x)\log p(x)dx \\ I(X;Y) &= \iint p(x,y)\log\frac{p(x,y)}{p(x)p(y)}dxdy, \end{aligned} \tag{2}$$

It is quite difficult to calculate probability density functions ($p(x)$, $p(y)$, $p(x, y)$) and to perform integration. Therefore, we divide the continuous input feature into several discrete partitions and calculate the entropy and mutual information using the definitions for discrete values. The discretization is a process for converting from continuous to discrete. The effective of discretization depends on the numbers of partitions that divide the continuous space [1].

2.2 Rough Sets Theory

An information system $IS = (U, A)$ consists of a nonempty, finite set U of objects and a nonempty, finite set A of attributes a, such that $a: U \rightarrow V_a$, where V_a is the set of values that attribute a may take. The attribute set has two parts ($A = C \cup D$)

consisting of the *condition (C)* and *decision (D)* attributes. Let B be a subset of A. Then a B – indiscernibility relation IND (B) is defined as:

$$\text{IND}(B) = \{(x, y) \in U: a(x) = a(y), \forall a \in B\}. \tag{3}$$

It is clear that IND (B) partitions the universe U into equivalence classes:

$$[x]_B = \{x_i \in U: (x_i, x_j) \in \text{IND}(B)\}, x_i \in U. \tag{4}$$

Let $X \subseteq U$. X can be approximated using only the information contained within B by constructing the *B-lower* approximations of X:

$$\underline{B}X = \{x \in U : [x]_B \subseteq X\}, \tag{5}$$

Let $B \subseteq C$, then the B - positive regions of D can be defined as:

$$POS_B(D) = \bigcup_{X \in U/D} \underline{B}X \cdot \tag{6}$$

For B, $D \subseteq A$ it is said that D depends on B in a degree k ($0 \leq k \leq 1$), denoted $B \Rightarrow_k D$, if

$$k = \gamma_B(D) = \frac{|POS_B(D)|}{|U|} \cdot \tag{7}$$

If $k = 1$, D depends totally on B, if $0<k<1$, D depends partially (in a degree k) on B, and if $k = 0$, then D does not depend on B.

3 Feature Selection by Using MIFS – U and Rough Sets

In this section, we describe detail of the MIFS and the MIFS – U methods. We describe the limitations of those methods using several examples (*Example 1*, *2)*. Last section, we propose an algorithm using Rough Sets which can breakdown this limitation and estimate the appropriate feature subsets.

3.1 MIFS and the MIFS – U

Batiti [1] propose MIFS algorithm which improve the *greedy algorithm*. The algorithm of MIFS is the same as greedy algorithm except for step 4. The reader is referred to [1] for a full algorithm description of MIFS algorithm. Kwak [2] proposed the MIFS – U method for improve the MIFS method, which did not work well in nonlinear problems. He revised some steps in the MIFS algorithm. The reader is referred to [2] for a full algorithm description of MIFS – U algorithm. Although, when we apply to nonlinear problems [2] the MIFS – U showed better performance than the MIFS. However, both the MIFS and the MIFS – U do not work when two or more combined features, such as dominant features. Moreover, they are not able to obtain the correct result when they are applying to highly nonlinear problem. We illustrate the limitation of those methods as in *Example 1*, *2* in section 4.1. Therefore, we propose to solve these problems by using rough set as post-process module. The detail of algorithm will be described in section 3.2.

3.2 Proposed Algorithm (MIFS – U Rough)

This section, we describe concept of Rough Set apply to feature selection problem. We used the Variable Precision Rough Set Model (VPRS) [10], the aim to increasing discriminatory capacities of Rough Set theory by using parameter – controlled grades of conditional probabilities and to the robustness to noise.

The μ - positive region $POS_{\mu}^{B}(X)$ is given by:

$$POS_{\mu}^{B}(X) = \cup\{[x]_B \mid P(X \mid [x]_B) \geq \mu, and\ B \subseteq C\}$$
$$P(X \mid [x]_B) = \frac{|X \cap [x]_B|}{|[x]_B|}, \tag{8}$$

where $X \subseteq U/D$, μ is threshold values($0 \leq \mu \leq 1$) , and |x| is cardinality of x.

For dependency of attribute defined in (7), we revised to apply which μ - positive region and new define as:

$$\gamma_B^{\mu}(D) = \frac{\sum_{[x]_B \in IND(B)} P(X \mid [x]_B)}{|IND(B)|},\ [x]_B \subseteq POS_{\mu}^{B}(X) \text{ and } \forall X \in U/D. \tag{9}$$

The feature selection algorithm using the MIFS – U and Rough set are summarized as follows:

The MIFS – U and Rough set in Feature Selection (MIFS – U Rough)

1) (*Filtering*) If there are too many features, reduce the number of feature to twice the number we want to select by using MIFS – U.
2) Using Rough set select feature from Step 1) as follows:
   2.1) $R \leftarrow \phi$
   2.2) $F \leftarrow$ set of n features and let $depC = \gamma_F^{\mu}(D)$.
   2.3) *(Select of the first feature)* find the feature that maximize $\gamma_{\{f_i\}}^{\mu}(D)$ and greater than α, set $F \leftarrow F - \{f_i\}$, $R \leftarrow \{f_i\}$.
   2.4) *Flag = false*
   2.5) *While*($\gamma_R^{\mu}(D) < \delta * depC$)
      2.5.1) *If*($R = \phi$), then find the pair of features that maximize $\gamma_{\{f_i, f_j\}}^{\mu}(D)$, set $F \leftarrow F - \{f_i, f_j\}$, $R \leftarrow \{f_i, f_j\}$.
      2.5.2) *Else*
         a) For each features in *set F*, find the feature that maximize $\gamma_{R \cup \{f_i\}}^{\mu}(D)$ and greater than $\gamma_R^{\mu}(D)$, set $F \leftarrow F - \{f_i\}$, $R \leftarrow \{f_i\}$.
         b) *if* condition in a) is *false* and *Flag = false* ,then find the pair of features that maximize $\gamma_{\{f_i, f_j\}}^{\mu}(D)$, set $F \leftarrow F - \{f_i, f_j\}$, $R \leftarrow \{f_i, f_j\}$ and set *Flag = true*.
         c) Otherwise Stop;
   2.6) Output the set R containing the selected features.

In our experimental, we set $\mu = 0.85$, $\alpha = 0.2$ and $\delta = 0.99$. These parameters are fixed for all datasets. The δ is a parameter for approximate reduct, if δ is large value that approximate reduct closely to exact reduct, if δ is small then performance is down.

4 Experimental Results

In this section, we apply the MIFS – U Rough algorithm to some of the classification problems and illustrate the effectiveness of the proposed method when comparative with other methods.

4.1 Simple Problems

For example 1, we illustrate limitation of the MIFS and the MIFS – U methods when dominant features consisted has two or more combined features. Table 1 shows the result of MIFS, MIFS-U and MIFS-U Rough.

Example 1: Let X_1 and X_2 be random variables which uniformly distributed on [-1, 1] and assume that there input feature X_1, X_2 and $X_1^2 X_2$. The output belongs to class C

$$C = \begin{cases} 0 & if\ X_1 X_2 < 0 \\ 1 & if\ X_1 X_2 \geq 0. \end{cases}$$

When we take 1000 patterns and partition each input feature space into ten, and normalized the input feature to have the values in [0, 1]. The order of selection by the MIFS and the MIFS – U are illustrate in Table 1.

As in Table 1, both the MIFS and the MIFS – U select $X_1^2 X_2$ as the first feature rather than X_1 and X_2. However, we can see that the MIFS – U Rough select the correct feature subset for this problem, while the MIFS and the MIFS – U cannot select the correct input feature. For example 2, we show limitation of the MIFS and the MIFS – U methods when deal with highly nonlinear problem.

Table 1. Shows order of selection and feature selected for Example 1 using the MIFS and the MIFS – U, and MIFS – U Rough methods

Input Feature	X_1	X_2	$X_1^2 X_2$
MI with Output	0.003	0.007	0.011
MIFS ($\beta = 1$)	2	3	1
MIFS – U ($\beta = 1$)	3	2	1
Dependency value of features	0.0	0.0	0.0
Selected feature by MIFS – U Rough		$\{X_1, X_2\}$	

Example 2: Let $X = \{X_1, X_2, X_3, X_4\}$ be generated in this example, which take 2000 patterns belonging to two classes {0, 1}. For input feature X_1, 500 patterns were generated from the normal distribution N(-0.4, 0.3), and other 1500 patterns were generated from N(0.2, 0.3); for X_2, 500 patterns were generated from the normal distribution N(-0.1, 0.3), and other 1500 patterns were generated from N(0.2, 0.3). The input features $X_3 = X_1^2$, $X_4 = X_2^2$ and output belonging to class C

$$C = \begin{cases} 0 & if\ X_3 - X_4 < 0 \\ 1 & if\ X_3 - X_4 \geq 0. \end{cases}$$

Table 2. Shows order of selection and feature selected for Example 2 using the MIFS and the MIFS – U, and MIFS – U Rough methods

Input Feature	X_1	X_2	X_3	X_4
MI with Output	0.251	0.232	0.178	0.163
MIFS ($\beta = 1$)	1	4	3	2
MIFS – U ($\beta = 1$)	1	2	3	4
Dependency value of features	0.588	0.453	0.884	0.655
Selected feature by MIFS – U Rough		$\{ X_3, X_4 \}$		

We normalized the input feature to have the values in [0, 1] and divided each input feature into ten partitions to calculate mutual information.

As in Table 2, the MIFS selects $\{X_1, X_4\}$ while the MIFS – U select $\{X_1, X_2\}$. Both methods, however, select incorrect features subset. We should select feature subsets $\{X_3, X_4\}$. While the MIFS – U Rough can solve this problem correctly.

4.2 Sonar Target and Ionosphere Datasets

Sonar dataset consists of 208 patterns including 104 training and testing patterns each. It has 60 input features and 2 output classes. For Ionosphere dataset consists of 34 continuous valued attributes with a binary class. This dataset consists of 351 patterns with 50% training and testing pattern each. Those datasets is from UC-Irvine machine learning database [6]. We normalized the input features to have the values in [0, 1] and we divided each input feature space into ten partitions to calculate the entropies and mutual information. We used MLP for training and testing those datasets, with one hidden layer were used and the hidden layer had three nodes. We trained the network for 300 epochs in all cases as in [2].

For Sonar dataset, we selected 3 ~ 12 features among the 60 features, and trained the MLP with the set of training patterns using these input features. The MLP was used with the momentum of 0.0 and learning rate of 0.2 as in [2].

Table 3. Classification rates with different numbers of features for Sonar Dataset (%)

Number of features	3	6	9	10	12	All(60)
MIFS	68.25(1.5)	69.81(2.2)	68.84(1.7)	68.84(1.6)	70.49(3.7)	
MIFS – U	75.24(2.4)	75.72(0.7)	73.78(1.2)	75.24(1.6)	74.85(1.4)	73.59 (2.1)
MIFS – U Rough	75.24(2.2)	76.69(1.3)	76.69(1.3)	78.93(2.1)	82.43(2.1)	
# features Selected	3	5	5	5	4	

Table 4. Classification rates with different numbers of features for Ionosphere Dataset (%)

Number of features	3	5	10	15	20	All(33)
MIFS	71.55(4.2)	71.46(1.8)	66.41(5.3)	72.82(1.7)	77.37(3.0)	
MIFS – U	86.69(2.0)	87.20(1.9)	88.31(1.4)	88.74(1.0)	87.66(2.0)	87.26 (1.5)
MIFS – U Rough	87.31(0.3)	87.60(1.4)	86.06(1.4)	88.80(1.2)	88.34(1.8)	
# features Selected	3	4	5	5	5	

In Table 3, we compared the performance of the MIFS – U Rough with MIFS and MIFS – U. The classification rates are average values of 10 experiments and the number in the parentheses is showed standard deviations that corresponding. For each column in table, we show an appropriate number of features which are filtered by the MIFS – U. We can see that MIFS – U Rough produce better performances than others and also with full features selection (60), while MIFS-U used only 4 features.

For Ionosphere dataset, we used MLP with momentum of 0.1 and learning rate of 1.0 as in [2]. We selected 3 to 20 features from 33 features. From Table 4, we can see that our proposed method produce best performance when select only 5 features from 33 features, while MIFS-U select 15 features.

4.3 Other UCI Datasets

To investigate the performance of our proposed method, we conducted another experiments on four datasets available from UCI [6]. The Wdbc dataset consists of 569 instances, 30 features with 2 classes. The Wine datasets consists of 178 instances, 14 features with 3 classes. The Dermatology dataset (Derm.) consists of 366 instances, 35 features with 6 classes. Lastly, the Letter dataset consists of 20,000 instances, 17 features with 26 classes.

Table 5. Classification rates with different numbers of features for Wdbc and WineDatasets(%)

Wdbc	Number of features	3	5	7	10	All(30)
	MIFS	9507(1.2)	95.42(1.0)	95.60(0.3)	96.27(0.5)	97.61
	MIFS – U	94.05(0.2)	96.66(0.4)	96.48(0.3)	96.58(0.2)	(0.6)
	MIFS – U Rough	-	96.58(0.3)	96.79(0.4)	96.73(0.5)	
	# features Selected	3	4	5	6	
Wine	Number of features	2	3	5	7	All(13)
	MIFS	86.82(3.0)	92.16(1.2)	91.93(0.8)	96.59(1.7)	98.64
	MIFS – U	89.89(1.5)	95.68(0.7)	95.91(1.0)	96.48(0.8)	(0.4)
	MIFS – U Rough	-	95.22(0.7)	95.83(0.7)	97.27(0.6)	
	# features Selected	2	3	4	4	

Table 6. Classification rates for Dermatology and Letter Datasets

Derm.	Number of features	3	5	6	8	12	All(34)
	MIFS	77.59	83.01	92.62	96.62	93.17	
	MIFS – U	77.59	82.79	94.26	94.26	94.26	
	MIFS – U Rough	77.59	78.41	94.26	94.26	94.54	93.38
	# features Selected	3	3	6	7	7	
Letter	Number of features	2	4	6	8		All(16)
	MIFS	36.23	64.56	71.61	73.76		
	MIFS – U	36.23	59.95	75.09	76.77		
	MIFS – U Rough	36.23	59.95	75.09	77.37		77.73
	# features Selected	2	4	6	7		

The comparison on feature subset selection and accuracy rate are shown in Table 5 and 6. All results are average over 10 times (train 50%, test 50%). We used MLP as classification tool for "Wdbc" and "Wine" datasets and train for 300 epochs. The momentum and learning rate of MLP are 0.3 and 0.1, respectively, for Wdbc dataset. For Wine dataset, the momentum and learning rate are 0.1 and 0.2 respectively. For "Dermatology" and "Letter" datasets, we use the decision tree classifier C4.5 as classification tool and all result are average over the 10 – fold cross validation. In most experiments, we can see the proposed method perform better than the MIFS and MIFS – U.

5 Conclusion

In this paper, we illustrated the limitations of the MIFS and the MIFS – U when applied to the feature selection. Although the MI are robustness to noise and data transform ,but the MIFS and the MIFS – U can not handling when two or more combined are dominant features. This problem is difficult to avoid as long as only mutual information is used. For highly nonlinear problem, we can solve by used the quadratic mutual information, but this method have highly computational effort. Then, we proposed MIFS – U Rough for overcoming these problems.

We tested our proposed method on two simple problems and several real-world datasets. It is shown that our proposed method can estimate the appropriate feature subsets and obtain better or equivalent performance compare with MIFS and MIFS-U.

References

1. Battiti, R.: Using mutual information for selecting features in supervised neural net learning. IEEE Trans. Neural Networks 5 (July 1994)
2. Kwak, N., Choi, C.-H.: Input Feature Selection for Classification Problems. IEEE Trans. Neural Networks 13(1), 143–159 (2002)
3. Al-ani, A., Deriche, M.: An optimal feature selection technique using the concept of mutual information. In: Proc. 6th ISSPA, Kuala Lumpar, Malaysia, pp. 477–480 (2001)
4. Li, W.T.: Mutual information functions versus correlation functions. Stat. Phys. 60(5/6), 823–837 (1990)
5. Principe, J.C., Xu, D., Fisher III, J.: Information theoretic learning. In: Haykin, S. (ed.) Unsupervised Adaptive Filtering, pp. 265–320. Wiley, New York (2000)
6. Murphy, P.M., Aha, D.W.: UCI Repository of Machine Learning Databases (1994)
7. Taguchi, G.: Taguchi on Robust Technology Development. Amer. Soc. Mech. Eng., New York (1993)
8. Pawlaw, Z.: Rough sets: Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishing, Dordrecht (1991)
9. Fraser, A.M., Swinney, H.L.: Independent coordinates for strange attractors from mutual information. Phys. Rev. A 33(2) (1986)
10. Katzberg, J., Ziarko, W.: Variable precision rough sets with asymmetric bounds. In: Proc. of the RSKD 1993, Banff, Canada, October 12–15, pp. 163–191 (1993)

Speeding Up Similarity Search on a Large Time Series Dataset under Time Warping Distance

Pongsakorn Ruengronghirunya, Vit Niennattrakul, and Chotirat Ann Ratanamahatana

Department of Computer Engineering, Chulalongkorn University,
254 Phyathai Road, Patumwan, Bangkok Thailand. 10330
{g51prn,g49vnn,ann}@cp.eng.chula.ac.th

Abstract. Time series data are a ubiquitous data type appearing in many domains such as statistics, finance, multimedia, etc. Similarity search and measurement on time series data are typically different from on other data types since time series data have the associations among adjacent dimensions. Accordingly, the classic Euclidean distance metric is not an accurate similarity measure for time series. Therefore, Dynamic Time Warping (DTW) has become a better choice for similarity measurement on time series in various applications regardless of its high computational cost. To speed up the calculation, many research works attempt to speed up DTW calculation using indexing method, which always has a tradeoff between indexing efficiency and I/O cost. In this paper, we propose a novel method to balance this tradeoff under indexed sequential access using Sequentially Indexed Structure (SIS), an approach to time series indexing with low computational cost and small overheads on I/O. Finally, we conduct experiments to demonstrate our superiority in speed performance over the best existing method.

Keywords: Similarity Search, Dynamic Time Warping, Time Series Data, Indexing, Lower Bounding Function.

1 Introduction

Behind the scenes of many innovative applications, the information from these applications sometimes appears in a complex format such as multimedia data. To extract knowledge from this complex information, data are usually represented as time series for the ease of data mining tasks since time series are just a sequence of real number or, in a different aspect, a multi-dimensional variable.

Comparing with other types of multi-dimensional data, time series data have their unique property that they have shifting in time domain, in another aspect, each value in a dimension can be related to another value from different dimensions. An approach for the distance measurement in general multi-dimensional data is to directly calculate the distance from each pair of values on the same dimension, well-known as Euclidean distance metric. A more flexible distance measurement is Dynamic Time Warping (DTW) distance measure [1]; DTW distance calculation allows shifting in time domain which can possibly match related data points from

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 981–988, 2009.
© Springer-Verlag Berlin Heidelberg 2009

contiguous dimensions. Therefore, similarity search on time series data with time shifting using DTW distance for the similarity measurement gives more accurate result than using the Euclidean distance. However, DTW distance has a major drawback on speed performance. The time complexity of the calculation for DTW distance is $O(n^2)$, where n is a total number of the dimensions or length of time series data, which is too far to be acceptable in practice. Therefore, many research works attempt to speed up DTW distance calculation.

A popular approach to speed up DTW calculation is to further compute the lower bound of DTW distance. The lower bounding function, which can approximate the DTW distance with a relatively small computational cost, helps the similarity search method ignore some of the DTW calculations. Many research works [2-4] have proposed various lower bounding functions. For example, Keogh and Ratanamahatana [3] proposed a lower bounding function using the constraint on time warping. This function uses Euclidean distance in calculation which requires only $O(n)$ time complexity. Even though lower bounding techniques can speed up the distance calculation, the distance calculation is just a small part of the similarity search. Some other tasks still need to be considered.

Indexing time series data seems to be cumbersome since each time series consists of a large number of dimensions. This leads to a great complexity for many tree-based index structures such as R*-Tree [5]. Furthermore, the tree-based structure has a major drawback that the order of the tree traversing is not sequential. If the search scans through the tree and retrieves indexed data from a leaf node, this can lead to large overheads on I/O to retrieve a data sequence using random access on the disk. However, an indexed sequential access can solve these problems that incur in tree-based index structure.

In this work, we propose a new similarity search method on time series data using **S**equentially **I**ndexed **S**tructure (SIS), an index structure under indexed sequential access which can directly index to the appropriate part of the dataset and retrieve the indexed data using sequential access on disk. Furthermore, SIS can also ignore most parts of the dataset which are pruned off by our new lower bounding function, LBG, for the search. Since SIS is under indexed sequential access, this leads to only small overheads on disk access. In addition, SIS is generalized to be used with any sequential search methods such as FTW (Fast Search Method for Dynamic Time Warping) [6], the fastest existing similarity search method on time series data under DTW distance. As a result, our method can significantly speed up the similarity search compared with the original FTW.

The rest of the paper is organized as follows. In Section 2, we describe background and related research work. We describe our proposed method, SIS, in Section 3, and in Section 4, we demonstrate the performance of our method in terms of pruning power and time consumption. Finally, Section 5 concludes our work with some suggestion for further research.

2 Background and Related Work

There are several similarity measurements for time series data such as Euclidean distance measure, Dynamic Time Warping (DTW) distance measure [1], longest

common subsequence (LCSS), etc. However, no conclusion has been drawn for the best similarity measurement since the accuracy for each similarity measurement depends on the type of data. For example, Query-by-Humming system [4] applies DTW distance for the similarity measurement because the great merit for DTW distance is that it is flexible and accurate for data that exhibit some shifting in the time axis which is a typical characteristic of the humming voice. However, DTW has a large time complexity of $O(n^2)$. Therefore, many researchers focus on speeding up the DTW calculation.

Since 1993, Agrawal et al. [7] has proposed an approach to speedup DTW on similarity search. They proposed an index structure using R*-Tree [5]. Although this method does not perform well because of the curse of dimensionality problem [8], it has been the first inspiration for many researchers to focus on indexing time series data.

Keogh and Ratanamahatana [3] proposed an improved index structure using R*-Tree [5] and a new lower bounding function for DTW distance using Piecewise Aggregate Approximation (PAA) [9]. Moreover, Zhu and Shasha [4] developed the lower bounding function using PAA which is tighter than Keogh and Ratanamahatana's method. In other words, Zhu and Shasha's lower bounding function gives results which are closer to the real DTW distance. Although this index structure can significantly reduce the number of data needed to be searched, each data retrieval causes large overheads on disk from random access on each node in tree-based index structure.

Sakurai et al. [6] proposed a new approach to speed up DTW using sequential access on disk. They proposed an efficient lower bounding function which can prune off most of the candidate sequences before DTW distance. Their experiments show that their method is much faster than Zhu and Shasha's calculation [4]. However, if an efficient index structure can be adapted to use with their lower bounding function, FTW will absolutely be able to speed up the search since it is obvious that there is no reason to retrieve all data in the dataset for the search.

Before moving on to explain our proposed work, we first provide background knowledge necessary for understanding our methodology.

2.1 Dynamic Time Warping Distance Measure

Dynamic Time Warping (DTW) distance measure [1] is a flexible and accurate distance measure for time series data. Since time series data is high-dimensional data with relationships among other dimensions, Euclidean distance measure, a straightforward distance measure using comparison of data from each dimension independently, is not quite accurate. As a result, DTW distance is developed for time series' domain. DTW distance measure allows comparisons among values from adjacent dimensions. Consider DTW distance calculation between two time series data. Let C be time series of length N, where $C = \{c_1, c_2, \ldots, c_i, \ldots, c_N\}$, and Q be time series of length M, where $Q = \{q_1, q_2, \ldots, q_j, \ldots, q_M\}$. DTW distance $DTW(C,Q)$ is calculated according to eq. (1) below.

$$DTW(C,Q) = \sqrt[p]{f(N,M)}$$
$$f(i,j) = d(c_i,q_j) + \min\begin{cases} f(i-1,j-1) \\ f(i-1,j) \\ f(i,j-1) \end{cases} \quad (1)$$
$$f(0,0) = 0, f(i,0) = f(0,j) = \infty$$

where $1 \le i \le N$, $1 \le j \le M$ and $d(c_i, q_j)$ is a distance between data points c_i and q_j from time series C and Q, respectively. The distance is calculated by using the following Lp-norm equation (2).

$$d(x, y) = |x - y|^p \quad (2)$$

Since DTW distance allows comparisons among values from different dimensions, DTW calculation sometimes gives an inaccurate result because of undesirable or sometimes inappropriate alignment. Therefore, global constraint is assigned to limit the scope on the comparisons; it does not allow any comparisons of any pairs of values which have large distance in the time domain.

3 Proposed Method

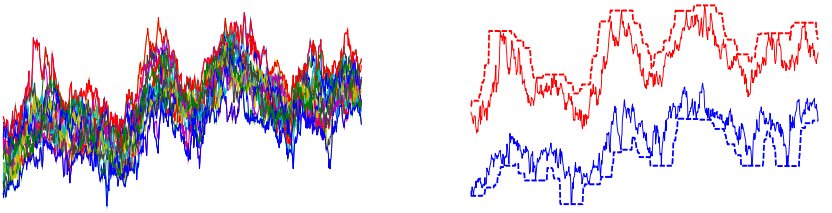
In this section, we introduce a new similarity search method on time series data under **S**equentially **I**ndexed **S**tructure (SIS). In our work, we propose a new lower bounding function on DTW distance for a group of time series data called **L**ower **B**ounding Function for a **G**roup of Time Series (LBG). Furthermore, our new index structure, SIS can significantly speed up the search under DTW distance measure.

SIS consists of two parts which are indexing phase and query phase. In an indexing phase, SIS clusters all data in the dataset into groups and generates a representative for each group. Note that we can complete the indexing phase in advance before the search (offline). In a query process, we use these representatives to calculate the lower bounding distances using our new lower bounding function. Finally, we use these lower bounding distances to guide the search.

3.1 Indexing Phase

Initially, we cluster all the data sequences from a dataset using any existing clustering methods, using Euclidean distance as a distance measure. In our experiments, we use *k*-means clustering [10] in order to minimize the maximum distance between each pair of time series within the same group. Afterwards, we generate an index for each data group. The indexing method is described as follows.

After we obtain groups of data sequences from the clustering method, we assign a bound for each data group. Each bound consists of the upper bound U and the lower bound L which are the maximum and the minimum of the values from each dimension among the data sequences in a group, respectively. Fig. 1a illustrates an example of the bound of a data group. The upper solid profile is the upper bound and the lower solid profile is the lower bound of the group.



(a) **(b)**

Fig. 1. (a) Illustration of a group of time series. (b) Illustration of the profile of the group of time series from (a), shown in solid, and the LB_Keogh's envelope, shown in dash.

Afterwards, we formulate an index of a group of time series data using its bound, as mentioned above, by transforming both upper and lower bounds into an envelope using the algorithm extended from LB_Keogh [3]. In other words, we transform each upper bound and lower bound into LB_Keogh's upper and lower envelope respectively. In Fig. 1, the upper and lower dashes represent the upper envelope and the lower envelope of the group, respectively.

Once we obtain k groups of time series and k envelopes generated for SIS in this phase, we will utilize it in the query phase.

3.2 Lower Bounding Function for a Group of Time Series

Lower **B**ounding Function for **G**roup of Time Series (LBG) is a lower bound of the DTW distance for all time series data in a group. In other words, LBG distance between a query sequence and a group of sequences is a lower bound for the distance between the query sequence and the nearest candidate sequence among sequences in that group. We can simply use the LB_Keogh function [3] to calculate the lower bounding distance. The detail of the calculation is described as follows.

Let Q be a query sequence of length n where $Q = \{q_1, q_2, \ldots, q_i, \ldots, q_n\}$, and let E be the envelope of the group of time series which contains U and L as the upper and the lower envelopes, respectively, where $U = \{u_1, u_2, \ldots, u_i, \ldots, u_n\}$ and $L = \{l_1, l_2, \ldots, l_i, \ldots, l_n\}$. The LBG distance $LBG(Q, E)$ calculation is shown in eq. (3)

$$LBG(Q,E) = \sqrt{\sum_{i=1}^{n} d(q_i, E)}$$
$$d(q_i, E) = \begin{cases} (q_i - u_i)^2 & ; \text{if } q_i > u_i \\ (q_i - l_i)^2 & ; \text{if } q_i < l_i \\ 0 & ; \text{otherwise} \end{cases} \tag{3}$$

3.3 Query Phase

According to the indexing phase from Section 3.1, the dataset has been clustered and a representative (LB_Keogh's upper envelope and lower envelope) is generated from each cluster. In this query phase, we use the LBG distance to indicate the order of the

search. Specifically, the search will sequentially scan in each cluster ordered in an ascending order of the LBG distance. Since LBG is a lower bounding distance, the search can also prune off data sequences from clusters which has larger LBG distance than the best-so-far distance.

4 Experiment

In this section, we demonstrate the efficiency of our proposed method in terms of speed performance with the best existing similarity search method, FTW [6]. Implementations are done in Java, running on an Intel Pentium 4 ® CPU 3.06GHz platform with 1GB of memory. We conduct experiments on one nearest neighbor search using both synthetic datasets and real world datasets. We emphasize on the search over a large dataset which makes FTW perform at its best efficiency. Accordingly, the experiment can demonstrate our superiority.

Eight real-world training datasets from the UCR Time Series Classification /Clustering Page [11], which cover various characteristic of time series data, are used in our experiment. To build a large dataset, each dataset is z-normalized and filled with additional randomwalk sequences of the same length to create a total of 100,000-sequence dataset. The properties of the real-world datasets are shown in Table 1. According to the UCR Time Series Classification/Clustering Page, we assign time warping's global constraints that give the most accurate results. In the indexing phase, we cluster each dataset into 1,000 groups using *k*-mean clustering method. Finally, we evaluate the efficiency of SIS, our proposed index structure, in terms of pruning power and time consumption. We conduct the one-nearest-neighbor search using each real-world test dataset as queries for each generated dataset.

Table 1. The details of training datasets

Name	*Size of training set*	*Size of test set*	*length*	*Global constraint*
Gun-Point	50	150	150	0%
Face (all)	560	1690	131	3%
Trace	100	100	275	3%
Wafer	1000	6174	152	1%
Face (four)	24	88	350	2%
Lightning-2	60	61	637	6%
Lightning-7	70	73	319	5%
Beef	30	30	470	0%

4.1 Pruning Power

Pruning power is a measure which can evaluate the efficiency of any indexing methods [3]. Note that pruning power is a fraction of the candidate sequences which are discarded before the search. Specifically, in this experiment, pruning power is the fraction of the total sequences from all groups which are discarded by SIS method. Fig. 2(a) demonstrates the pruning power of each test dataset. The Lightning2 dataset gives the best pruning power performance for SIS since the shapes of its training data

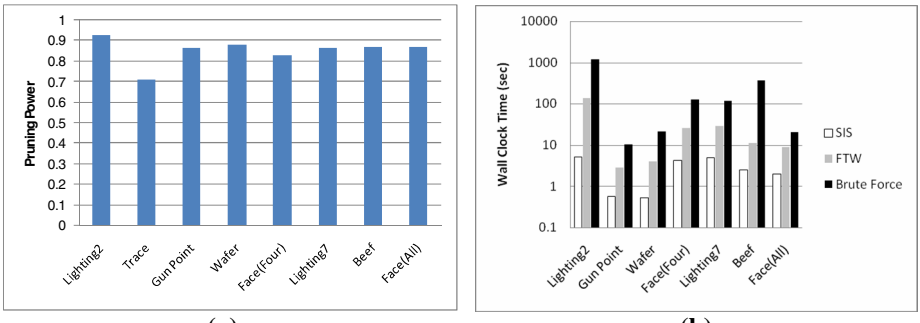


Fig. 2. (a) Pruning power of SIS on our test datasets. (b) Wall clock time, comparing FTW with our proposed SIS on various dataset

are greatly distinguished from random-walk data. Consequently, the clustering method can efficiently distinguish the Lightning2 sequences from the random-walk sequences.

4.2 Time Consumption

We evaluate time consumption on similarity search under SIS by comparing with FTW, the fastest existing similarity search method, and the brute force method, sequential search using the original DTW distance measure. Fig. 2(b) demonstrates the comparison of time consumption between FTW and SIS in terms of average wall clock time per one query sequence. The result shows that our method outperforms FTW in every dataset, particularly in datasets which have high pruning power, by an order of magnitude. Note that both FTW and brute force method use sequential access on disk. As a result, they do not achieve any pruning power since accesses to all candidate sequences are required.

5 Conclusion

We propose a new index structure for similarity search on time series data called SIS, Sequentially Indexed Structure, which can guide the search to a more appropriate part of the database. SIS can also prune off a large number of candidate sequences, which are distinguishable from the query sequence without accessing most of the raw candidate sequences in the database. Moreover, data retrieval applied in our method uses the sequential access on disk. Consequently, this can significantly reduce the I/O cost which can greatly speed up the search. Our experimental results show the superiority of our method, SIS, over the rival method, FTW [6], in terms of speed performance up to 26 times faster. For further works, we will focus on our preprocessing steps, clustering method since the cohesion of the clustered data leads to the efficiency on pruning power of SIS.

Acknowledgement

This work is partially supported by the Thailand Research Fund (Grant No. MRG5080246)

References

1. Ratanamahatana, C.A., Keogh, E.: Three Myths about Dynamic Time Warping. In: Proceedings of SIAM International Conference on Data Mining, Newport Beach, CA, USA, pp. 506–510 (2005)
2. Euachongprasit, W., Ratanamahatana, C.A.: Accurate and Efficient Retrieval of Multimedia Time Series Data under Uniform Scaling and Time Warping. In: Proceedings of The Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, Osaka (2008)
3. Keogh, E., Ratanamahatana, C.A.: Exact Indexing of Dynamic Time Warping. Knowledge and Information Systems (KAIS) 7, 358–386 (2005)
4. Zhu, Y., Shasha, D.: Warping indexes with envelope transforms for query by humming. In: Proceedings of the 2003 ACM SIGMOD international conference on Management of data, pp. 181–192. ACM Press, San Diego (2003)
5. Beckmann, N., Kriegel, H.-P., Schneider, R., Seeger, B.: The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. In: Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, Atlantic City, NJ, USA, pp. 322–331 (1990)
6. Sakurai, Y., Yoshikawa, M., Faloutsos, C.: FTW: Fast Similarity Search under the Time Warping Distance. In: Proceedings of 24th ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, pp. 326–337. ACM Press, Baltimore (2005)
7. Agrawal, R., Faloutsos, C., Swami, A.: Efficient Similarity Search In Sequence Databases. In: Lomet, D.B. (ed.) FODO 1993. LNCS, vol. 730, pp. 69–84. Springer, Heidelberg (1993)
8. Berchtold, S., Bohm, C., Kriegal, H.-P.: The pyramid-technique: towards breaking the curse of dimensionality. In: Proceedings of the 1998 ACM SIGMOD international conference on Management of data. ACM, Seattle (1998)
9. Yi, B.-K., Faloutsos, C.: Fast Time Sequence Indexing for Arbitrary Lp Norms. In: Proceedings of 26th International Conference on Very Large Data Bases, pp. 385–394. Morgan Kaufmann Publishers Inc., Cairo (2000)
10. MacQueen, J.B.: Some Methods for classification and Analysis of Multivariate Observations. In: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
11. Keogh, E., Xi, X., Wei, L., Ratanamahatana, C.A.: The UCR Time Series Classification/Clustering Homepage, http://www.cs.ucr.edu/~eamonn/time_series_data/

A Novel Fractal Representation for Dimensionality Reduction of Large Time Series Data

Poat Sajjipanon and Chotirat Ann Ratanamahatana

Department of Computer Engineering, Chulalongkorn University
254 Phyathai Road, Patumwan, Bangkok Thailand, 10330
{g50psj,ann}@cp.eng.chula.ac.th

Abstract. Recent research has attempted to speed up time series data mining tasks which focus on dimensionality reduction, indexing, and lower bounding function, among many others. For large time series data, current dimensionality reduction techniques cannot reduce the total dimensions of time series data by a large margin without losing their global characteristics. In this paper, we introduce a novel Fractal Representation which uses merely three real values to represent a whole time series data sequence. Moreover, our proposed representation can be efficiently used under Euclidean distance. We demonstrate effectiveness and utility of our novel Fractal Representation on classification problems and our proposed method outperforms existing methods in terms of speed performance and accuracy. Our results reconfirm that this representation can effectively represent global characteristics of the data, especially in larger time series data.

Keywords: Time Series Mining, Fractal Representation, Fractal Dimension.

1 Introduction

The ultimate goal to improve time series mining tasks is both speedup and increasing accuracy. Unfortunately, there is a tradeoff between accuracy and computational cost; for example, Euclidean distance is very fast, but the accuracy may be low when comparing with other similarity measures. On the other hand, Dynamic Time Warping (DTW) distance measure [1] is relatively slow, but it has been widely accepted as one of the best distance measures for time series data, particularly for relative short time series. Although this method consumes a large amount of computation time, time complexity can be reduced by combining DTW distance with other algorithms, especially bounding function [2]. High computational cost for DTW distance is clearly inappropriate for large time series data; therefore, a recent Compression-Based Dissimilarity Measure [3] has been proposed for similarity measure between two time series data based on file compression, which focuses mainly on large time series data. However, CDM is I/O bounded, which affects the computational time.

Alternatively, the speedup has been improved through indexing, dimensionality reduction/data representation, among many others. For large time series data, the

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 989–996, 2009.
© Springer-Verlag Berlin Heidelberg 2009

current dimensionality reduction techniques cannot reduce the dimension by a large margin without losing their global characteristics. Most of these methods are used for pruning off time series data that are not the answer, such as Symbolic Aggregate Approximation (SAX) [4], and clipped data representation [5].

The ideal case for dimensionality reduction is to represent time series data with only one dimension, while preserving original data's characteristics. Once these resulted dimensions are sorted, we can compare two time series using constant time. Consequently, this research work focuses on reducing the number of dimensions of large time series data.

In this work, we introduce a novel representation, Fractal Representation, for large time series data based on fractal dimension, using merely three real values to represent the internal structure of a time series data. In our experiment, Fractal Representation can accomplish a large speedup in a wide range of data mining tasks. In addition, we demonstrate the superiority of our proposed method over the well-known distance measures, i.e., Euclidean distance, DTW distance, and CDM, on classification problems. The results have demonstrated that our representation can accurately classify the data, especially in longer time series, since Fractal Representation can effectively represent global characteristics of the data.

The rest of this paper is organized as follows. In section 2, we review related work on time series mining and dimensionality reduction techniques. Section 3 describes background knowledge of the fractal dimension. After that, our proposed Fractal Representation is introduced in section 4. Section 5 contains our experimental evaluation and discussion, and, finally, section 6 draws some conclusions and gives suggestions for future work.

2 Background and Related Work

Generally, the dimension of any data, such as a point, a line, a plane, a cube, is represented by integer value. This dimension is called topological dimension. However, the dimension, i.e., Fractal dimension [8], can be a real value which considers the local structure of the data by seeking self similarity within. Both fractal theory and fractal dimension are based on chaos theory, which is a non linear and deterministic dynamic system. Fractal theory's key characteristic is the self similarity; the shapes of the data still look similar at any scaling levels. In the other words, a fractal dimension is the dimension of an irregular geometric object with an infinite nesting of structure at all scales. To calculate the fractal dimension, many techniques, e.g., compass dimension, box-counting dimension, information dimension, and correlation dimension, have been proposed. Generally, framework of fractal dimension is commonly calculated iteratively in different levels of local structure, as shown in Fig. 1.

Concept of fractal dimension is not commonly applied for the dimensionality reduction on data mining tasks. More specifically, fractal dimension is typically used for determining the dimension of a picture. Some research applies fractal dimension for data points and time series data, i.e., Barbara and Chen [6] applied fractal dimension for data points classification, and Xiao et al. [7] used fractal dimension to classify EMG signals. However, the results of fractal dimension still cannot distinguish the classes very effectively.

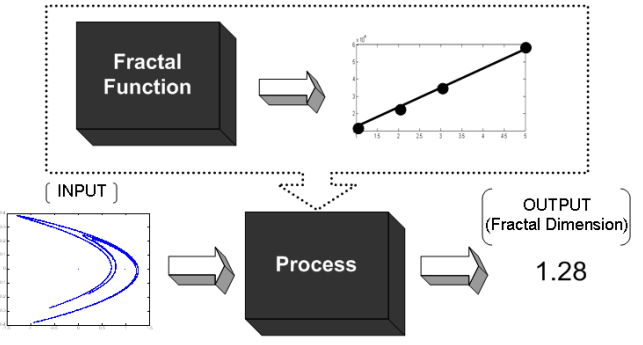


Fig. 1. General framework of fractal dimension calculation

2.1 Compass Dimension

The idea of compass dimension [8] is to estimate the dimension by seeking self similarity using a compass segment or a ruler to travel along the perimeter of input data based on distance. For example, in the first step, the compass segment with length s is assigned, and this compass segment traverses from the starting position to the ending position of the input data. The overall image perimeter length is then estimated from the compass function. Next, a new compass segment's length is assigned as half of the previous length, and the compass function calculation is repeated. Each coordinate of the compass segment and the compass function output is plotted on log-log scale, and the slope of the best-fitted line for these points is the compass dimension (D_c), shown in equation (1).

$$D_C = \lim_{s \to 0} \frac{\ln N(s)}{\ln(1/s)}. \tag{1}$$

Where s is the compass length, and $N(s)$ is the sum of the compass segments' lengths s from any starting position to the ending position.

2.2 Correlation Dimension

Correlation dimension [8, 9], proposed by Grassberger and Procaccia, measures the self-similarity by calculating correlation among all data points in space. Correlation of the data points, also called Correlation Integral, is measured as distances between a fixed point and every other point. When the distance is less than a threshold distance r, it is counted as '1'. More specifically, Correlation Integral is defined as the total number of a pair of data points, which has distance less than the threshold distance r. This correlation integral is estimated by the following equation.

$$C_{CI}(r) = \lim_{N \to \infty} \frac{1}{N} \sum_{i=1}^{N} \frac{1}{N} \sum_{j=1}^{N} \Theta(r - |\vec{x}_i - \vec{x}_j|). \tag{2}$$

where N is the total number of points, $\bar{x}$ is a data point of space, and r is a threshold distance. $\Theta(x)$ is the Heaviside step function, and is defined in equation (3).

$$\Theta(x) = \begin{cases} 0 & \text{when x} \leq 0 \\ 1 & \text{when x} > 0 \end{cases} \tag{3}$$

To calculate the correlation dimension, the initial threshold distance r is first assigned to an arbitrary value. Next, the correlation integral $C_{CI}(r)$ is calculated, and then r is set to half of its previous value. Correlation integral is repeatedly calculated. Finally, each coordinate of correlation integral $C_{CI}(r)$ (y-axis) and scale factor r (x-axis) is plotted on a log-log diagram. The slope of the best-fitted line is the correlation dimension (D_{C2}), defined by an equation below.

$$D_{C2} = \lim_{r \to 0} \frac{\ln C_{CI}(r)}{\ln r}. \tag{4}$$

3 Our Proposed Method

In this section, we introduce a novel Fractal Representation which uses merely three real values to represent each time series sequence efficiently, according to our proposed fractal dimension approaches below.

Although fractal dimension can significantly reduce dimensions of time series, a bad set of parameters does affect the accuracy. Hence, our proposed work also includes automatic parameter tuning, which is demonstrated to achieve higher accuracy than those of existing methods, especially for z-normalized large time series data.

3.1 Equi-width Compass Dimension

Equi-width compass dimension is extended from the original compass dimension to be applied specifically to time series data. We divide a sequence on x-axis into equal widths, called a time slice, and then traverse a line within each time slice. As shown in Fig. 2, a time series length is 200 units, and the time slice is 20 units; therefore, time series is equally divided to 10 periods, and then travel along from period 1 to 10. The compass function is based on y-axis values only, and the time slice in each iteration is assigned according to equation (5).

$$Time_Slice(k) = ceil\left(\frac{\log_2(L)}{2^k}\right) \tag{5}$$

where $Time_Slice(k)$ is a period assigned in each iteration k, k is the number of the current iteration, and L is length of time series data.

From equation (5), in iteration 1 ($k = 1$), the time slice is initialized by $\log_2$ of time series length because large time slice leads to an inaccurate approximation. The time slice is divided into half, until the time slice equals 1; then it terminates. Finally, values of compass function and time slice are then plotted on graph, and then the equi-width compass dimension is determined by the slope of the best-fitted line of the points in the log-log scale, shown in equation (1).

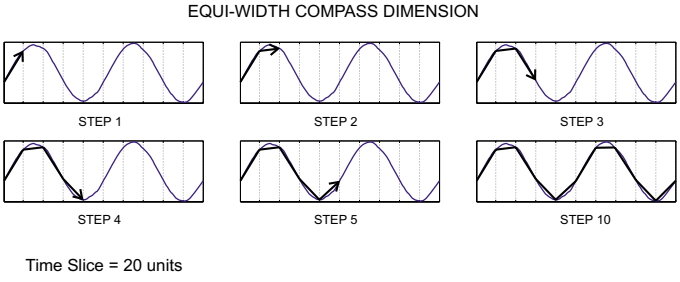


Fig. 2. An example of equi-width compass dimension with a time slice of 20 units

3.2 Equi-length Compass Dimension

Equi-length compass dimension resembles the original compass dimension. A line with a scaling factor s is the compass segment which uses a fixed length to travel around time series. Total summation of all the compass segment length is a compass function value of each iteration, as shown in Fig. 3. The length of compass segment for each iteration $s(k)$ of equi-length compass dimension is shown in equation (6).

$$s(k) = \left| \frac{\max(ts) - \min(ts)}{2^k} \right| \tag{6}$$

where $s(k)$ is the length of compass segment for each iteration k, max(ts) is a maximum value of time series data, and min(ts) is a minimum value of time series data.

From equation (6), we use the difference between max(ts) and min(ts) value to initialize the compass segment. For each iteration, $s(k)$ is divided into half of previous length. When every data point of the time slice are traversed in any particular $s(k)$, it terminates. We obtain equi-length compass dimension from the slope of the best-fitted line of the points in the log-log scale, as shown in equation (1).

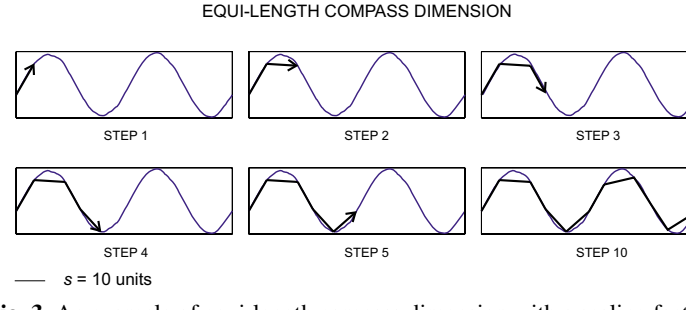


Fig. 3. An example of equi-length compass dimension with a scaling factor s

3.3 Correlation Dimension

Correlation dimension for time series data resembles the original method. We use correlation integral, as in equation (2), to calculate distance on a y-axis between a

fixed point and every other points of time series data. When distance is less than threshold distance *r*, correlation integral accumulates number of correlation in each point, and then the threshold distance *r* is reduced to half. Until correlation integral is zero or invariant values, it terminates. Finally, the log-log plot between *r*(*k*) values and the correlation integrals is obtained, and correlation dimension is determined by the slope of the best-fitted line. The threshold distance *r* for each iteration *r*(*k*) of correlation dimension is shown in equation (7).

$$r(k) = \left| \frac{\max(ts) - \min(ts)}{2^k} \right| \tag{7}$$

where *r*(*k*) is a threshold distance of each iteration *k*, max(*ts*) is maximum value of a time series data, and min(*ts*) is minimum value of a time series data.

Finally, our proposed method are three real values to represent a time series called Fractal Representation, which also give a new identity to the time series data, with a massive reduction. Moreover, our proposed method is automatic parameter.

4 Experiment

In this section, we demonstrate efficiency of our Fractal Representation on classification problem in terms of accuracy and computation time. We compare our proposed method with Euclidean Distance, DTW distance, and CDM.

4.1 Datasets

In our experiment, we build our datasets by generating time series data from the benchmark datasets of the UCR Time Series Data Mining Archive [10]. We test on both long and short time series data. Note that all datasets are z-normalized. A total of eight datasets are described in Table 1.

Table 1. Detail of each dataset of time series in our experiment

Datasets	Number of Data	Data Points	Number of Class	Typical Data
Dataset 1	36	1,000	18	Evaporator, furnace, balloon, reel, dryer, etc.
Dataset 2	70	1,000	7	Aspect ratio, cement, field, hooked, etc.
Dataset 3	80	2,000	8	Power demand, standing, wall, EEG, etc.
Dataset 4	65	1,000	13	ERP, koski ecg, kung fu, 4 class of ECG etc.
Dataset 5	44	2,000	11	Power data, nprs, stemgen, random walk, etc.
Dataset 6	130	3,000	13	CinC, symbols, faces, random walk, words, etc.
Dataset 7	70	128	7	Adiac, CBF, face-all, wafer, etc.
Dataset 8	60	420	6	Beef, fish, lighting-2, Olive-oil, OSU leaf, etc.

4.2 Experiment

We evaluate our proposed method, Fractal Representation, on classification problem, using 1-nearest-neighbor classifier with leaving-one-out technique in terms of both

Table 2. Classification results for comparing Fractal Representation with existing methods

Datasets	Euclidean Distance		DTW		CDM		Fractal Representation	
	Accuracy (%)	Time (sec)	Accuracy (%)	Time (sec)	Accuracy (%)	Time (sec)	Accuracy (%)	Time (sec)
Dataset 1	30.56	0.182	63.89	140.5	33.33	46.928	**75**	**0.02**
Dataset 2	75.72	0.46	88.57	685.71	55.72	156.12	**95.72**	**0.042**
Dataset 3	83.75	1.48	**90**	3742.29	81.25	318.18	88.75	**0.041**
Dataset 4	21.54	0.38	**83.08**	570.46	46.15	132.25	73.85	**0.055**
Dataset 5	31.82	0.39	70.45	1052.1	65.91	95.75	**81.82**	**0.032**
Dataset 6	66.15	3.85	80	18766.1	37.69	1111.7	**80.77**	**0.979**
Dataset 7	85.71	0.13	**100**	14.087	14.28	96.05	84.29	**0.0636**
Dataset 8	88.33	0.189	**96.67**	86.30	26.66	88.81	75	**0.0659**

accuracy and running time, and then compare our algorithm with Euclidean distance, DTW distance, and CDM. Results from 8 datasets are shown in Table 2.

From Table 2, Fractal Representation outperforms other methods in all datasets in terms of the running time. In some cases, we could achieve as much as 4 orders of speedup magnitude in wall clock time, especially when comparing with DTW. For large time series (datasets 1-6), our Fractal Representation performs well in many cases in terms of the accuracy. In dataset 3, we get the accuracy less than DTW distance, but our running time is almost 90,000 times faster. For dataset 4, our accuracy is not high, comparing to other datasets because this dataset consists of very similar ECG signals (4 classes); therefore, Fractal Representation may not differentiate those global structures very well. In general, DTW outperforms our Fractal Representation when dealing with short time series data. However, the results of Fractal Representation on both accuracy and time of short time series perform well in all cases. In addition, CDM does not seem to work very well here since CDM generally works well with unnormalized data, whereas every dataset here is z-normalized. Amplitude of each time series therefore is highly similar, and its compressions have similar file size which directly affects and degrades the classification results.

5 Conclusion

Our Fractal Representation is a novel dimensionality reduction technique that does not need extra pruning of unnecessary data, providing great speedup for the mining tasks because of the large reduction in the number of dimensions down to only three real values. Our main advantage is that this representation can be done offline. The Fractal Representation has demonstrated to be both effective and efficient for time series data under automatic parameter selection. For our future work, if we could develop representation which uses fewer values, ideally one. It will provide an ideal speedup to any data mining tasks.

Acknowledgement

This work is partially supported by the Thailand Research Fund (Grant No. MRG5080246).

References

1. Ratanamahatana, C.A., Keogh, E.: Three Myths about Dynamic Time Warping. In: Proceedings of SIAM International Conference on Data Mining (SDM 2005), pp. 506–510 (2005)
2. Keogh, E., Ratanamahatana, C.A.: Exact Indexing of Dynamic Time Warping. In: Knowledge and Information Systems (KAIS), pp. 358–386 (2005)
3. Keogh, E., Lonardi, S., Ratanamahatana, C.A.: Towards Parameter-Free Data Mining. In: Proceedings of 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2004)
4. Lin, J., Keogh, E., Lonardi, S., Chiu, B.: A Symbolic Representation of Time Series, with Implications for Streaming Algorithms. In: Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, pp. 460-469 (2003)
5. Bagnall, A., Janacek, G.: Clustering Time Series with Clipped Data. Machine Learning Journal, 151–178 (2005)
6. Barbará, D., Chen, P.: Using the Fractal Dimension to Cluster Datasets. In: Proceedings of 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, pp. 260–264 (August 2000)
7. Xiao, H., Zhi-Zhong, W., Xiao-mei, R.: Classification of surface EMG signal with fractal dimension. Journal of Zhejiang University SCIENCE (JZUS) (May 30, 2005)
8. Peitgen, H.O., Jurgens, H., Saupe, D.: Chaos and Fractals New Frontiers of Science (2003)
9. Grassberger, P., Procaccia, I.: Measuring the strangeness of strange attractors. Physica D
10. Keogh, E., Xi, X., Wei, L., Ratanamahatana, C.A.: The UCR Time Series Data Mining Archive, University of California, Computer Science & Engineering Department, Riverside CA (2006), http://www.cs.ucr.edu/~eamonn/TSDMA/datasets.html

Clustering Data Streams in Optimization and Geography Domains

Ling-Yin Wei and Wen-Chih Peng

Institute of Computer Science and Engineering
National Chiao Tung University
Hsinchu, Taiwan, ROC
{lywei.edu, wcpeng}@gmail.com

Abstract. In this paper, we formulate a dual clustering problem in spatial data streams. A spatial data stream consists of data points with attributes in the optimization and geography domains. We aim at partitioning these objects into disjoint clusters such that at each time window (1) objects in the same cluster satisfy the transitively r-connected relation in the optimization and geography domains, and (2) the number of clusters is as minimal as possible. We propose a Hierarchical-Based Clustering algorithm (HBC). Specifically, objects are represented as a graph structure, called RGraph, where each node represents an object and edges indicate their similarity relationships. In light of RGraph, algorithm HBC iteratively merges clusters. Experimental results show the performance of the algorithm.

1 Introduction

With the growing of sensor networks, a huge amount of sensor data are available. Without loss of generality, sensor data points usually have attributes in two domains (i.e., optimization domain and geography domain). Attributes in the optimization domain (respectively, the geography domain) are referred to as nongeometric (respectively, geometric) attributes. Sensing measurements are nongeometric attributes and the corresponding locations of sensors are geometric attributes. As such, a dual clustering problem is formulated as clustering objects over the optimization and geography domains. Explicitly, the goal of a dual clustering over the optimization domain and geography domains is that data points with similar nongeometric attributes are clustered and the corresponding geometric attributes are satisfied a spatial constraint. Some related works have been elaborated on the dual clustering problem and the problem of clustering sensor data [1,2,3,4,5,6,7].

We mention in passing that the authors in [1,6,7] proposed a dual clustering problem over the optimization and geography domains. A dual clustering problem over the geography and optimization domains is that given a set of objects whose attributes are in both domains, objects would be partitioned into disjoint clusters such that objects in the same group are connective in the geography domain as the clustering cost in the optimization domain is minimized. Moreover,

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 997–1005, 2009.
© Springer-Verlag Berlin Heidelberg 2009

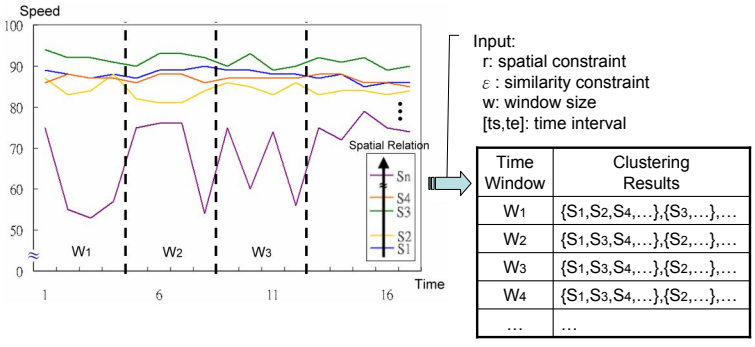


Fig. 1. An example of clustering results in highway traffic data streams

the authors in [1,6] proposed more efficient algorithms for a dual clustering problem. Explicitly, a compact region, in which data points have similar attributes in the optimization domain and their locations are connected, is developed. However, the constraint in the geography domain is not clearly specified. Thus, in [7], a dual clustering problem with a generalized connected constraint for spatial databases is proposed. The problem is that given the number of clusters and a distance threshold, objects whose distance in the geography domain is within a given threshold are grouped into a cluster and the average cost of clustering results in the optimization domain is minimized. In [2], given a set of data streams, the goal is to cluster objects with higher similarity in each non-overlapping window. However, the work in [2] does not consider the spatial constraint among objects. Without a specified number of clusters, the authors in [3] partition objects into a set of clusters in which objects have similar phenomena. Although considering both domains, the authors in [3] focus on the communication cost for clustering sensors in a distributed manner, let alone considering the dual clustering problem in a data stream environment.

Consider an example of traffic databases in Figure 1, where each object S_i represents a sensor and detects a series of speeds over time and some examples of cluster results at each time window are derived. Generally speaking, nongeometric attributes (i.e., the speeds) of sensor data consist of a series of readings. Such data generated from a set of sensors could be viewed as spatial data streams. Given a set of sensor data with their nongeometric attributes and geometric attributes, for each time window, we aim at clustering sensors into groups, where sensors in the same group have similar nongeometric readings and their locations satisfy the spatial constraint.

In data streams, data points are usually evolved with time. The general dual clustering problem in [7] is not feasible in that the number of clusters is usually varied with time and is hard to be pre-specified. Furthermore, minimizing the average cost of clusters at each sliding time is not necessary due to that the dynamic features of data streams. Thus, in this paper, the dual clustering problem in spatial data streams is formulated. Specifically, we intend to partition these

objects into disjoint clusters such that at each time window (1) objects in the same cluster satisfy the transitively r-connected relation in the optimization and geography domains, and (2) the number of clusters is as minimal as possible. In this paper, objects are represented as a graph structure, called RGraph, where each node represents an object and edges indicate their similarity relationships. In light of RGraph, we first propose a Hierarchical-Based Clustering algorithm (HBC) to cluster data objects at each time window.

The rest of the paper is organized as follows. In Section 2, we formally define the dual clustering problem in a data stream environment. We propose an algorithm in Section 3. The performance of our proposed algorithm is shown in Section 4. Section 5 concludes with this paper.

2 Preliminaries

A set of spatial data streams is denoted as $STD = \{S_1, S_2, ..., S_n\}$, where each object S_i has attributes in the optimization domain and the geography domain. Furthermore, a data stream S_i is represented as $\{L_i, V_i, [t_0, t_{now}]\}$. Clearly, the attribute in the geography domain is the location of S_i, denoted as L_i, and the attribute in the optimization domain is represented as a vector $V_i = (v_i^{t_1}, v_i^{t_2}, \ldots, v_i^{t_m})$, where $v_i^{t_j}$ is the value of S_i at the time stamp t_j and $t_0 \leq t_1 < \ldots < t_m \leq t_{now}$.

In this paper, given a set of data streams, a spatial constraint r (used in the geography domain), an error ε (for the optimization domain), a window size w, for each time window with its size as w, we aim at partitioning the input data streams into several groups such that (1) objects in the same cluster satisfy the transitively r-connected relation in the optimization and geography domains, and (2) the number of clusters is as minimal as possible.

In the following, we will clearly define the dissimilarity in the optimization domain and the spatial constraint in the geography domain. Among a variety of distance functions, we adopt the average Euclidean distance, which is the most common employed, as the dissimilarity measurement in the optimization domain. As such, we have the following definition:

Definition 1. *(Dissimilarity) Given a time interval $[t_p, t_q]$ and two objects S_i and S_j, the values of objects S_i and S_j in $[t_p, t_q]$ are $V_i = (v_i^{t_1}, v_i^{t_2}, \ldots, v_i^{t_m})$ and $V_j = (v_j^{t_1}, v_j^{t_2}, \ldots, v_j^{t_m})$ respectively, where $t_p \leq t_1 < \ldots < t_m \leq t_q$. The dissimilarity between S_i and S_j in this time interval is defined as*

$$diss(S_i, S_j)|_{[t_p,t_q]} = \frac{1}{m} \times \sqrt{\sum_{t=t_1}^{t_m} (v_i^t - v_j^t)^2}.$$

With the above definition, if $diss(S_i, S_j)|_{[t_p,t_q]} \leq \varepsilon$, S_i and S_j are *similar* in a time interval $[t_p, t_q]$. Then, the r-connected relation between objects is thus defined as follows:

Definition 2. *(r-connected relation between two objects) Given a spatial constraint r and an error ε, two sensors S_i and S_j are said to be r-connected*

in a time interval $[t_p, t_q]$ *if* S_i *and* S_j *are similar and* $ED(S_i, S_j) \leq r$*, where* $ED(S_i, S_j)$ *is the Euclidean distance between objects* S_i *and* S_j *in the geography domain.*

Notice that r-connected relation indicates that two objects satisfy both the spatial constraint in the geography domain and the error threshold in the optimization domain. Now, given a set of objects with their r-connected relations, we could further define the transitively r-connected relation between two objects as follows:

Definition 3. *(Transitively r-connected relation between two objects) Given a spatial constraint* r *and an error* ε*, two objects* S_i *and* S_j *are said to be transtively r-connected in a time interval* $[t_p, t_q]$ *if* $ED(S_i, S_j) > r$*, and there exists a chain of objects* $S_i, S_{k_1}, S_{k_2}, \ldots S_{k_l}, S_j$ *such that (1)* $S_{i'}$ *and* $S_{j'}$ *are similar for any two objects,* $S_{i'}$ *and* $S_{j'}$*, in this chain, (2)* S_{k_h} *and* $S_{k_{h+1}}$ *are r-connected for* $k_1 \leq k_h < k_l$*, (3)* S_i *and* S_{k_1} *are r-connected, and (4)* S_{k_l} *and* S_j *are r-connected.*

As pointed out early, the goal of our dual clustering problem is to partition objects into several clusters at each time window. Clearly, clustering results of the dual clustering problem in a data stream environment should comply with r-connected relations or transitively r-connected relations. Hereafter, the cluster results are referred to as transitively r-connected clusters.

3 Algorithms of Dual Clustering Data Streams

In this section, a graph representation for data objects is described. Then, we propose an algorithm to derive transitively r-connected clusters at each time window.

3.1 Graph Representation: RGraph

For each time window (e.g., w_i), data objects and their relationships are represented as $G_{w_i} = (V, E)$, where each node is referred to one data object and there exist edges if these data objects are similar for capturing the r-connected or transitively r-connected relations. Hence, we have the following two definitions:

Table 1. Example of a highway traffic dataset

SID	Location(km)	Speed(km/hr)	Time
S_1	25.1	(91,89,91,89,89,88,82,85,87,90)	[1,11)
S_2	27.1	(95,93,91,94,92,94,91,90,92,95)	[1,11)
S_3	49.1	(93,94,94,91,93,89,95,91,91,90)	[1,11)
S_4	55.1	(93,96,97,96,96,97,98,98,94,97)	[1,11)
S_5	57.0	(91,90,91,93,90,91,89,91,90,91)	[1,11)
S_6	62.4	(89,92,95,92,91,96,93,96,88,87)	[1,11)
S_7	65.0	(87,83,85,88,87,87,85,87,87,82)	[1,11)
S_8	67.3	(83,83,83,83,82,84,85,83,80,82)	[1,11)
S_9	71.4	(93,91,92,92,93,88,93,93,89,90)	[1,11)

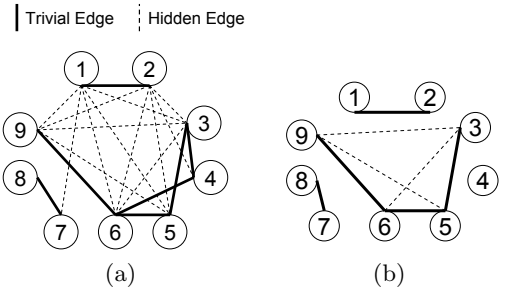


Fig. 2. (a) Graph representation. (b) An example of clustering results.

Definition 4. *(Trivial edge) If two nodes S_i and S_j are r-connected, there exists a trivial edge between S_i and S_j, denoted as $e_t(S_i, S_j)$.*

Definition 5. *(Hidden edge) If two nodes S_i and S_j are similar and $ED(S_i, S_j) > r$, there exists a hidden edge between S_i and S_j, denoted as $e_h(S_i, S_j)$.*

Such a graph is called RGraph. For example, given the r=10 km, $\varepsilon = 5$ km/hr, w=10, and the set of data streams in Table 1, at the first time window (i.e.,

Algorithm 1. Hierarchical-Based Clustering (HBC) Algorithm

input : A set of data streams STD, an error threshold ε, a spatial constraint r, and the number of time slots of each window w
output : The transitively r-connected clusters R_{w_k} with respect to the window w_k
variable: R'_{w_k}: a temporal clustering result in the window w_k

```
1  for each window w_k do
2      for each two sensors S_i, S_j in STD do
3          if S_i != S_j then
4              /*Graph Generation Phase*/
5              Compute diss(S_i, S_j) and ED(S_i, S_j);
6              if diss(S_i, S_j) ≤ ε then
7                  if ED(S_i, S_j) ≤ r then
8                      Generate a trivial edge e_t(S_i, S_j);
9                  else Generate a hidden edge e_h(S_i, S_j);
           R_{w_k} = ⋃_{∀S_h ∈ STD} {S_h};
10         while |R_{w_k}| > 1 do
11             Using the hierarchical-based clustering approach to generate R'_{w_k};
12             if ||R'_{w_k}|| < ||R_{w_k}|| then
13                 R_{w_k} = R'_{w_k};
14             else break;
15 return R_{w_k};
```

w_1), RGraph in Figure 2(a) is built according to their nongeometric and geometric attributes. The solid line (respectively, the dotted line) is a trivial edge (respectively, a hidden edge). For instance, S_1 and S_2 are r-connected since $diss(S_1, S_2)|_{[1,10]} = 1.6 \leq 5$ and $ED(S_1, S_2) = 2 \leq 10$. Then, a trivial edge $e_t(S_1, S_2)$ exists between S_1 and S_2. S_1 and S_9 are similar but far because $diss(S_1, S_9)|_{[1,10]} = 1.5 \leq 5$ and $ED(S_1, S_9) = 46.3 > 10$. Therefore, there is a hidden edge $e_h(S_1, S_9)$ between S_1 and S_9.

Once we have the graph at each time window, the transitively r-connected clusters are subgraphs in which nodes are connected via trivial edges and these subgraphs are complete graph via both trivial edges and hidden edges. For example, in Figure 2(b), there are four clusters at the time window w_1. The subgraphs refer the cluster results and agree with our above claim.

3.2 Hierarchical-Based Clustering Algorithm

We propose a Hierarchical-Based Clustering algorithm (abbreviated as HBC) in this section. For each time window, we will build RGraph in which both the trivial edges and hidden edges are determined according to their attribute values over the optimization and the geography domains. In light of the RGraph, clustering results are generated by iteratively merging other clusers. Explicitly, in the beginning, each node is regarded as a single cluster. For each cluster, those clusters that are reached via trivial edges are candidate clusters for merging. To determine which trivial edge should be selected, the weight of a trivial edge is assigned by the dissimilarity of two nodes. Among trivial edges, we select the one with the minimal dissimilarity for possible merging operations. Note that these two clusters will be merged if these two clusters form a complete graph. Otherwise, we will continue to select the next trivial edge. Following the same process, cluster results are generated. The process of merging will stop if there is no more trivial edges. The detail of clustering processes is shown in algorithm HBC.

Given an example of data streams in Table 1, the RGraph is shown in Figure 2(a). Then, the weights of trivial edges are obtained as shown in Figure 3. Let $M = \{e_t |$ each e_t is an unused trivial edge$\}$. Initially, M is the set of all trivial edges. In the beginning, $e_t(S_3, S_5)$, whose weight is the minimal in M, is selected and removed from M, and S_3 and S_5 are merged into a new cluster, i.e., $\{S_3, S_5\}$. In Figure 3(b), $e_t(S_5, S_6)$, whose weight is the minimal in M, is selected and removed from M. S_6 could be merged into $\{S_3, S_5\}$, because $\{S_3, S_5, S_6\}$ forms a connected subgraph via trivial edges and there exists a hidden edge $e_h(S_3, S_6)$ such that $\{S_3, S_5, S_6\}$ forms a complete subgraph. While $e_t(S_4, S_6)$ is considered, we would check whether S_4 could be merged into $\{S_3, S_5, S_6\}$. Although $\{S_3, S_4, S_5, S_6\}$ are connected, $\{S_3, S_4, S_5, S_6\}$ cannot form a complete subgraph since there is no edge between S_4 and S_5. Therefore, S_4 could not be merged into $\{S_3, S_5, S_6\}$. After repeating similar steps, we could derive a clustering result shown in Figure 3(f).

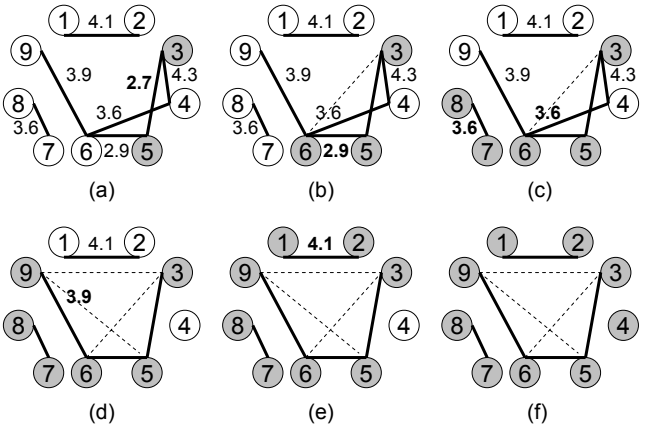


Fig. 3. A running example of algorithm HBC

4 Performance Study

4.1 Dataset and Settings

In this section, we utilize the real dataset to evaluate the performance of our proposed algorithm with respect to different parameters. These parameters are: a number of sensors N, a number of time slots L, a size of each time window W, a number of time windows D, a spatial constraint (in km) r, and a similarity threshold (in km/hr) ε. The freeway traffic database was obtained from Taiwan Area National Freeway Bureau. We evaluate the performance on the real dataset related to the traffic monitor along with No.1 Freeway in Taiwan. There are 158 sensors on No.1 Freeway including both directions and the length of No.1 Freeway is 372.7 kilometers. We accessed speeds per 10 minutes from each sensor. In our experiments, the default setting is that N=79, L=434, r=10, W=10, and ε=5. Experiments were conducted on a computer with 2.80GHz Intel CPU and 2GB of memory, running Microsoft Windows Server 2003.

4.2 Experimental Results

Figure 4 shows the performance of algorithm HBC. In Figure 4(a), as the number of sensors increases, the runtime of algorithm HBC tends to increase. Furthermore, we implemented a brute-force algorithm for the optimal solution, called Opt, and the efficiency and quality of clustering results of algorithms HBC and Opt are compared in Figure 4(b). As can be seen in Figure 4(b), the number of clustering results of algorithm HBC is quite similar to that of algorithm Opt, but the runtime of algorithm HBC is less than that of algorithm Opt. The impact of varying both the spatial constraint and the error is investigated in Figure 5. In Figure 5(a), with a larger setting in r, the runtime of algorithm HBC increases. The reason is that with a larger number of r, more sensors are considered for

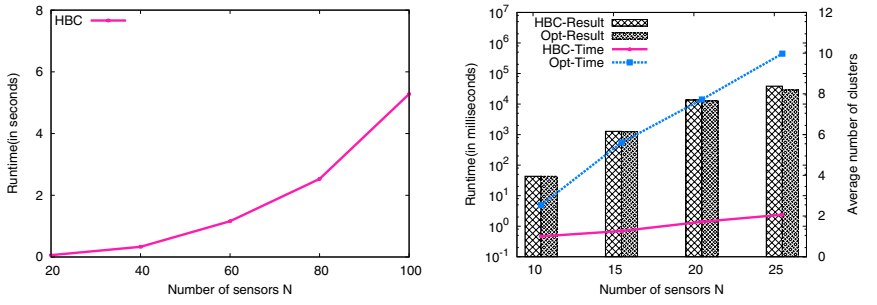


Fig. 4. Performance of HBC

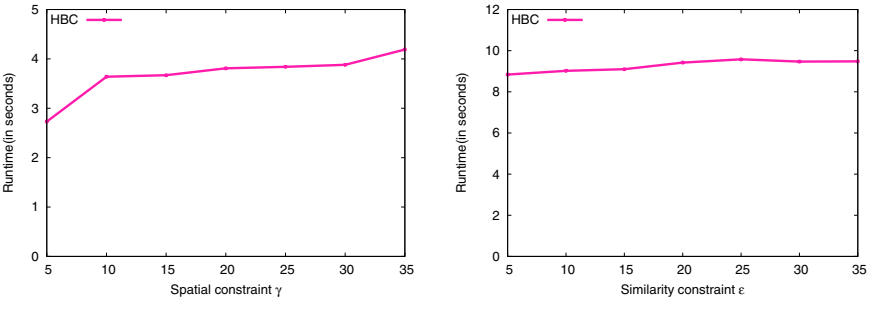


Fig. 5. Sensitivity analysis on r and ε

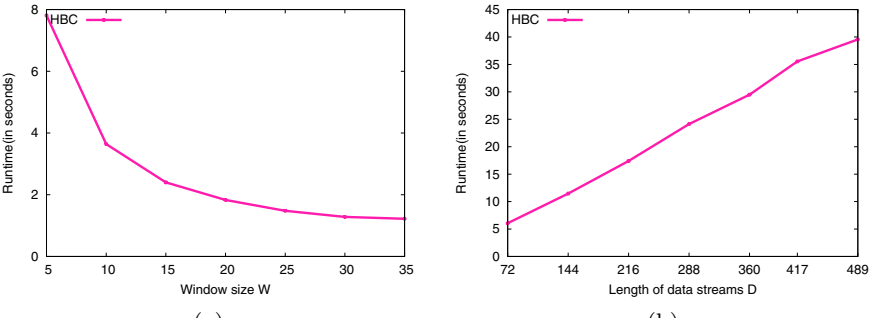


Fig. 6. Sensitivity analysis on W and D

clustering, thereby increasing the runtime of algorithm HBC. Figure 5(b) illustrates the performance of algorithm HBC with the similarity constraint ε varied. The effect of ε of algorithm HBC is not obvious with varying ε from 5 to 35 since the runtime of each time window increases a little as ε increases.

Figure 6(a) shows the performance of algorithm HBC with W varied. The runtime of algorithm HBC decreases when W increases. The reason is the number of time windows decrease while W increases, and thus algorithm HBC has smaller runtimes. Finally, we vary the length of data streams and Figure 6(b) shows the performance of algorithm HBC, where the size of time window is fixed (i.e., W=10). As the length of data streams increases, more number of time windows is involved. Thus, the runtime of HBC drastically increases.

5 Conclusion

In this paper, we formulated a dual clustering problem in spatial data streams, in which objects are partitioned into disjoint clusters such that at each time window (1) objects in the same cluster satisfy the transitively r-connected relation in the optimization and geography domains, and (2) the number of clusters is as minimal as possible. We propose algorithm HBC to deal with the dual clustering problem. Specifically, objects are represented as a graph structure, called RGraph, where each node represents an object and edges indicate their similarity relationships. In light of RGraph, algorithm HBC iteratively merges clusters. Experimental results show the performance of our proposed algorithm.

References

1. Lin, C.R., Liu, K.H., Chen, M.S.: Dual Clustering: Integrating Data Clustering over Optimization and Constraint Domains. IEEE Trans. on Knowledge and Data Engineering 17(5), 628–637 (2005)
2. Dai, B.R., Huang, J.W., Yeh, M.Y., Chen, M.S.: Adaptive Clustering for Multiple Evolving Streams. IEEE Trans. on Knowledge and Data Engineering 18, 1166–1180 (2006)
3. Meka, A., Singh, A.K.: Distributed Spatial Clustering in Sensor Networks. In: Ioannidis, Y., Scholl, M.H., Schmidt, J.W., Matthes, F., Hatzopoulos, M., Böhm, K., Kemper, A., Grust, T., Böhm, C. (eds.) EDBT 2006. LNCS, vol. 3896, pp. 980–1000. Springer, Heidelberg (2006)
4. Ge, R., Ester, M., Jin, W., Davidson, I.: Constraint-Driven Clustering. In: Proc. of the 13th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 320–329 (2007)
5. Moser, F., Ge, R., Ester, M.: Joint Cluster Analysis of Attribute and Relationship Data Without A-Priori Specification of the Number of Clusters. In: Proc. of the 13th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 510–519 (2007)
6. Tai, C.H., Dai, B.R., Chen, M.S.: Incremental Clustering in Geography and Optimization Spaces. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS, vol. 4426, pp. 272–283. Springer, Heidelberg (2007)
7. Lo, C.H., Peng, W.C.: Efficient Joint Clustering Algorithms in Optimization and Geography Domains. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS, vol. 5012, pp. 945–950. Springer, Heidelberg (2008)

CBDT: A Concept Based Approach to Data Stream Mining

Stefan Hoeglinger, Russel Pears, and Yun Sing Koh

School of Computing and Mathematical Sciences,
Auckland University of Technology, New Zealand
{stehoe92, rpears, ykoh}@aut.ac.nz

Abstract. Data Stream mining presents unique challenges compared to traditional mining on a random sample drawn from a stationary statistical distribution. Data from real-world data streams are subject to concept drift due to changes that take place continuously in the underlying data generation mechanism. Concept drift complicates the process of mining data as models that are learnt need to be updated continuously to reflect recent changes in the data while retaining relevant information that has been learnt from the past. In this paper, we describe a Concept Based Decision Tree (CBDT) learner and compare it with the CVDFT algorithm, which uses a sliding time window. Our experimental results show that CBDT outperforms CVFDT in terms of both classification accuracy and memory consumption.

1 Introduction

There has been increasing agreement amongst the data mining community that certain types of applications that produce large volumes of data on an open-ended basis cannot be mined by inducing models on a randomly drawn sample. This is primarily because data streams are non-stationary in nature and models built on a small random sample will not suffice as concept drift causes a loss of accuracy over time. Examples of such data streams include network event logs, telephone call records, credit card transactional flows, sensor networks and surveillance video streams.

Research in data stream mining has focused on incremental learning approaches that do not re-build models on every batch of new data, but update models continuously as new data arrives in the stream [1,2,3]. In high-speed data streams that produce tens of thousands of instances per second, the challenge lies in accurately capturing concept drift and building accurate models, while keeping pace with the data that streams in continuously. A variety of approaches have been proposed [2,4,5,6,7,8] that are all based on some form of incremental learning.

Most incremental approaches use a sliding time window that is used to update statistics maintained in tree nodes that determine when structural changes need to be made. In certain scenarios time windows are ineffective, as data encapsulating a certain pattern of behavior slides out of the window to be replaced by a new pattern. The re-emergence of the old pattern then presents the learner with a severe disadvantage as its usage statistics now do not hold information on the old pattern, thus causing a loss in accuracy until the old pattern has been captured once again. We believe that a more effective way of

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 1006–1012, 2009.
© Springer-Verlag Berlin Heidelberg 2009

capturing patterns in a data stream is to use actual information content to decide which information should be retained in the model. In this paper, we present a concept-based window approach, which is integrated with a high-speed decision tree learner.

In the next section, we present previous approaches to mining high speed data streams. Thereafter, in section 3, we present our methodology for implementing a concept-based window. Our experimental results and a comparison with CVFDT are presented in section 4. The paper then concludes in section 5 and explores some avenues for future research. The complete version of this paper is available from: http://www.hoeglinger.co.nz/cbdt

2 Related Work

One of the earliest high speed data stream mining algorithms was presented by Domingos and Hulten [1]. The authors describe and evaluate VFDT (Very Fast Decision Tree learner), a decision tree learning system based on the Hoeffding bound. The Hoeffding bound provides a confidence interval in which the mean of a set of n independent observations lies with a given probability. This bound was exploited by determining the number of instances required at each internal node of the decision tree in order that the mean information gain resulting from splitting on a given attribute is within a certain guaranteed interval. One important feature of the Hoeffding bound is that it is independent of the probability distribution of the original dataset. Although VFDT's speed of processing is impressive, its main limitation is its inability to deal with concept drift.

In response to the above problem, VFDT was extended into a new algorithm called CVFDT [2] which was capable of making changes at decision nodes. CVFDT basically uses the same engine as VFDT for growing a tree, but also considers alternate trees at each decision node. This gives it the capability of reacting to changes in concept by replacing the main tree with an alternate as it becomes more accurate as a result of concept drift. While CVFDT is a definite improvement over VFDT, it is still not equipped to deal with certain types of concept drift as mentioned in the introduction to this paper.

In [4], Wang, Fan and Yu take a radically different approach. According to them the most fundamental problem in learning drifting concepts is how to identify data in the training set that are no longer consistent with the current concept. They propose that the expiration of such outdated data must rely on the data's distribution rather than the arrival time of the data.

Hoeglinger and Pears [9] take a similar stand to Wang et al [4] in dispensing with the time-based window. They illustrate the advantages of the concept-based approach and propose a decision tree classifier that uses the idea of a least-used decision path to decide which nodes represent outdated concepts.

3 CBDT System

The Concept Based Decision Tree (CBDT) learner maintains a forest of trees with each tree rooted on a different attribute and grows independently. The fact that every attribute is represented at the root level allows CBDT to be more responsive to changes

in concept drift, by simply switching to the winner tree having the highest classification accuracy.

The information needed for model induction is based on usage statistics that is maintained at each tree node and consists of a 3 dimensional array $S_{i,j,k}$ which records the count of class i for attribute j with value k, where the identity of the class is determined by the model induced so far. Whenever a new instance arrives, it is routed to all trees in the forest and is incorporated into each of the models by updating the node statistics. When a sufficient number of instances have arrived at a node, a statistically significant decision whether or not the node should be split can be made on the basis of the information gain metric and the Hoeffding bound.

The usage statistics S above are incremented each time an instance is presented and are never decremented, as in CVFDT. In addition to these basic statistics, we also maintain a counter C for each node, which records the accumulated total of the number of instances that has passed through that node at any given point in time.

The critical component is the purge control mechanism that decides which information is outdated and therefore needs to be removed from the model. The counter C maintained at each tree node is used to identify the sub-tree containing the least amount of information content, which thus becomes a candidate for removal. The removal of such sub-trees enables us to reverse a split that was made earlier that is no longer viable due to its low information content. Such removal enables new sub-trees to be grown which can capture the evolving concept more faithfully than the previous structure. We now formally define the concept of the *Information Content* of a sub-tree.

Definition 1. *Decision sub-tree*

A decision sub-tree is a sub-tree that consists of a set of leaf nodes and the parent decision node.

Definition 2. *Information Content*

The information content I_{ST} of a decision sub-tree (ST) is a weighted sum of the information gain contained across all leaf nodes within that decision sub-tree. $I_{ST} = \sum_{L=1}^{n_L} W_L.G_L$ where $W_L = \frac{C_L}{C_{Root} - C_{DecisionNode}+1}$ represents the weight associated with a decision path comprising the decision node and leaf L; C_L, C_{root} and $C_{DecisionNode}$ are the counter values at the leaf, root and decision nodes respectively.

When a leaf is split, the new decision node's counter value is replaced by the current counter value in the root node of the tree, thereby allowing us to determine the relative age of the split later on. Thus, immediately after a split, the weights of all leaves in the sub-tree are at their highest possible value. As new data streams in, each leaf weight will gradually change to reflect its importance in terms of the relative number of instances that reach the leaf through its governing decision node. To obtain the information content of a sub-tree, we scale the weight by the information gain G, where G is the highest information gain (taken over all attributes qualifying for a split at the leaf node) contained in the leaf, if it were to split after C_L number of instances has been presented to it. Thus it can be seen that IC_{ST} records the potential information content contained in a decision sub-tree ST. We now need to determine the sub-tree to be removed when CBDT's memory allocator requires memory to be freed. We define the concept of a Least Attractive Sub-Tree (LAST).

Definition 3. *Least Attractive Sub-Tree (LAST)*

The least-attractive sub tree for a given tree T is that decision sub-tree with the least information content taken over all decision sub-trees maintained by tree T.

More formally, we have $LAST_T = \arg\min \bigcup_T IC_{ST}$ where T is a tree in the collection of trees maintained by CBDT. The identification of the $LAST_T$ allows CBDT to keep individual trees within certain size boundaries, but still adjust to conceptual changes in the data stream by learning new information and discarding information with the least impact on the tree's error rate.

The purge-controller module that implements the LAST detection functionality is shown below.

```
Procedure purge controller(i: Tree Number):
/* determine least attractive sub-tree(s) */
for each decision sub tree{
   IC = 0
   for each leaf L in the tree i
      set IC = IC + (C_Leaf/(C_Root − C_DecisionNode + 1)) * G
}
LAST = decision sub-tree with the least information content IC
create a new leaf and combine counters of leaves in LAST into a new leaf
reset the link from the decision node to point to the new leaf
remove the former decision node and its leaves from memory
```

We now turn our attention to CBDT's memory allocation mechanism. As CBDT allows each tree in its forest to grow independently, each tree competes with the others for memory when the total memory available to CBDT is exhausted. The memory is divided amongst the trees in proportion to classification accuracy. However, in order to reward trees that are performing at a much higher accuracy level than its rivals, a logarithmic bias function is used whose value falls sharply as classification accuracy decreases.

```
function memory-allocator(i: Tree Number):
d = total number of attributes = total number of trees maintained by CBDT
rank [i] = rank of tree i in terms of its classification accuracy
for(i = 0; i < d; i++){ //bias each tree's current accuracy
   fbias(rank[i]) = 1 - (log_d(rank[i])-(1-(log_d(rank[i])/(rank[i]*3))))
   bias[i] = accuracy[i] * fbias(rank[i])
   total bias += bias[i]
   // calculate the total biased accuracy over all trees
}
memory allocation[i] = total nodes * bias[i] / total bias
// update each tree's maximum node allowance
return memory allocation
```

We next present the pseudo code for the rest of the CBDT system. The main module runs for each tree in the forest and accumulates split requests from trees into a queue. As the Hoeffding bound requires a number of instances to be collected at a node before a statistically reliable decision can be made on a split, access to the queue only occurs on a periodic basis. When a node is split the memory allocation for the tree involved needs to be re-assessed as memory allocation is dependent on tree accuracy.

```
c the number of classes; d is the number of attributes; SQ is the
sequence of instances in the data stream; X is a set of symbolic attributes
x; y is the class for X; F is the forest of trees; S_ijk node
statistics for class i for attribute j on value k; C_Node counter for the
number of instances at a node

procedure CBDT (c, d, S, X, F, S ,C)
let F be an empty set of trees
for each x in X
   initialise a new tree in F with x as a single node (the root)
for each instance (X, y) in SQ {
   sort (X, y) into each tree in F
   update all statistics
   if Hoeffding bound suggests split put split-request in a FIFO-queue
   if split-requests exist {
      for(i = 0; i < d; i + +) { // poll each tree T in F
        new-max-tree-size [i] = memory_allocator(i)
        //re-assess memory allocation as memory as this is tied to accuracy
        if current-tree-size(i) > new-max-tree-size [i]
          call purge_controller(i)
        for each FIFO-queue entry call split_leaf(i)
      }
}
return most accurate tree from F
```

```
function split-leaf (i: Tree Number):
/* create and initialise a new decision node and new leaves*/
set C_DecisionNode to current C_Root
for each new leaf L that is split on attribute j on value k{
   if designated leaf class is y set C_y = 1 + C_y for leaf to be split
   else
      distribute statistics S to the leaves created
      compute counter C_L = Σ_{i=1}^{c} S_ijk
}
```

4 Experimental Results

We conducted a series of experiments to assess CBDT's ability to cope with different types and levels of concept drift. Our experimentation was conducted on different datasets generated with the synthetic generator used by Hulten and Domingos [1] and Wang et al. [4].

4.1 Experiments

The first dataset, D1 that we used represents a data stream that starts off with a concept that changes slowly over time, followed by a period that exhibits progressively higher levels of drift. We tracked the variation of classification accuracy with the changes in the data stream. We captured the rate of concept drift as the ratio of instances in a particular class (A) to that of the other class (B) and plotted this on a minor axis so that its effect on accuracy can be assessed.

Fig 1 shows that both algorithms exhibit the same behavior when the level of drift is relatively low, with both algorithms maintaining virtually the same level of accuracy. However, as the rate of concept drift gathers momentum the difference between the algorithms becomes apparent. As the rate of concept drift increases, CBDT starts to outperform CVFDT with the rate of improvement increasing as the drift level grows. This is to be expected as CVFDT is heavily influenced by the data in its current time window and changes in the data stream from previous time windows are forgotten as data slides out of scope. Due to this, the changes in CVDFTs alternate trees are not substantial enough to capture the long-term effects of a progressive shift in concept.

In contrast to CVFDT which has a relatively large tree, CBDT performs better due to two very different reasons. First of all, it remembers past trends in data that are

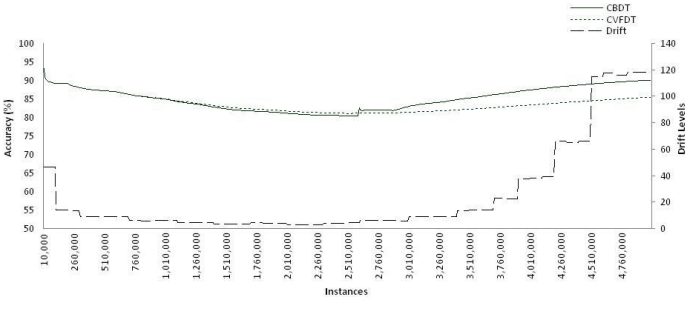


Fig. 1. Comparison of accuracy for CBDT and CVFDT with a smooth drift (Ratio)

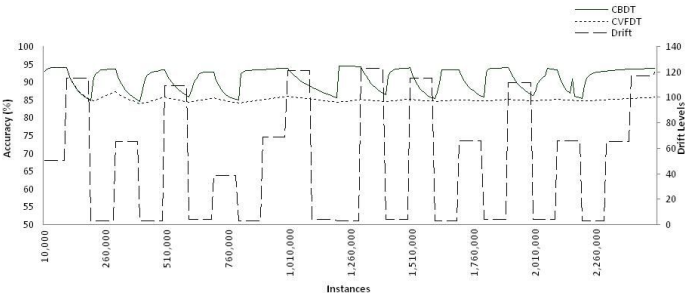


Fig. 2. Comparison of accuracy for CBDT and CVFDT with a dramatic drift (Ratio)

significant. Secondly, its structure is more flexible than that of CVFDTs as it maintains a forest of small trees, rather than one large tree. Thus changes in the stream are easily accommodated, either by making changes at the upper level nodes of trees or the winner switching from one tree to another.

We now examine CBDTs performance on dataset D2 which exhibits very different characteristics to that of D1. Dataset D2 represents a data stream that is highly volatile with a very high volume of high intensity changes in drift. Due to space constraints, we only show the results for the first half of the stream, which is representative of the results for the entire stream. Fig 2 shows that CBDT consistently outperforms CVFDT throughout the range tested. The effect of short-term memory for CVFDT is amplified for dataset D2 and as a result it is insensitive in the face of vast changes in the data stream. CBDT on the other hand reacts to the changes and produces patterns of accuracy that are well synchronized with the changes in the data stream. The only time when the two algorithms perform similarly are in periods of stability in the stream that takes place between rises and falls in the drift level.

We now examine how the algorithms scale with respect to the number of attributes in the dataset. In Fig 3 we track the accuracy of both algorithms as the number of attributes is varied in the range from 5 to 100. We ran both algorithms across 10 datasets with similar drift characteristics to that of D1 and for each value of dimensionality we conducted ten runs and recorded the average accuracy value across the 10 runs. Fig 3 show that both algorithms scale well with the dimensionality of the data with CBDT showing better accuracy.

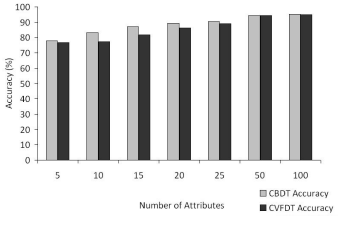


Fig. 3. Variation of Classification Accuracies with data dimensionality

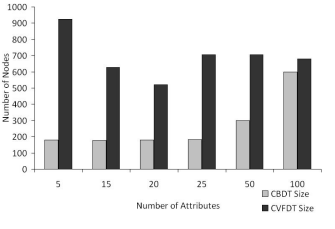


Fig. 4. Memory Consumption by data dimensionality

We also collected data on the memory consumption of both systems in the same runs when running the previous experiment. Fig 4 shows that CVFDTs memory consumption is consistently greater across the range of dimensionality as CBDT does not grow deep trees in periods of high drift, but switches to the winner tree that contains such attributes in its upper levels.

5 Conclusion and Future Work

In this research we have demonstrated the superiority of CBDT over window based approaches such as CVFDT in mining concept drifting data streams. CBDT maintains a higher level of accuracy than CVFDT while using lesser memory. Our experimentation showed that CBDT is able to retain a high level of accuracy in the face of massive changes in concept drift, as demonstrated in our experimentation with dataset D2.

In future research we will address two issues: one is to parallelize the operation of CBDT as each tree is grown independently. Another option would be to devise a metric that would quantify the importance of recurring patterns.

References

1. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 71–80. ACM Press, New York (2000)
2. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: KDD 2001: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 97–106. ACM, New York (2001)
3. Hulten, G., Domingos, P.: Mining complex models from arbitrarily large databases in constant time. In: KDD 2002: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 525–531. ACM, New York (2002)
4. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: KDD 2003: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 226–235. ACM, New York (2003)
5. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: On demand classification of data streams. In: KDD 2004: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 503–508. ACM, New York (2004)
6. Gaber, M.M., Zaslavsky, A., Krishnaswamy, S.: Mining data streams: a review. SIGMOD Rec. 34(2), 18–26 (2005)
7. Ganti, V., Gehrke, J., Ramakrishnan, R.: Mining data streams under block evolution. SIGKDD Explor. Newsl. 3(2), 1–10 (2002)
8. Gama, J., Medas, P., Rodrigues, P.: Learning decision trees from dynamic data streams. In: SAC 2005: Proceedings of the 2005 ACM symposium on Applied computing, pp. 573–577. ACM, New York (2005)
9. Hoeglinger, S., Pears, R.: Use of Hoeffding trees in concept based data stream mining. In: Proceedings of the Third International Conference on Information and Automation for Sustainability, pp. 57–62 (2007)

Meaningful Subsequence Matching under Time Warping Distance for Data Stream

Vit Niennattrakul and Chotirat Ann Ratanamahatana

Department of Computer Engineering, Chulalongkorn University
Phayathai Rd., Pathumwan, Bangkok 10330 Thailand
{g49vnn, ann}@cp.eng.chula.ac.th

Abstract. Since the era of data explosion, research on mining data stream has become more and more active, particularly focusing on improving time and space complexity in similarity subsequence matching problems for data stream. Recently, SPRING algorithm and its variance have been proposed to solve the subsequence matching problem under time warping distance. Unfortunately, these algorithms produce meaningless results since no normalization is taken into account before distance calculation. In this work, we propose a novel subsequence matching algorithm which fully supports global constraint, uniform scaling, and normalization called MSM (Meaningful Subsequence Matching). As expected, our MSM algorithm is much faster and much more accurate than the current existing algorithms in terms of computational cost and accuracy by a very large margin.

Keywords: Subsequence Matching, Dynamic Time Warping Distance, Data Stream, Normalization.

1 Introduction

Since the data explosion era, analysis of data stream in real time is crucial in many data mining tasks including classification, clustering, anomaly detection, and pattern discovery. Generally, these tasks require a subsequence matching algorithm as an important subroutine. Recently, SPRING [1], a breakthrough subsequence matching algorithm for data stream under Dynamic Time Warping (DTW) distance [2] has been proposed. SPRING can report an optimal subsequence in linear time. More specifically, SPRING incrementally updates DTW distance, for each new streaming data point, only in time complexity of the query sequence's length. After the proposal of SPRING, many authors [3,4] have introduced fast algorithms to improve the subsequence matching. We claim that all of those past research works [1,3,4] are meaningless because the query sequence and candidate sequences from the data stream are not normalized. Normalization [5] is essential to achieve accurate and meaningful distance calculation, as it normalizes the data to have similar offset and distribution, regardless of the distance measure used, especially for DTW distance measure. Unfortunately, as we have mentioned above, current subsequence matching algorithms concern mostly about speed enhancement, not on accuracy nor meaningfulness.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 1013–1020, 2009.
© Springer-Verlag Berlin Heidelberg 2009

In this work, we introduce a novel subsequence matching algorithm called MSM (**M**eaningful **S**ubsequence **M**atching) for data stream under DTW distance. MSM consists of two new ideas, i.e., LB_GUN (**L**ower-**B**ounding distance function under **G**lobal constraint, **U**niform scaling, and **N**ormalization) and SSM (**S**caling **S**ubsequence **M**atrix). LB_GUN is a new lower-bounding distance function which is extended from the well-known lower-bounding function LB_Keogh [6]. The distances for every scaled query sequence are stored in SSM, and then MSM algorithm monitors SSM to report the optimal range query or the optimal top-k query when a new streaming data point is received. From these two ideas, MSM can monitor data stream nearly in linear time, and it also achieves much higher accuracy than existing algorithms as we expect.

The remainder of this paper is organized as follows. We provide some essential background in Section 2, and state the problem definitions in Section 3. MSM, our proposed method, is described in Section 4. Experimental results are reported in Section 5, and our work is concluded in Section 6.

2 Background

In this section, we provide essential background knowledge of Dynamic Time Warping distance measure, global constraint, lower-bounding function for DTW distance, uniform scaling, and normalization.

Dynamic Time Warping Distance Measure. Dynamic Time Warping (DTW) distance measure [2] is a well-known shape-based similarity measure for time series data. It uses a dynamic programming technique to find an optimal warping path between two time series. Suppose we have two time series sequences, a sequence X of length n and a sequence Y of length m. The distance is calculated by the following equations.

$$D(X_{1\ldots n}, Y_{1\ldots m}) = d(x_n, y_m) + \min \begin{cases} D(X_{1\ldots n-1}, Y_{1\ldots m-1}) \\ D(X, Y_{1\ldots m-1}) \\ D(X_{1\ldots n-1}, Y_{1\ldots m}) \end{cases} \quad (1)$$

where $D(X_{1\ldots n}, \emptyset) = D(\emptyset, Y_{1\ldots m}) = \infty$, $D(\emptyset, \emptyset) = 0$, and $\emptyset$ is an empty sequence. Any distance metrics can be used for $d(x_i, y_j)$, including L_1-norm, i.e., $d(x_i, y_j) = |x_i - y_j|$.

Global Constraint. Global constraint efficiently limits the optimal path to give a more suitable alignment. Recently, an R-K band [7], a general model of global constraints, has been proposed. R-K band represents a global constraint by a one-dimensional array R, i.e., $R = \langle r_1, r_2, \ldots, r_i, \ldots, r_n \rangle$, where n is the length of time series, and r_i is the height above the diagonal in y-axis and the width to the right of the diagonal in x-axis. Each r_i value is arbitrary; therefore, R-K band is also an arbitrary-shaped global constraint.

Lower-bounding Function for DTW Distance. Although DTW outperforms many other distance measures, it is known to consume huge computational

cost. LB_Keogh has then been proposed to speed up similarity search. $LB_{Keogh}(Q, C)$ between the query sequence Q and a candidate sequence $C = \langle c_1, \ldots, c_i, \ldots, c_n \rangle$ can be computed by the following equation.

$$LB_{Keogh}(Q, C) = \sum_{i=1}^{n} \begin{cases} |c_i - u_i| & ; \text{if } c_i > u_i \\ |l_i - c_i| & ; \text{if } c_i < l_i \\ 0 & ; \text{otherwise} \end{cases} \quad (2)$$

where $u_i = \max\{q_{i-r_i}, \ldots, q_{i+r_i}\}$ and $l_i = \min\{q_{i-r_i}, \ldots, q_{i+r_i}\}$ are envelope elements calculated from a global constraint $R = \langle r_1, \ldots, r_i, \ldots, r_n \rangle$.

Uniform Scaling. Many research works [8,9] have been shown that when the uniform scaling technique is applied, performance, especially the accuracy, significantly increases. More specifically, uniform scaling technique shinks/ stretches a time series sequence $X = \langle x_1, \ldots x_i, \ldots, x_n \rangle$ to a new time series sequence $Y = \langle y_1, \ldots y_j, \ldots, y_m \rangle$, where $y_j = x_{\lceil j \cdot n/m \rceil}$. We also define a scaling factor f, a ratio between length m of new time series Y and length n of original time series X or $f = \frac{m}{n}$, and define a scaling range $[f_{min}, f_{max}]$, where f_{min} and f_{max} are minimum and maximum scaling factors which give lengths n_{min} and n_{max}, respectively.

Normalization. The two time series sequences are compared using any similarity measure, all the data should first be normalized. Z-normalization [5] has been proposed and widely used in time series data mining community, making mean and standard deviation values of the new time series sequence to be zero and one, respectively. Suppose we normalize time series sequence $X = \langle x_1, \ldots x_i, \ldots, x_n \rangle$ to sequence $Y = \langle y_1, \ldots y_i, \ldots, y_n \rangle$, we can simply formulate transformation function as $y_i = \frac{x_i - \mu_X}{\sigma_X}$, where μ_X and σ_X are the mean and standard deviation of time series sequence X, respectively.

3 Problem Definition

In this paper, we focus on two main query problems, i.e., optimal range query and optimal top-k query. The objective of the optimal range query is to find non-overlapping normalized subsequences from a data stream, whose distance between a candidate sequence and a query sequence must be less than a threshold ϵ, where the query sequence is scaled and normalized under uniform scaling between scaling range $[f_{min}, f_{max}]$. On the other hand, optimal top-k query reports top-k non-overlapping normalized subsequences. Nevertheless, the scaled query sequences and all subsequences in the data stream must be normalized in order to return meaningful results. A naïve method to monitor a data stream first initializes a set of normalized scaled query sequences, and then candidate sequences are extracted from the data stream using sliding-window model. After normalization, distance calculation is performed and optimal results are reported (if any).However, this naïve method requires a complexity as high as $O(n^3)$ for each new incoming streaming data point.

4 Proposed Method

Since the naïve method consumes too large time complexity, we propose a novel approach for subsequence matching which gives meaningful result and gives the optimal result nearly in linear time. We call our proposed method as an MSM algorithm (**M**eaningful **S**ubsequence **M**atching), which contains two new ideas, i.e., LB_GUN (**L**ower-**B**ounding function under **G**lobal constraint, **U**niform scaling, and **N**ormalization) and SSM (**S**caling **S**ubsequence **M**atrix). LB_GUN is a lower-bounding function of DTW distance which incrementally calculates lower-bounding distances under global constraint, uniform scaling, and normalization in linear time while guaranteeing no false dismissals. Three following subsections of LB_GUN, SSM, and MSM algorithm are precisely described.

4.1 Lower-Bounding Distance under Global Constraint, Uniform Scaling, and Normalization

LB_GUN (**L**ower-**B**ounding distance under **G**lobal constraint, **U**niform scaling, and **N**ormalization) is a lower-bounding function of DTW distance extended from LB_Keogh [6] whose distance calculation can be done in linear time. Before calculation, LB_GUN first creates an envelope E' from scaled and normalized envelopes. More specifically, three sequence sets are generated, i.e., sets of $\mathbb{Q}$, $\mathbb{R}$, and $\mathbb{E}$. The scaled query set $\mathbb{Q} = \left\{Q'_{n_{min}}, \ldots, Q'_k, \ldots, Q'_{n_{max}}\right\}$ is first generated by scaling and normalizing a query sequence Q to every normalized scaled query sequence $Q'_k = \left\langle q'_1, \ldots, q'_j, \ldots, q'_k \right\rangle$, and the scaled global constraint $\mathbb{R} = \left\{R'_{n_{min}}, \ldots, R'_k, \ldots, R'_{n_{max}}\right\}$ set is derived from scaling a specific global constraint set R with all possible scaling lengths from n_{min} to n_{max}. An envelope E_k of a normalized scaled query sequence Q'_k and a scaled global constraint R'_k for sequence length k is created as in LB_Keogh, and is stored in the envelope set $\mathbb{E} = \{E_{n_{min}}, \ldots, E_k, \ldots, E_{n_{max}}\}$. Then, E' is generated by merging all envelopes in the set $\mathbb{E}$ together, where $E' = \left\{\langle u'_1, l'_1\rangle, \ldots, \langle u'_i, l'_i\rangle, \ldots, \left\langle u'_{n_{max}}, l'_{n_{max}}\right\rangle\right\}$. To find lower-bounding distance between a query sequence Q and a candidate sequence C under global constraint, uniform scaling, and normalization, an envelope E' of a query sequence Q is generated as mention above. $LB_{GUN}(Q, C, n)$ is shown in Equation 3.

$$LB_{GUN}(Q, C, n) = \frac{1}{\sigma_{C_{1\ldots n}}} \left(\sum_{i=1}^{n} \alpha_i + \mu_{C_{1\ldots n}} \sum_{i=1}^{n} \beta_i \right) + \sum_{i=1}^{n} \gamma_i \tag{3}$$

where $\alpha_i = \begin{cases} c_i & ; c'_i \geq u'_i \\ -c_i & ; c'_i \leq l'_i \\ 0 & ; \text{otherwise} \end{cases}$, $\beta_i = \begin{cases} -1 & ; c'_i \geq u'_i \\ 1 & ; c'_i \leq l'_i \\ 0 & ; \text{otherwise} \end{cases}$, $\gamma_i = \begin{cases} -u'_i & ; c'_i \geq u'_i \\ l'_i & ; c'_i \leq l'_i \\ 0 & ; \text{otherwise} \end{cases}$,

$\mu_{C_{1\ldots n}}$ and $\sigma_{C_{1\ldots n}}$ are arithmetic mean and standard deviation of data points 1 to n of a candidate sequence C, $c'_i = \frac{c_i - \mu_{C_{1\ldots i}}}{\sigma_{C_{1..i}}}$, n_{min} and n_{max} are desired scaling lengths, and $n_{min} \leq n \leq n_{max}$.

4.2 Scaling Subsequence Matrix

SSM (**S**caling **S**ubsequence **M**atrix) is another important component in MSM algorithm. It stores lower-bounding distances determined by LB_GUN for each new incoming streaming data point s_t at time t from data stream S. Suppose we have a query sequence Q; each element $\langle t, j\rangle$ of the matrix contains five values, i.e., $v_{t,j}$, $w_{t,j}$, $x_{t,j}$, $y_{t,j}$, and $z_{t,j}$, where $v_{t,j}$, $w_{t,j}$, $x_{t,j}$, $y_{t,j}$, and $z_{t,j}$ are summation of α_t, β_t, γ_t, s_t, and $(s_t)^2$ from time $t-j$ to time t, respectively. Therefore, values in matrix element $\langle t, j\rangle$ can be incrementally updated from the matrix element $\langle t-1, j-1\rangle$ according to the following equations.

$$
\begin{aligned}
v_{t,j} &= v_{t-1,j-1} + \begin{cases} s_t & ; s'_t \geq u'_j \\ -s_t & ; s'_t \leq l'_j \\ 0 & ; \text{otherwise} \end{cases} \qquad w_{t,j} = w_{t-1,j-1} + \begin{cases} -1 & ; s'_t \geq u'_j \\ 1 & ; s'_t \leq l'_j \\ 0 & ; \text{otherwise} \end{cases} \\
x_{t,j} &= x_{t-1,j-1} + \begin{cases} -u'_j & ; s'_t \geq u'_j \\ l'_j & ; s'_t \leq l'_j \\ 0 & ; \text{otherwise} \end{cases} \qquad y_{t,j} = y_{t-1,j-1} + s_t \\
z_{t,j} &= z_{t-1,j-1} + (s_t)^2 \qquad lb_{t,j} = \frac{1}{\sigma_{t,j}} \left(v_{t,j} + \mu_{t,j} \cdot w_{t,j}\right) + x_{t,j}
\end{aligned}
\tag{4}
$$

where $s'_t = \frac{s_t - \mu_{t,j}}{\sigma_{t,j}}$, $\mu_{t,j} = \frac{y_{t,j}}{j}$, $\sigma_{t,j} = \sqrt{\frac{z_{t,j}}{j} - (\mu_{t,j})^2}$, u_j and l_j are from an enveloped E' generated from a query sequence Q, $1 \leq j \leq n_{max}$, $n_{min} \leq j \leq n_{max}$, and $lb_{t,j}$ is a lower-bounding distance LB_GUN for an element $\langle t, j\rangle$.

4.3 Meaningful Subsequence Matching

Since SSM is updated at every arrival of new streaming data point s_t, our MSM algorithm can monitor both optimal range query and optimal top-k query. More specifically, for optimal range query, MSM first calculates and updates values including lower-bounding distances in SSM, and then checks whether best-so-far distance d_{best} is smaller than threshold ϵ. If so, MSM reports an optimal subsequence when there is no overlapping subsequence or when lower-bounding distance is larger than d_{best} for all elements, and MSM resets d_{best} and values in SSM. For all lower-bounding distance which is smaller than d_{best} in range from n_{min} to n_{max}, DTW distance is computed to update d_{best} and the optimal subsequence's position. Additionally, MSM uses only two columns of SSM that are values in time t and values in time $t-1$. The MSM algorithm for optimal range query is described in Table 1.

MSM algorithm for optimal top-k query is implemented based on the optimal range query. With a priority queue, MSM stores the k-best non-overlapping subsequence with DTW distance from the result of MSMOptimalRangeQuery. First, we initialize a threshold ϵ to positive infinity. Then, for every new streaming data point s_t, the queue is updated, and the threshold ϵ is set to the largest DTW distance in the queue. The MSM algorithm for optimal top-k query is described in Table 2.

Table 1. MSM algorithm for optimal range querying

ALGORITHM MSMOPTIMALRANGEQUERY

```
Input: a new streaming data point s_t
Output: an optimal subsequence (if any)
update v_i, w_i, x_i, y_i, and z_i for all i, 1 ≤ i ≤ n_max and lb_i for all i, n_min ≤ i ≤ n_max
if (d_best < ε and ∀i, t_best^end ≤ t − i ∨ lb_i > d_best)
    REPORT(d_best, S[t_best^end, t_best^start])
    d_best = ∞
    reset v_i, w_i, x_i, y_i, and z_i = ∞ for all i, t_best^end > t − i
for (i = n_min to n_max)
    if (lb_i ≤ d_best)
        distance = DTW(Q'_i, NORMALIZE(S[t − i : t]))
        if (distance ≤ d_best)
            d_best = distance; t_best^end = t − i; t_best^start = t
substitute v'_i, w'_i, x'_i, y'_i, and z'_i for v_i, w_i, x_i, y_i, and z_i
```

Table 2. MSM algorithm for optimal top-k querying

ALGORITHM MSMOPTIMALTOPKQUERY

```
1 Input: a new streaming data point s_t
2 Output: update set P of top-k subsequence
3 [C, d_C] = MSMOPTIMALRANGEQUERY(s_t, ε)
4 if (C ≠ NULL)
5    P.push(C, d_C)
6    if (size(P) > k)
7       P.pop()
8       ε = P.peek().d_C
```

5 Experiment

Since none of the current subsequence matching algorithms under DTW distance can handle the changes of data distribution, offset, and scaling, we compare our proposed method with naïve approach in terms of computational time only since our proposed method and the naïve method will both achieve the same accuracy. On the other hand, we compare our accuracy with SPRING, the best existing subsequence matching under DTW distance. Note that we do not compare our running time with that of SPRING; while SPRING will have smaller running time, its results are inaccurate without normalization, and therefore is not a reasonable comparison.

Streaming datasets are generated by combining training data sequences between UCR classification/clustering datasets [10] and synthesized random walk sequences. A stream is initialized with a random walk sequence, and then a training data sequence is appended to the stream. To smooth the stream, before concatenation, each sequence is offset by the last value of the stream.

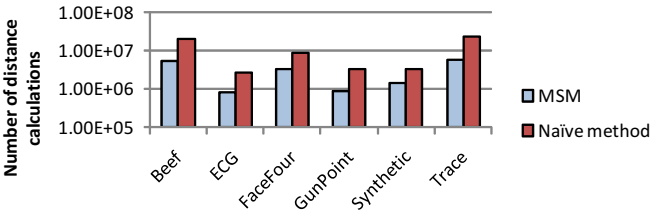


Fig. 1. MSM compares with the naïve method in terms of distance calculation

In the first experiment, we compare our MSM algorithm with naïve method in terms of computational cost by measuring the number of distance calculations. As expected, MSM is much faster than the naïve method by a large margin as shown in Figure 1. Additionally, in MSM, LB_GUN is efficiently used to filter out several candidate sequences while guaranteeing no false dismissal in linear time; therefore, MSM algorithm requires only a small number of DTW distance calculations comparing with the naïve method.

Then, we compare our MSM algorithm with SPRING to measure performance in terms of accuracy, both Accuracy-on-Retrieval (AoR) and Accuracy-on-Detection (AoD). AoR reflects quality of an algorithm that is able to find the patterns in data stream; on the other hand, AoD reflects quality of the returned results. Suppose we have data stream S, a set of expected pattern sequences E, and a set of retrieved sequences R. We first define an overlapping subsequence. Let $S[t_s : t_e]$ be the subsequence starting at t_s and ending at t_e. Overlapping subsequence $O_{X,Y}$, where $X = S[a : b]$ and $Y = S[c : d]$, and overlap percentage $P_{X,Y}$ are defined as $O_{X,Y} = S[\max\{a, c\} : \min\{b, d\}]$ and $P_{X,Y} = \frac{|O_{X,Y}|}{\max\{b,d\}-\min\{a,c\}+1}$, respectively. Both AoR and AoD can be defined over overlapping subsequence $O_{X,Y}$ and overlapping percentage $P_{X,Y}$ as $AoR = \frac{|\{O_{X,Y}|P_{X,Y}>p, X\in R, Y\in E\}|}{|E|}$ and $AoD = \frac{\sum\{P_{X,Y}|P_{X,Y}>p, X\in R, Y\in E\}}{|\{O_{X,Y}|P_{X,Y}>p, X\in R, Y\in E\}|}$, respectively, where p is a threshold of $P_{X,Y}$ that defines a sequence in R as a found sequence. The results, shown in Table 3, illustrate that MSM produces more meaningful result since SPRING does not support global constraint, uniform scaling, nor normalization.

Table 3. Our proposed method, MSM, outperforms SPRING in both AoR and AoD

Dataset	AoR		AoD	
	SPRING	MSM	SPRING	MSM
Beef	6.67%	**50.00%**	0.17%	**89.64%**
ECG200	25.47%	**78.26%**	46.54%	**91.19%**
FaceFour	15.91%	**62.50%**	5.52%	**94.90%**
Gun Point	19.56%	**66.67%**	17.07%	**89.86%**
Synthetic Control	8.93%	**36.00%**	30.65%	**56.08%**
Trace	0.96%	**73.07%**	12.70%	**86.07%**

6 Conclusion

This paper proposes a novel and meaningful subsequence matching algorithm, so called MSM (**M**eaningful **S**ubsequence **M**atching), under global constraint, uniform scaling, and normalization. Two ideas have been introduced in MSM algorithm, i.e., a lower-bounding distance (LB_GUN) and a scaling subsequence matrix (SSM). LB_GUN can update lower-bounding distance incrementally under normalization, while guaranteeing no false dismissals in linear time. The values of calculation are then stored in SSM for each candidate sequence. With these two ideas, MSM algorithm can efficiently monitor data stream and can answer both optimal range query and optimal top-k query problems. Since none of the current algorithm produces meaningful result, we evaluate our proposed method comparing with the naive method in terms of time consumption and SPRING, the best existing subsequence matching under DTW distance, in terms of accuracies. As we have expected, our MSM algorithm is much faster and more accurate by a very large margin.

Acknowledgement

This research is partially supported by the Thailand Research Fund given through the Royal Golden Jubilee Ph.D. Program (PHD/0141/2549 to V. Niennattrakul) and the Thailand Research Fund (Grant No. MRG5080246).

References

1. Sakurai, Y., Faloutsos, C., Yamamuro, M.: Stream monitoring under the time warping distance. In: ICDE 2007, April 15-20, pp. 1046–1055 (2007)
2. Ratanamahatana, C.A., Keogh, E.J.: Three myths about dynamic time warping data mining. In: SDM 2005, April 15-20, pp. 506–510 (2005)
3. Athitsos, V., Papapetrou, P., Potamias, M., Kollios, G., Gunopulos, D.: Approximate embedding-based subsequence matching of time series. In: SIGMOD 2008, June 10-12, pp. 365–378 (2008)
4. Zou, P., Su, L., Jia, Y., Han, W., Yang, S.: Fast similarity matching on data stream with noise. In: ICDEW 2008, April 7-12, pp. 194–199 (2008)
5. Han, J., Kamber, M.: Data Mining: Concepts and Techniques (2001)
6. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowledge and Information Systems 7(3), 358–386 (2005)
7. Ratanamahatana, C.A., Keogh, E.: Making time-series classification more accurate using learned constraints. In: SDM 2004, April 22-24, pp. 11–22 (2004)
8. Fu, A.W.C., Keogh, E.J., Lau, L.Y.H., Ratanamahatana, C.A., Wong, R.C.W.: Scaling and time warping in time series querying. The VLDB Journal 17(4), 899–921 (2008)
9. Yankov, D., Keogh, E.J., Medina, J., Chiu, B., Zordan, V.B.: Detecting time series motifs under uniform scaling. In: KDD 2007, August 12-15, pp. 844–853 (2007)
10. Keogh, E., Xi, X., Wei, L., Ratanamahatana, C.A.: UCR time series classification/clustering page, `http://www.cs.ucr.edu/~eamonn/time_series_data`

An Aggregate Ensemble for Mining Concept Drifting Data Streams with Noise

Peng Zhang[1], Xingquan Zhu[2], Yong Shi[1,3], and Xindong Wu[4]

[1] FEDS Center, Chinese Academy of Sciences, Beijing, 100190, China
zhangpeng04@mails.gucas.ac.cn
[2] Dept. of Computer Sci. & Eng., Florida Atlantic Univ., Boca Raton, FL, 33431, USA
xqzhu@cse.fau.edu
[3] College of Inform. Sci. & Tech., Univ. of Nebraska at Omaha, Omaha, NE 68182, USA
yshi@gucs.ac.cn
[4] Dept. of Computer Science, University of Vermont,Burlington, Vermont 05405, USA
xwu@cs.uvm.edu

Abstract. Recent years have witnessed a large body of research work on mining concept drifting data streams, where a primary assumption is that the up-to-date data chunk and the yet-to-come data chunk share identical distributions, so classifiers with good performance on the up-to-date chunk would also have a good prediction accuracy on the yet-to-come data chunk. This "stationary assumption", however, does not capture the concept drifting reality in data streams. More recently, a "learnable assumption" has been proposed and allows the distribution of each data chunk to evolve randomly. Although this assumption is capable of describing the concept drifting in data streams, it is still inadequate to represent real-world data streams which usually suffer from noisy data as well as the drifting concepts. In this paper, we propose a Realistic Assumption which asserts that the difficulties of mining data streams are mainly caused by both concept drifting and noisy data chunks. Consequently, we present a new Aggregate Ensemble (AE) framework, which trains base classifiers using different learning algorithms on different data chunks. All the base classifiers are then combined to form a classifier ensemble through model averaging. Experimental results on synthetic and real-world data show that AE is superior to other ensemble methods under our new realistic assumption for noisy data streams.

Keywords: Ensemble Learning, Noisy Data Streams, Concept Drifting.

1 Introduction

Recent years have seen a large body of work on pattern discovery and data mining from data streams. From the supervised learning perspective, all existing solutions build stream data mining models under two major assumptions: Stationary Assumption [1, 2, 3, 4, 5, 6, 8,11] and Learnable Assumption [9, 10].

The Stationary Assumption assumes that training data chunks have a similar or identical distribution as the yet-to-come data chunk, so classifiers built from the training data chunks will perform well in classifying data from the yet-to-come data

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 1021–1029, 2009.
© Springer-Verlag Berlin Heidelberg 2009

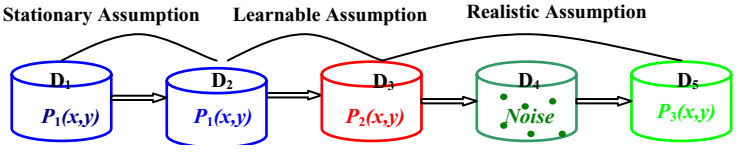


Fig. 1. A conceptual demonstration of the realistic data stream

chunk. However, this stationary assumption violates the concept drifting reality in data streams where prior knowledge on when and where the concepts may change is very difficult, if not impossible, to acquire. As a result, the stationary assumption is not capable of capturing all realistic situations of data streams.

Noticing the limitations of the stationary assumption, a recent paper [9] proposed a "Learnable Assumption" which assumes that the training chunks have different distributions $p(x,y)$ (where x denotes the feature vector and y denotes the class label) from that of the yet-to-come data chunk, and classifiers built on the training set may perform only slightly better than random guessing or simply predicting all examples to a single class. Under the learnable assumption, building classifiers on the up-to-date data chunk to predict the yet-to-come data chunk is better than building classifiers on a lot of buffered chunks because the buffered chunks will degenerate the ensemble performance. However, the disadvantage of the learnable assumption is also obvious, in the sense that it doesn't discriminate concept drifting from data errors. If the up-to-date data chunk is a noisy chunk, building classifiers on this noisy data chunk to predict the yet-to-come data chunk may cause more errors than using a classifier ensemble built on previously buffered data chunks. Consequently, although the learnable assumption is more reasonable than the stationary assumption for data streams, in practice, it is still not capable of describing the stream data.

Considering a toy data stream consisting of five data chunks as shown in Figure 1, the stationary assumption can only cover the process from D_1 to D_2, where the distribution $p_1(x,y)$ remains unchanged. The learnable assumption covers the process from D_2 to D_3, where the concept drifts from $p_1(x,y)$ to $p_2(x,y)$ without interrupted by noisy data chunks. However, the most common situation of a realistic data stream is depicted as the process from D_3 to D_5, where the concept drifting ($p_2(x,y)$ evolves to $p_3(x,y)$) is mixed with noise (a noisy data chunk D_4 is observed). To explicitly describe this type of data streams, we define a realist assumption as follows:

Realistic Assumption: *In realistic data streams, the streams may experience concept drifting and data errors simultaneously.*

The realistic assumption addresses both concept drifting and data errors in data streams. It is much more general than the stationary and learnable assumptions. So it can be widely adopted to help solve various stream data mining challenges. In this paper, we employ this realistic assumption to formulate data streams, and we further propose an aggregate ensemble framework to mine data streams with both concept drifting and data errors.

The rest of this paper is organized as follows. In Section 2, we introduce conventional methods adopted in data streams and then propose a new aggregate ensemble (AE) framework. We analyze the performance of the Aggregate Ensemble method in Section 3 and report experimental results on both synthetic and real-life datasets in Section 4. The conclusions and remarks are given in Section 5.

2 Ensemble Approaches for Stream Data Mining

The nature of continuous volumes of the stream data raises the needs of designing effective classifiers with high *accuracy* in predicting the future testing data chunk as well as good *efficiency* to learn from the massive volumes of training instances[12,13]. To overcome these challenges, a number of ensemble methods have been proposed.

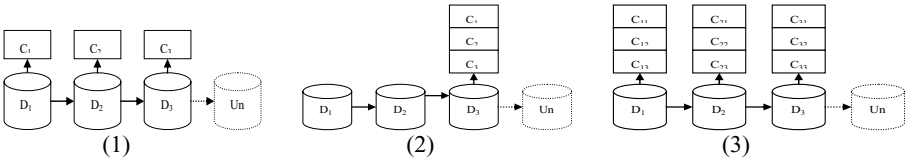


Fig. 2. A conceptual flowchart of the classifier ensemble framework for stream data mining where (1) shows the horizontal ensemble framework, which builds different classifiers on different data chunks; (2) shows the vertical ensemble framework, which builds different classifiers on the up-to-date data chunk with different learning algorithms; and (3) shows the aggregate ensemble framework, which builds classifiers on different data chunks using different learning algorithms

Horizontal Ensemble Framework. Consider a data stream S containing an infinite number of data chunks D_i ($i= -\infty,\ldots, +\infty$). Due to the space limitations, we can only buffer at most n blocks each of which contains a certain number of instances. Assume at the current time stamp we are observing the n^{th} chunk D_n, and the buffered data chunks are denoted by D_1, D_2, …, D_n. In order to predict the data in a new arrival chunk D_{n+1}, one can [2,3,5,6] choose a learning algorithm $\mathcal{L}$ to build a base classifier f_i from each of the buffered data chunks D_i, say $f_i=\mathcal{L}(D_i)$, and then predict each instance x in D_{n+1} by combining the predictions of the base classifiers f_i, i=1, 2, …, N to form a classifier ensemble through the model averaging mechanism as follow:

$$f_{HE}(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x) \tag{1}$$

An illustration of the horizontal ensemble is also given in Figure 2(1). The advantage of the horizontal ensemble is twofold: (1) they can reuse information of the buffered data chunks, which may be useful for the testing data chunk; and (2) they are robust to noisy streams because the final decisions are based on the classifiers trained from different chunks. Even if noisy data chunks may degenerate some base classifiers, the ensemble can still maintain relatively stable prediction accuracy. The disadvantage of such an ensemble framework, however, lies on the fact that if the concepts of the stream continuously change, information contained in previously buffered classifiers may be invalid to the current data chunk. Consequently, combining old-fashioned classifiers may not improve the overall prediction accuracy.

Vertical Ensemble Framework. Assume we have m learning algorithms $\mathcal{L}_j$ ($j=1,2,\ldots,m$), a vertical ensemble [9,10] builds base classifiers using each algorithm on the up-to-date data chunk D_n as, $f_j = \mathcal{L}_j(D_n)$ and then combines all the base classifiers through model averaging. Figure 2(2) gives an illustration of the vertical ensemble framework. In the case that prior knowledge of the yet-to-come data chunk is unknown, model averaging on the most recent chunk can achieve a minimal expectation error on the test set. In other words, building classifiers using different learning algorithms can decrease the expected bias error compared to any single classifiers. For example, assuming a data stream with its joint probability $p(x,y)$ evolves continuously, if we only use a stable learner such as SVM, then SVM may perform better than an unstable classifier when $p(x)$ changes while $p(y|x)$ remains unchanged. On the other hand, if we only use an unstable learner such as decision trees, then decision trees may perform better than SVM when $p(x)$ does not evolve much but $p(y|x)$ changes dramatically. Although the vertical ensemble has a much looser condition than the stationary assumption, it also has a severe pitfall for realistic data streams. The vertical ensemble builds classifiers only on a single up-to-date data chunk. If the up-to-date data chunk is a noisy data chunk, the results may suffer from severe performance deterioration. Without realizing the noise problems, the vertical ensemble limits itself merely to the concept drifting scenarios, but not to the realistic data streams.

Aggregate Ensemble Framework. The disadvantages of the above two ensemble frameworks motivate our proposed Aggregate Ensemble framework as illustrated in Figure 2(3). We first use m learning algorithms $\mathcal{L}_i$ ($i = 1,2,\ldots,m$) to build classifiers on n buffered data chunks j ($j = 1,\ldots, n$), and then train m-by-n base classifiers $f_{ij} = \mathcal{L}_i(D_j)$, where i denotes the i^{th} algorithm, and j denotes the j^{th} data chunk. Then we combine these base classifiers to form an aggregate ensemble through model averaging defined in Eq. (2), which indicates that the aggregate ensemble is a mixture of the horizontal ensemble and vertical ensemble, and its base classifiers constitute a Classifier Matrix (CM) in Eq.(3).

$$f_{AE} = \frac{1}{mn}\sum_{i=1}^{n}\sum_{j=1}^{m} f_{ij}(x) \tag{2}$$

$$CM = \begin{bmatrix} f_{11} & f_{12} & \cdots\cdots & f_{1n} \\ f_{21} & f_{22} & \cdots\cdots & f_{2n} \\ \cdots\cdots & \cdots\cdots & \cdots\cdots & \cdots\cdots \\ f_{m1} & f_{m2} & \cdots\cdots & f_{mn} \end{bmatrix}_{m*n} \tag{3}$$

Each element f_{ij} in CM represents a base classifier built by using algorithm i on data chunk j. As we have mentioned in the vertical ensemble, classifiers on each column of CM (*i.e.*, classifiers built on the same data chunk by different algorithms) are used to reduce the expected classifier bias error on an unknown test data chunk. Classifiers on each row of CM (*i.e.*, classifiers built on different data chunks by using the same algorithm) are used to eliminate the impact of noisy data chunks. By building a classifier matrix CM, the aggregate ensemble is capable of solving a realistic data stream which contains both concept drifting and data errors.

3 Performance Study of the Aggregate Ensemble

As we described above, on each data chunk, the aggregate ensemble builds *m* classifiers by using *m* different learning algorithms. To a specific test instance *x* in the yet-to-come data chunk, the horizontal ensemble uses classifiers on a row in matrix *CM* to predict *x*, *i.e.*, if we choose learning algorithm i ($1 \le i \le m$), then the horizontal ensemble can be denoted by Eq. (4)

$$f_{HE}^{i}(x) = \frac{1}{n}\sum_{j=1}^{n} f_{ij}(x) \tag{4}$$

The vertical ensemble can be denoted by model averaging on the last column (column *n*) of Matrix *CM*, which is given in Eq. (5)

$$f_{VE}^{n}(x) = \frac{1}{m}\sum_{i=1}^{m} f_{in}(x) \tag{5}$$

An aggregate ensemble combines all classifiers in *CM* as base classifiers, through the averaging rule defined by Eq. (2). From this point of view, the horizontal ensemble and vertical ensemble are, in fact, two special cases of the aggregate ensemble. Gao [9] has proven that in data stream scenarios, the performance of a single classifier within a classifier ensemble is expected to be inferior to the performance of the entire classifier ensemble. The horizontal ensemble and vertical ensemble, as special cases of the aggregate ensemble, are not expected as good as the aggregate ensemble. For example, when combining each column in *CM*, we get a variant of *CM* as $CM_c = [g_1, g_2, ..., g_n]$, where each $g_i = [f_{1i}, f_{2i}, ..., f_{mi}]^T$ is independent and shares the same distribution, say *p(g)*. Then the mean squared error of the horizontal ensemble (with the i^{th} algorithm) on a test instance *x* (with class label *y*) can be denoted as:

$$\begin{aligned} &MSE_{HE}^{i}(x) = E_{p(g)}(y - g_i(x))^2 \\ &= E_{p(g)}(y^2 - 2yg_i(x) + g_i^2(x)) = y^2 - 2y \cdot E_{p(g)}g_i(x) + E_{p(g)}g_i^2(x) \end{aligned} \tag{6}$$

For the aggregate ensemble, the mean squared error on *x* can be calculated by

$$\begin{aligned} &MSE_{AE}(x) = E_{p(g)}(y - E_{p(g)}g_i(x))^2 \\ &= E_{p(g)}(y^2 - 2y \cdot E_{p(g)}g_i(x) + E_{p(g)}^2 g_i(x)) = y^2 - 2y \cdot E_{p(g)}g_i(x) + E_{p(g)}^2 g_i(x) \end{aligned} \tag{7}$$

So, the difference between Eqs. (7) and (6) can be calculated by

$$MSE_{AE}(x) - MSE_{HE}^{i}(x) = E_{p(g)}^2 g_i(x) - E_{p(g)}g_i^2(x) \le 0. \ (\text{since } E^2(x) \le E(x^2)) \tag{8}$$

Thus, we assert that the error rate of the aggregate ensemble is expected to be less than or equal to the horizontal ensemble. Similarly, if we regard *CM* as a column vector where each element is a combination of different rows in *CM*, we can show that the mean squared error of the aggregate ensemble is also expected to be less than or equal to that of the vertical ensemble. Another important issue with the aggregate ensemble is to determine the number of data chunks we should use to build the

ensemble. If we buffer too many data chunks, the ensemble may contain lots of invalid patterns that deteriorate the prediction performance, while too few data chunks may not be able to eliminate the noise impact. In our experiments below, we set the number of buffered data chunks to be 3.

4 Experiments

Our testing system is implemented in Java with an integration of the WEKA [14] tool.

Synthetic data streams: We create synthetic data streams as follows. Firstly, we generate instances x_t at time stamp t by a Gaussian distribution $x_t \sim N(\mu_t, \Sigma_t)$, where μ_t is a vector that denotes the distribution center and Σ_t is the covariance matrix. Then we define the potential pattern $p(y|x)$ at time stamp t as:

$$y_t = \frac{1}{r}\sum_{i=1}^{r} a_t \sin(x_t) + \frac{1}{r}\sum_{i=1}^{r} b_t x_t^2 + \varepsilon \tag{9}$$

where r is the number of dimensions, a_t and b_t are r-dimensional vectors, the first nonlinear part ($a\sin x$) of Eq. (9) generates discriminative groups, the second nonlinear part (bx^2) of Eq. (9) generates a nonlinear classification boundary, and the third part is a random noise variable which follows a Gaussian distribution $\varepsilon \sim N(0,0.3^2)$. To simulate concept drifting, we let $p(x,y)$ change randomly. To evolve $p(x)$, we let x's distribution center μ_t move forward by time as $\mu_{t+1} = \mu_t + (-1)^s d$, where s denotes the direction of μ_t, which has a 10% chance to reverse, and d denotes the step length of evolution. To evolve $p(y|x)$, we let b_t have a 50% chance to become $b_{t+1} = b_t + 1$. Finally, we let y_t have a 20% chance to suddenly change to $-y_t$, which is to say, a data chunk at time stamp t has a 20% chance to be a noisy chunk. We can assign class labels by equally dividing the first part of Eq. (9).

KDDCUP'99 Data. Since many research efforts have reported that the concepts underlying this dataset [15] appear to be linearly separable, we complicate the learning task by using the following four approaches to build different types of data streams. (1) Random Selection: we randomly select 100 data chunks, each of which contains 1000 instances with an equal class distribution; (2) Random Noisy Selection: we randomly select 20% data chunks from (1), and then arbitrarily assign each instance a class label which does not equal its original label; (3) Rearranged Selection: we first find the most informative attribute by using the information gain, then we sort all of the instances by this attribute, and the sorted instances are finally put into 100 data chunks each of which contains 1000 instances. (4) Rearranged Noisy Selection: we add 20% noisy data chunks in (3) in a similar way to the procedure in (2).

4.1 Experimental Results

According to each classifier's accuracy, we rank all of the algorithms in an order of 1 (which is denoted as #W) to 5 (which is denoted as #L). We measure the

performances of all the algorithms by using the average accuracy (Aacc), average ranking (AR), standard deviation of the ranks (SR), and the number of #W and #L.

Results on Synthetic Streams. From the results in Tables 1 and 2, we can observe that *among the five algorithms, AE performs the best, with the highest average accuracy and ranking; VE performs the second best, with the second highest accuracy and ranking; HE and WE are considered the third with a tie; and the single tree is the least accurate method for stream data.* Additionally, we can draw several other conclusions: (1) With the same base learners, HE and WE appear to perform similarly, and adding a weight to each base classifier does not seem to be very helpful; (2) HE and WE mostly have the least AR, and they are consistently ranked inferior to AE and VE, but superior to the single tree; (3) VE always has the most winning chance, whereas AE always has the least chance to lose; and (4) compared to other four methods, the single decision tree is the least accurate method for stream data mining, which has the lowest accuracy and ranking, and minimal winning chance.

Table 1. Binary data stream, *p(x,y)* evolves with 20% noisy data chunks

Measures	N = 100, B = 100					N = 1000, B=100				
	Tree	HE	WE	VE	AE	Tree	HE	WE	VE	AE
Aacc	0.572	0.575	0.575	0.596	**0.614**	0.680	0.679	0.679	0.701	**0.704**
AR	3.192	2.697	2.697	2.505	**2.495**	3.323	3.010	3.010	2.392	**2.194**
SR	0.033	**0.001**	**0.001**	0.023	0.003	0.028	**0.000**	**0.000**	0.000	0.020
#W	9	14	14	**45**	33	8	9	9	**54**	35
#L	33	31	31	26	**20**	30	37	37	23	**12**

Table 2. Multi-Class data stream, *p(x,y)* evolves with 20% noisy data chunks

Measures	N = 100, B = 100					N = 1000, B=100				
	Tree	HE	WE	VE	AE	Tree	HE	WE	VE	AE
Aacc	0.401	0.422	0.422	0.476	**0.485**	0.572	0.616	0.616	0.602	**0.646**
AR	3.768	2.849	2.849	2.081	**2.071**	3.091	3.030	3.030	2.404	**2.394**
SR	**0.006**	0.013	0.013	0.012	0.012	0.037	**0.000**	**0.000**	0.020	0.002
#W	7	12	12	**48**	39	8	10	10	**53**	29
#L	51	36	36	15	**11**	15	48	48	26	**9**

Table 3. Random selection results

	Tree	HE	WE	VE	AE
Aacc	0.991	0.993	0.993	0.994	**0.995**
AR	3.232	2.192	2.192	1.778	**1.475**
SR	0.015	0.000	0.000	0.006	0.126
#W	21	34	34	62	**73**
#L	70	39	39	16	**11**

Table 4. Random selection with noise

	Tree	HE	WE	VE	AE
Aacc	0.694	0.822	0.822	0.695	**0.823**
AR	2.929	2.263	2.263	2.182	**2.121**
SR	0.009	0.001	0.001	**0.000**	0.013
#W	20	29	29	**54**	**54**
#L	45	35	35	29	**26**

Table 5. Rearranged selection results

	Tree	HE	WE	VE	AE
Aacc	0.911	0.825	0.825	**0.926**	0.879
AR	1.525	2.000	2.000	**1.475**	2.267
SR	0.022	0.010	0.010	**0.002**	0.016
#W	**74**	58	58	**74**	59
#L	17	35	35	**8**	26

Table 6. Rearranged selection with noise

	Tree	HE	WE	VE	AE
Aacc	0.669	0.676	0.676	0.670	**0.682**
AR	2.424	1.899	1.8990	1.838	**1.737**
SR	0.025	0.008	0.0082	0.047	**0.006**
#W	54	54	54	62	**65**
#L	30	33	33	27	**20**

Results on KDD Cup'99 Data. The results reported in Table 3 validate our conclusion that the weighted ensemble does not have any difference from the horizontal ensemble, and AE always performs the best. Since random selection selects data chunks from the raw data set without any revisions, we can regard it as a realistic data stream. It is safe to say that AE performs the best on this realistic data stream. Table 4 reports a random selection with 20% noise and we can observe VE and the single tree are more vulnerable to noise. Compared with Table 4, the accuracy of VE and the single tree significantly drops, while AE, HE and WE marginally drop. This tells us that buffering a small number of data chunks can prevent a significant drop of accuracy caused by noise. Table 5 lists the results of a rearranged method, and we can observe that VE performs the best on all of the five measurements. The rearrangement procedure, in fact, generates a special data stream according to the learnable assumption. So the classifier ensemble built on the most recent data chunk is better than the classifier ensemble built with several buffered data chunks. That is why VE and the single tree perform better than AE, HE and WE. Table 6 reports the results of a rearranged noise selection method, with 20% noise in addition to the change of $p(x,y)$. Among them, AE achieves the best. This is because the rearranged noise selection method generates a data stream that experiences concept drifting and data errors simultaneously, and AE performs the best under this circumstance.

5 Conclusions

Accurately formulating real-world data stream is the first step in developing effective stream data mining algorithms. In this paper, we argued that the existing stationary assumption, which assumes that the training and test data chunks share similar or identical distributions, is unrealistic for characterizing real-world data streams mainly because it violates the concept drifting nature. On the other hand, the learnable assumption which was tailored to solve the concept drifting problem takes no consideration on data errors. In this paper, we have formulated a Realistic Assumption by taking both concept drifting and data errors into consideration to characterize realistic data streams. In our assumption, we claimed that real data streams may experience concept drifting and data errors simultaneously. Under this assumption, we have proposed an Aggregate Ensemble (AE) method to mine data streams with both concept drifting and data errors. Experimental results on both synthetic and real-life data have demonstrated that AE outperforms all other methods including horizontal ensemble and vertical ensemble methods. Our empirical studies have also investigated the

overall performance of the popular stream mining methods on the data streams characterized by the proposed realistic assumption.

Acknowledgements. This research has been supported by the National Science Foundation of China (NSFC) under Grants No. 60674109 and No. 70621001.

References

[1] Fan, W.: Systematic data selection to mine concept-drifting data streams. In: Proc. of KDD 2004, pp. 128–137 (2004)
[2] Kolter, J., Maloof, M.: Using additive expert ensembles to cope with concept drift. In: Proc. of ICML 2005, pp. 449–456 (2005)
[3] Scholz, M., Klinkenberg, R.: An ensemble classifier for drifting concepts. In: Proc. of ECML/PKDD 2005 Workshop on Knowledge Discovery in Data Streams, pp. 53–64 (2005)
[4] Wang, H., Fan, W., Yu, P., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proc. of KDD 2003, pp. 226–235 (2003)
[5] Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Machine Learning 23, 69–101 (1996)
[6] Street, W., Kim, Y.: A streaming ensemble algorithm (SEA) for large-scale classification. In: Proc. of KDD 2001, pp. 377–382 (2001)
[7] Wang, H., et al.: Suppressing model overfitting in mining concept-drifting data streams. In: Proc. of KDD 2006, pp. 736–741 (2006)
[8] Zhu, X., Zhang, P., Lin, X., Shi, Y.: Active learning from data streams. In: Proc. of ICDM 2007, pp. 757–762 (2007)
[9] Gao, J., Fan, W., Han, J.: On appropriate assumptions to mine data streams: Analysis and Practice. In: Proc. of ICDM 2007, pp. 143–152 (2007)
[10] Zhang, P., Zhu, X., Shi, Y.: Categorizing and mining concept drifting data streams. In: Proc. of KDD 2008, pp. 812–820 (2008)
[11] Yang, Y., Wu, X., Zhu, X.: Combining proactive and reactive predictions of data streams. In: Proc. of KDD 2005, pp. 710–715 (2005)
[12] Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proc. of KDD 2000, pp. 71–80 (2000)
[13] Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proc. of KDD 2001, pp. 97–106 (2001)
[14] Witten, I., Frank, E.: Data mining: practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)
[15] Asuncion, A., Newman, D.J.: UCI Machine Learning Repository, Irvine, CA (2007), `http://www.ics.uci.edu/~mlearn/MLRepository.html`

On Pairwise Kernels: An Efficient Alternative and Generalization Analysis

Hisashi Kashima[1], Satoshi Oyama[2],
Yoshihiro Yamanishi[3], and Koji Tsuda[4]

[1] IBM Research, Tokyo Research Laboratory
[2] Kyoto University, Graduate School of Informatics
[3] Mines ParisTech, Centre for Computational Biology
[4] Max Planck Institute for Biological Cybernetics

Abstract. Pairwise classification has many applications including network prediction, entity resolution, and collaborative filtering. The pairwise kernel has been proposed for those purposes by several research groups independently, and become successful in various fields. In this paper, we propose an efficient alternative which we call *Cartesian kernel*. While the existing pairwise kernel (which we refer to as Kronecker kernel) can be interpreted as the weighted adjacency matrix of the Kronecker product graph of two graphs, the Cartesian kernel can be interpreted as that of the Cartesian graph which is more sparse than the Kronecker product graph. Experimental results show the Cartesian kernel is much faster than the existing pairwise kernel, and at the same time, competitive with the existing pairwise kernel in predictive performance. We discuss the generalization bounds by the two pairwise kernels by using eigenvalue analysis of the kernel matrices.

Keywords: Kernel Methods, Pairwise Kernels, Link Prediction.

1 Introduction

Most phenomena in the world can be represented by sets of entities, and sets of static and dynamic relationships among the entities. Such relationships include friendships among people, actions such as someone clicking an on-line advertisement, and physical interactions among proteins. Supervised pairwise prediction aims to predict such pairwise relationships based on known relationships. It has many applications including network prediction, entity resolution, and collaborative filtering. Models for pairwise prediction should take a pair of instances as its input, and output the relationship between the two instances. In this paper, we focus on pairwise classification problem, where the task is to predict whether or not a relation exists between given two nodes, and we apply kernel methods [1] to this problem. To apply kernel methods to pairwise classification, we need to define a kernel function between two pairs of instances. Interestingly, three research groups have independently proposed an exactly same pairwise kernel by combining two instance-wise kernel functions [2,3,4]. The proposed pairwise kernel matrix is considered as a Kronecker product of two instance-wise kernel matrices. However, the pairwise kernel is significantly time-and-space-consuming since the pairwise kernel matrix is huge. For this reason, only sampled training data have been used in most of its applications.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 1030–1037, 2009.
© Springer-Verlag Berlin Heidelberg 2009

In this paper, we propose a new pairwise kernel called *Cartesian kernel* as a more efficient alternative to the existing pairwise kernel (which we refer to as *Kronecker kernel*). The proposed kernel is defined as a Kronecker sum of two instance-wise kernel matrices, and therefore more computational- and space-efficient than the existing pairwise kernel. The experimental results using numbers of real network data show that the proposed pairwise kernel is much faster than the existing pairwise kernel, and at the same time, competitive with the existing pairwise kernel in predictive performance. Finally, we give the generalization bounds of the two pairwise kernels by using eigenvalue analysis of the kernel matrices [5,6].

2 Pairwise Classification Problem and the Pairwise Kernel

In this section, we introduce the definition of the (binary) pairwise classification problem and review the existing pairwise kernel independently proposed by three research groups [2,3,4]. The standard binary classification problem aims to learn a function $f : V \to \{+1, -1\}$, where V indicates the set of all possible instances. On the other hand, in the (binary) pairwise classification, the goal is to learn a function $f : V^{(1)} \times V^{(2)} \to \{+1, -1\}$, where $V^{(1)}$ and $V^{(2)}$ are two sets of all possible instances. Let us assume that we are given a $|V^{(1)}| \times |V^{(2)}|$ class matrix F whose elements have one of $+1$ (positive class), -1 (negative class), and 0 (unknown). Our task is to fill in the unknown parts of the class matrix which have 0 value. In the context of link prediction, the F can be regarded as the adjacency matrix for a network including $V^{(1)}$ and $V^{(2)}$ as its nodes. The $[\mathsf{F}]_{i_1,i_2} = +1$ indicates that there is a link between $v^{(1)}_{i_1} \in V^{(1)}$ and $v^{(2)}_{i_2} \in V^{(2)}$, the $[\mathsf{F}]_{i_1,i_2} = -1$ indicates that there is no link, and $[\mathsf{F}]_{i_1,i_2} = 0$ indicates that we do not know if there is a link. If the two sets are exclusive, i.e. $V^{(1)} \cap V^{(2)} = \phi$, the network is regarded as a bipartite graph. On the other hand, if the two sets are exchangeable, i.e. $V^{(1)} = V^{(2)} := V$, the F is considered as a $|V| \times |V|$ adjacency matrix F for a set $V = (v_1, v_2, \ldots, v_{|V|})$. If the network is undirected, the F becomes symmetric. If the network is directed, the F is asymmetric, and $[\mathsf{F}]_{i_1,i_2}$ indicates whether or not a link exists from $v_{i_1} \in V$ to $v_{i_2} \in V$. In addition to the adjacency matrix, we have two kernel matrices $\mathsf{K}^{(1)}$ and $\mathsf{K}^{(2)}$ for $V^{(1)}$ and $V^{(2)}$, respectively. In exchangeable cases, $\mathsf{K}^{(1)} = \mathsf{K}^{(2)} := \mathsf{K}$. Note that those kernel matrices are positive semi-definite.

Since we are interested in classification of pairs of instances, we need a kernel function between two pairs of instances if we apply kernel methods [1] to this problem. In many case, it is rather easy to design kernels for two basic instances, so we construct pairwise kernels by using these instance-wise kernels as building blocks. Assume that we want to define a similarity between two pairs of instances $(v^{(1)}_{i_1}, v^{(2)}_{i_2})$ and $(v^{(1)}_{j_1}, v^{(2)}_{j_2})$. It is natural to say two pairwise relationships are similar if elements from two relationships are similar. In other words, they are similar to each other if $v^{(1)}_{i_1}$ and $v^{(1)}_{j_1}$ are similar, and at the same time, $v^{(2)}_{i_2}$ and $v^{(2)}_{j_2}$ are similar. This idea motivates to define the pairwise similarity as the product of two instance-wise similarities as

$$k_{\otimes}((v^{(1)}_{i_1}, v^{(2)}_{i_2}), (v^{(1)}_{j_1}, v^{(2)}_{j_2})) = [\mathsf{K}^{(1)}]_{i_1,j_1}[\mathsf{K}^{(2)}]_{i_2,j_2}. \quad (1)$$

Since products of Mercer kernels are also Mercer kernels [1], the above similarity measure is also a Mercer kernel if the element-wise kernels are Mercer kernels. In

exchangeable and symmetric cases, the pairwise kernel between (v_{i_1}, v_{i_2}) and (v_{j_1}, v_{j_2}) is symmetrized as

$$k_{\otimes}^{\mathrm{SYM}}((v_{i_1}, v_{i_2}), (v_{j_1}, v_{j_2})) = [\mathsf{K}]_{i_1,j_1}[\mathsf{K}]_{i_2,j_2} + [\mathsf{K}]_{i_1,j_2}[\mathsf{K}]_{i_2,j_1}. \quad (2)$$

The prediction of a kernel machine for a pair $(v_{i_1}^{(1)}, v_{i_2}^{(2)})$ is given as $[\mathsf{F}]_{i_1,i_2} = \sum_{(j_1,j_2)} \alpha(v_{j_1}^{(1)}, v_{j_2}^{(2)}) k_{\otimes}((v_{i_1}^{(1)}, v_{i_2}^{(2)}), (v_{j_1}^{(1)}, v_{j_2}^{(2)}))$, where αs are the model parameters of the kernel machine. In exchangeable and symmetric cases, it becomes $[\mathsf{F}]_{i_1,i_2} = \sum_{(j_1,j_2):j_1<j_2} \alpha(v_{j_1}, v_{j_2}) k_{\otimes}^{\mathrm{SYM}}((v_{i_1}, v_{i_2}), (v_{j_1}, v_{j_2}))$.Note that $\alpha(v_{j_1}, v_{j_2})$ for (v_{j_1}, v_{j_2}) such that $j_1 \geq j_2$ is not used because of symmetry. The kernel matrix for the pairwise kernel is equivalently written as the Kronecker product [7] of two instance-wise kernel matrices as $\mathsf{K}_{\otimes} = \mathsf{K}^{(2)} \otimes \mathsf{K}^{(1)}$, where its $(v_{i_1}^{(1)} + v_{i_2}^{(2)}|V^{(2)}|), v_{j_1}^{(1)} + v_{j_2}^{(2)}|V^{(2)}|)$-th element is $k_{\otimes}((v_{i_1}^{(1)}, v_{i_2}^{(2)}), (v_{j_1}^{(1)}, v_{j_2}^{(2)}))$. The pairwise kernel matrix can be interpreted as a weighted adjacency matrix of the Kronecker product graph [8] of the two graphs whose weighted adjacency matrices are the instance-wise kernel matrices. Therefore, we refer to this pairwise kernel as *Kronecker kernel* to distinguish from the one we will propose in the next section.

3 Cartesian Kernel: A New Pairwise Kernel

In this section, we propose a more efficient pairwise kernel. At the end of the previous section, we mentioned the relationship between the existing pairwise kernel and a Kronecker product graph. So, it is natural to imagine that we can design another pairwise kernel based on another kind of product graph. In this paper, we adopt another kind of product graph called Cartesian product graph [8]. Assume that we have two graphs $G^{(1)}$ and $G^{(2)}$ whose sets of nodes are $V^{(1)}$ and $V^{(2)}$, respectively. The product graph of $G^{(1)}$ and $G^{(2)}$ has nodes $V^{(1)} \times V^{(2)}$, each of whose nodes is defined as a pair of nodes from the original graphs. Let $(v_{i_1}^{(1)}, v_{i_2}^{(2)})$ and $(v_{j_1}^{(1)}, v_{j_2}^{(2)})$ be two node pairs in the product graph. In Kronecker product graphs, a link between these two pairs exists if and only if there is a link between $v_{i_1}^{(1)}$ and $v_{j_1}^{(1)}$ in $G^{(1)}$ and there is a link between $v_{i_2}^{(2)}$ and $v_{j_2}^{(2)}$ in $G^{(2)}$. On the other hand, in Cartesian product graphs, a link between these two pairs exists if and only if $v_{i_1}^{(1)} = v_{j_1}^{(1)}$ in $G^{(1)}$ and there is a link between $v_{i_2}^{(2)}$ and $v_{j_2}^{(2)}$ in $G^{(2)}$, or there is a link between $v_{i_1}^{(1)}$ and $v_{j_1}^{(1)}$ in $G^{(1)}$ and $v_{i_2}^{(2)} = v_{j_2}^{(2)}$ in $G^{(2)}$. We can notice that the condition for a link existing in Cartesian product graphs is more strict than that for Kronecker product graphs.

Inspired by the definition of the Cartesian product graph, we define the *Cartesian kernel* between $(v_{i_1}^{(1)}, v_{i_2}^{(2)})$ and $(v_{j_1}^{(1)}, v_{j_2}^{(2)})$ as

$$k_{\otimes}((v_{i_1}^{(1)}, v_{i_2}^{(2)}), (v_{j_1}^{(1)}, v_{j_2}^{(2)})) = [\mathsf{K}^{(1)}]_{i_1,j_1}\delta(i_2 = j_2) + \delta(i_1 = j_1)[\mathsf{K}^{(2)}]_{i_2,j_2}, \quad (3)$$

where δ is an indicator function, which returns 1 when its argument is true and 0 otherwise. Since δ is considered as an identity kernel, the above similarity measure is also

a Mercer kernel if the element-wise kernels are Mercer kernels. In exchangeable and symmetric cases, the kernel between (v_{i_1}, v_{i_2}) and (v_{j_1}, v_{j_2}) is symmetrized as

$$k_{\otimes}^{\mathrm{SYM}}((v_{i_1}, v_{i_2}), (v_{j_1}, v_{j_2})) = [\mathsf{K}]_{i_1,j_1}\delta(i_2 = j_2) + \delta(i_1 = j_1)[\mathsf{K}]_{i_2,j_2}] \\ +[\mathsf{K}]_{i_1,j_2}\delta(i_2 = j_1) + \delta(i_1 = j_2)[\mathsf{K}]_{i_2,j_1}. \quad (4)$$

The kernel matrix of the Cartesian kernel is equivalently written as the Kronecker *sum* [7] of two instance-wise kernel matrices as $\mathsf{K}_{\oplus} = \mathsf{K}^{(2)} \oplus \mathsf{K}^{(1)}$, where the Kronecker sum operation is defined as $\mathsf{K}^{(2)} \oplus \mathsf{K}^{(1)} = \mathsf{K}^{(2)} \otimes \mathsf{I} + \mathsf{I} \otimes \mathsf{K}^{(1)}$. At the first sight, the size of the Cartesian kernel matrix is the same as that of the Kronecker kernel. But, the number of the non-zero elements in the kernel matrix is much smaller, since the Cartesian kernel is based on the Kronecker products of an element-wise kernel matrix and an identity matrix.

Finally, we mention computational efficiency of the Cartesian kernel. While the Kronecker kernel can give a score greater than 0 between arbitrary two pairs, the Cartesian kernel can give a non-zero value only to the pairs which share one of their instances. This fact indicates that the Cartesian kernel is much faster than the Kronecker kernel.

4 Experiments

In this section, we show several experimental results in which we compare the Kronecker kernel and the Cartesian kernel.

We used three data sets for network prediction. Two of them are biological networks, and one of them is a social network. All data are symmetric networks. The first data set [9] contains the metabolic pathway network of the yeast S. Cerevisiae in the KEGG/PATHWAY database [10]. Proteins are represented as nodes, and a symmetric edge indicates that the two proteins are enzymes that catalyze successive reactions. The number of nodes in the network is 618, and the number of links is 2,782. In this data set, four element-wise kernel matrices based on gene expressions, chemical information, localization sites, and phylogenetic profiles are given. We used them as the kernel matrices or the similarity matrices[1]. The second data set is a protein-protein interaction network constructed by von Mering et al. [11]. We followed Tsuda and Noble [12], and used the medium confidence network. This network contains $2,617$ nodes and $11,855$ symmetric links. Each protein is given a 76-dimensional binary vector, each of whose dimensions indicates whether or not the protein is related to a particular function. We used the inner products of the vectors as the element-wise kernel matrix[2]. The third data set is a social network representing the co-authorships in the NIPS conferences, containing $2,865$ nodes and $4,733$ links. Authors correspond to nodes, and a symmetric link between two nodes indicates that there is at least one co-authored paper by the corresponding authors. Each author is given a feature vector, each of whose dimensions corresponds to occurrences of a particular word in the author's papers. We used the inner product of the vectors as the element-wise kernel matrix[3]. All of the element-wise kernel matrices are normalized so that all of their diagonals are 1. The models were

[1] Available at http://web.kuicr.kyoto-u.ac.jp/supp/yoshi/ismb05/.

[2] Available at http://noble.gs.washington.edu/proj/maxent/.

[3] Available at http://ai.stanford.edu/~gal/data.html.

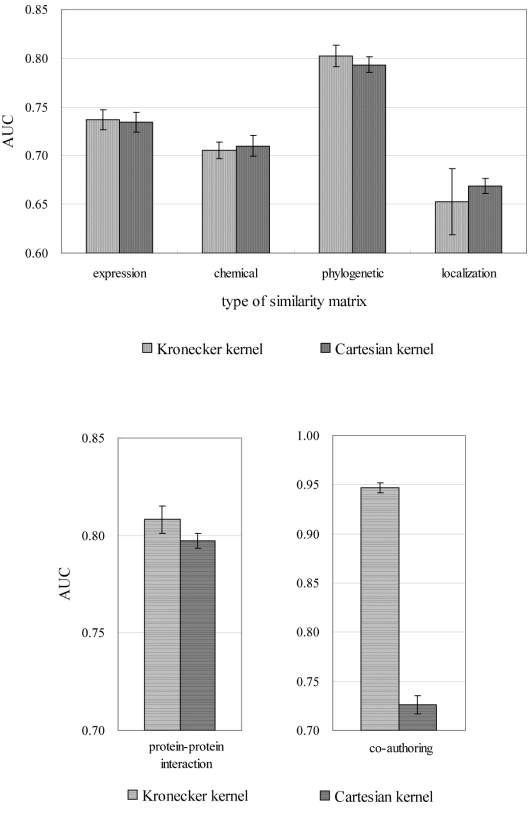


Fig. 1. Summary of results for the KEGG metabolic network (upper), the protein-protein interaction network (lower-left) and the co-authoring network (lower-right)

trained by using PUMMA [13], an on-line learning algorithm whose solutions asymptotically converge to those by the support vector machine with squared hinge loss. The hyperparameter for regularization was set to $C = 1$. All of the training data were processed thee times in the training phase. The results were evaluated in AUC by 5-fold cross validation with 20% of all of the pairwise relationships as training data.

Now, we show the predictive performances and execution times by the two pairwise kernels. In Fig. 1, the gray bars indicate the AUCs by the Kronecker kernel, and the black bars represent the AUCs by the Cartesian kernel. The error bars indicate the standard deviations of the AUC values. In the upper figure of Fig. 1, each of the pairs of AUC bars indicates the results when we used gene expressions, chemical information, phylogenetic profiles, or localization sites for element-wise kernel matrices. In the lower figures, the left pair of the AUC bars is for the protein-protein interaction networks, and the right one is for the co-authoring network.

We can observe that the predictive performance of the Cartesian kernel is competitive with that of the Kronecker kernel except for the co-authoring network. The reason for the degraded performance by the Cartesian kernel in the co-authoring network is not

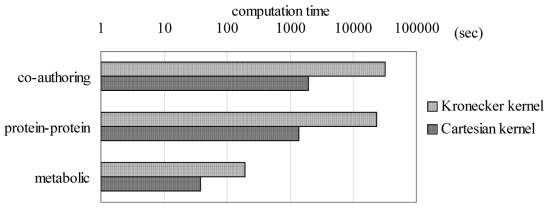


Fig. 2. Comparison of computation time

clear, and we could not find the reason in the theoretical analysis in the following section. But it might be relaled to network sparsity (the co-authoring network is the most sparse), or differences between natures of biological networks and social networks. Figure 2 shows the average training time for each data set in log scale. We can see that the Cartesian kernel is at most 16 times faster than the Kronecker kernel. The differences are remarkable when the network size is large. Based on the above results, we conclude that the Cartesian kernel is a promising alternative to the Kronecker kernel especially for large data sets.

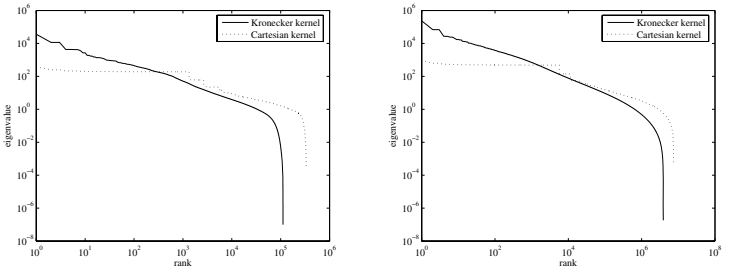
5 Generalization Bounds for the Pairwise Kernels

In this section, we discuss generalization bounds for the Kronecker kernel and the Cartesian kernel. It is known that the generalization bound for a kernel machine such as a support vector machine is given by using the distribution of the eigenvalues of the kernel matrix [5,6]. To compute the generalization bounds, we need to compute the eigenvalues of the Kronecker product $\mathsf{K}_{\otimes}$ or the Kronecker sum $\mathsf{K}_{\oplus}$ of the two instance-wise kernel matrices. It is difficult to directly compute the eigenvalues for the Kronecker product or the Kronecker sum, since their size are very huge.However, we can compute them from the eigenvalues of the instance-wise kernel matrices by using the following theorem [7].

Theorem 1. *Let* $\{\lambda_i^{(1)}\}$ *and* $\{\lambda_j^{(2)}\}$ *be the sets of eigenvalues of kernel matrices* $\mathsf{K}^{(1)}$ *and* $\mathsf{K}^{(2)}$*, respectively. The set of eigenvalues of the Kronecker product* $\mathsf{K}^{(2)} \otimes \mathsf{K}^{(1)}$ *is* $\{\lambda_i^{(1)}\lambda_j^{(2)}\}$ *and the set of eigenvalues of the Kronecker sum* $\mathsf{K}^{(2)} \oplus \mathsf{K}^{(1)}$ *is* $\{\lambda_i^{(1)} + \lambda_j^{(2)}\}$.

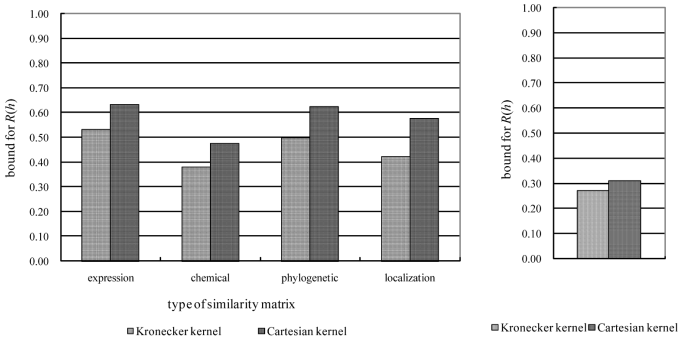
Figure 3 (left) shows the eigenvalues of the Kronecker kernel matrix and the Cartesian kernel matrix derived from phylogenetic profiles for the KEGG metabolic network. Also, Figure 3 (right) shows the eigenvalues for the co-authoring network. The two pairwise kernels, the Kronecker kernel and the Cartesian kernel, yield very different eigenvalue distributions. Although we do not show the eigenvalues for the other networks, the general trend is that the high-ranked eigenvalues of the Kronecker kernel are larger than those of the Cartesian kernel, while the low-ranked eigenvalues of the Cartesian kernel are larger than those of the Kronecker kernel. In other words, the eigenvalues of the Kronecker product decay faster than those of the Kronecker sum.

In the following analysis, we assume that the set of all possible data points $X = \{x_1, x_2, \ldots, x_M\}$ ($X = V^{(1)} \times V^{(2)}$ in our case of pairwise classification) are known



metabolic (phylogenetic) co-authoring

Fig. 3. Eigenvalues of the Kronecker kernel matrices and the Cartesian kernel matrices



metabolic co-authoring

Fig. 4. Bounds of the expected risks for the Kronecker kernel and the Cartesian kernel

in advance of the training phase. Let $Y = \{1, -1\}$ be the set of labels. We also assume that $(x, y) \in Z = X \times Y$ follows a certain distribution $P(x, y)$. The *expected risk* of a hypothesis $h \in \mathcal{H}$ is given by $R(h) = \sum_{(x,y)\in Z} \delta(h(x)y \leq 0)P(x, y)$. Given a set of labeled samples $\{(x_i, y_i) : i \in \{1, \ldots, m\}\}$ from Z with size $m < M$, the *empirical margin risk* for a certain margin γ is defined by the rate of the samples with $h(x_i)y_i < \gamma$: $R_s^\gamma(h) = \frac{1}{m}\sum_{i=1}^m \delta(h(x_i)y_i < \gamma)$. Then, the following theorem [6] gives an upper bound for the expected risk.

Theorem 2. *Let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M$ be the set of eigenvalues of the kernel matrix derived from the set of all possible data points. Consider the hypothesis class $\mathcal{F}(c)_B = \{\langle w, x\rangle + b : \|w\| \leq c, |b| \leq B\}$. Then the following inequality holds simultaneously for all $\gamma \in (8\Upsilon(n), 1]$:*

$$P_{s\in Z^m}\left(\exists h \in \mathcal{H}(c)_B : R(h) \geq R_s^\gamma(h) + \sqrt{\left(n \ln 2 + \ln\left(\frac{\lceil c\rceil}{\theta\gamma}\right)\left\lceil\frac{8B}{\gamma}\right\rceil\right)/(2m)}\right) \leq \theta,$$

where

$$\Upsilon(n) = \min_{j \in \{1,\ldots,n-1\}} 6 \cdot 2^{-\frac{(j-1)}{k(2^j-1)}} \left(\lambda_1, \cdots, \lambda_{k(2^{j-1})}\right)^{\frac{1}{2k(2^j-1)}} c(n,j)$$

$$k(l) = \min \left\{ k \in \{1,\ldots,M\} : \lambda_{k+1} \leq \left(\lambda_1 \cdots \lambda_k l^2\right)^{\frac{1}{k}} \right\}$$

$$c(n,j) = \min \left(1, 1.86\sqrt{\log_2 \left(M/(n-j)+1\right)/(n-j)}\right) \quad .$$

The generalization bounds computed for the metabolic network and the co-authoring network are shown in Fig. 4. The bounds for the Kronecker kernels are smaller than the bounds for the Cartesian kernels. Those theoretical results are consistent with the experimental results in the previous section, where the Kronecker kernels were slightly superior to the Cartesian kernels, although there were a few exceptions and the differences were subtle in the most of the cases. As mentioned in the preceding work [5], the bounds are not very tight. Also, the theoretical result gives no explanation of the large performance difference in the co-authoring network. In future work, we will investigate tighter bounds including the several possibilities for improvements mentioned in the preceding work [5].

References

1. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press, New York (2004)
2. Basilico, J., Hofmann, T.: Unifying collaborative and content-based filtering. In: Proceedings of the 21st International Conference on Machine Learning (ICML) (2004)
3. Ben-Hur, A., Noble, W.S.: Kernel methods for predicting protein-protein interactions. Bioinformatics 21(suppl. 1), i38–i46 (2005)
4. Oyama, S., Manning, C.D.: Using feature conjunctions across examples for learning pairwise classifiers. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) ECML 2004. LNCS, vol. 3201, pp. 322–333. Springer, Heidelberg (2004)
5. Schölkopf, B., Shawe-Taylor, J., Smola, A., Williamson, R.: Generalization bounds via eigenvalues of the gram matrix. Technical Report 99-035, NeuroColt (1999)
6. Kroon, R.: Support vector machines, generalization bounds and transduction, masters thesis, stellenbosch university (2003)
7. Laub, A.J.: Matrix Analysis for Scientists and Engineers. Society for Industrial and Applied Mathematics (2005)
8. Imrich, W., Klavzar, S.: Product Graphs: Structure and Recognition. Wiley, Chichester (2000)
9. Yamanishi, Y., Vert, J.P., Kanehisa, M.: Supervised enzyme network inference from the integration of genomic data and chemical information. Bioinformatics 21, i468–i477 (2005)
10. Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y., Hattori, M.: The KEGG resources for deciphering the genome. Nucleic Acids Research 32, D277–D280 (2004)
11. von Mering, C., Krause, R., Snel, B., Cornell, M., Olivier, S., Fields, S., Bork, P.: Comparative assessment of large-scale data sets of protein-protein interactions. Nature 417, 399–403 (2002)
12. Tsuda, K., Noble, W.S.: Learning kernels from biological networks by maximizing entropy. Bioinformatics 20(suppl. 1), i326–i333 (2004)
13. Ishibashi, K., Hatano, K., Takeda, M.: Online learning of approximate maximum p-norm margin classifiers with biases. In: Proceedings of the 21st Annual Conference on Learning Theory (COLT 2008) (2008)

A Family-Based Evolutional Approach for Kernel Tree Selection in SVMs

Ithipan Methasate[1,2] and Thanaruk Theeramunkong[1]

[1] Sirindhorn International Institute of Technology (SIIT), Thammasat University, 131 Moo 5, Tiwanont Road, Muang, Phathumthani 12000, Thailand
thanaruk@siit.tu.ac.th
[2] National Electronics and Computer Technology Center,112 Phahon Yothin Road, Klong Luang, Pathumthani 12120, Thailand
ithipan.methasate@nectec.or.th

Abstract. Finding a kernel mapping function is a key step towards construction of a high-performanced SVM-based classifier. While some recent methods exploited an evolutional approach to construct a suitable multifunction kernel, most of them searched randomly and diversely. In this paper, the concept of a family of identical-structured kernel trees is proposed to enable exploration of structure space using genetic programming whereas to pursue investigation of parameter space on a certain tree using evolutional strategy. To control balance between structure and parameter search towards an optimal kernel, the trade-off strategy is introduced. By experiments on a number of benchmark datasets from UCI and text classification datasets, the proposed method is shown to be able to find a better optimal solution than other search methods.

Keywords: Genetic Programming, Support Vector Machine, Kernel Selection.

1 Introduction

Support Vector Machine (SVM) is a well-known kernel-based method that is successfully applied in many applications such as text categorization, information retrieval, DNA detection, protein structure detection, intrusion detection and image/speech recognition. SVM classifier requires a kernel mapping function to transform original data in a linear space to a non-linear high-dimensional space by means of dot product of kernel function on a pair of data points, known as kernel trick. However, a common problem in SVM-based classification is how to select a suitable kernel function and its hyperparameters (kernel parameters and SVM tradeoff parameter) that leads to satisfactory performance.

As an early work in kernel selection problem, Chapelle and Vapnik[1] presented a gradient descent algorithm, that guides searching process to an optimal solution by adjusting kernel parameters under consideration of their gradients in order to improve time complexity of searching and also still maintain classification accuracy. Later, Keerthi [2] pointed out an analytical exploration on

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 1038–1045, 2009.
© Springer-Verlag Berlin Heidelberg 2009

several different kernel functions with implementation of radius-margin bound using the gradient descent approach.

As a more complicated approach, Howley and Madden proposed a new approach to use a so-called genetic kernel, which is represented by a tree with a feature vector located at each leaf node. Recently, there have been a series of works on a more flexible kernel so-called multiple kernel learning (MKL). Originally Lanckriet et al. [3] proposed the framework of a linear combination of multiple kernels that resulted in a quadratically-constrained quadratic program (QCQP). Later, Sonnenberg et al. [4] reformulated the MKL problem as a semi-infinite linear program (SILP) and applied it on large scale data. These MKL approaches were designed to find a set of weights in a fixed form of a multi-RBF kernel using numerical analysis. As an alternative, an evolutional method on multi-RBF kernels was proposed by Phienthrakul and Kijsirikul [5]. In their experiments, evolutional strategy was shown to be effective in tuning parameters of a multi-RBF kernel when the number of RBFs increases.

More recently, Sullivan and Luke's work[6] applied genetic programming to search an optimal function in a more general form, where operators and kernels can be arbitrary under the Mercer's condition. Later, Methasate and Theeramunkong [7,8] combined genetic programming and gradient search method to find an optimal kernel function. The concept of a so-called kernel tree was introduced to systematically represent a kernel in the form of a tree under the Mercer's conditions. A hybrid of genetic programming and evolutional strategy was presented in [9] to find an optimal for both kernel structure and kernel parameters.

Towards this issue, we propose an adaptive control mechanism into the evolutional method to find an optimal kernel more efficiently. The concept of a family of identical-structured kernel trees is introduced to explore structure space (a global space) using genetic programming and investigate parameter space (a local subspace) on a specific tree using evolutional strategy.

In the rest of this paper, section 2 presents the kernel tree and kernel tree family. In section 4, the strategy for searching an optimal tree is described. In section 4, experimental results from UCI datasets and text classification are performed. Finally, the conclusion is made in section 5.

2 Kernel Trees and Kernel Tree Family

In this section, the definitions of a kernel tree [8] and a tree family are given. A kernel tree enables the representation of a multi-kernel function in the form of a tree structure. In this work, we introduce the concept of a kernel tree family as a group of structurally identical trees (or isostructural trees). Any two trees are considered as structurally identical when they have the same tree structure with the same node type for each node position but may not occupy the same parameter values for any corresponding edge and any corresponding leaf node. Moreover, we also proposed a mechanism to control the population generation in generic programming, based on this kernel tree family. In the rest, the formal definition of kernel trees is given and then a family of kernel tree is explained.

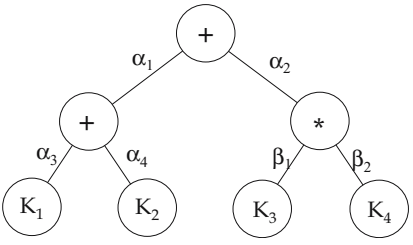


Fig. 1. A kernel tree $T(x_i, x_j)$ of a kernel function with four sub-kernels $K(x_i, x_j) = \alpha_1 \left(K_1(x_i, x_j)^{\beta_1} K_2(x_i, x_j)^{\beta_2} \right) + \alpha_2 (\alpha_3 K_3(x_i, x_j) + \alpha_4 K_4(x_i, x_j))$

2.1 Definition of Kernel Trees

A kernel function $K(x_i, x_j)$ have been used to implement a kernel trick which is a method to exploit a linear classifier to solve a non-linear problem by mapping the original non-linear observations into a higher-dimensional space by a dot product between two points x_i, x_j, as $K(x_i, y_j) = \varphi(x_i)\varphi(y_j)$

Definition 1 (A kernel tree). *A kernel tree $T(\cdot)$ encodes a kernel function $K(\cdot)$ by a tree structure $T = (N_I, N_L, E)$, where an intermediate node $n_i \in N_I$ represents an operator, and a leaf node $n_l \in N_L$ specifies a single kernel function and an edge $e \in E$ expresses a link between node n_i and node n_j with a coefficient.*

In other words, a kernel tree is characterized by the way to assign an operator, a coefficient and a kernel function at each intermediate node, each edge and each leaf node, respectively, in a kernel tree. In this work, an operator in consideration is one of two types: addition and multiplication, a coefficient is a real number, a kernel function is one of the three types: linear, RBF, polynomial function and branching factor is set to be two. Figure 2 displays a kernel tree $T(x_i, x_j)$ of a kernel function with four sub-kernels.

2.2 A Family of Kernel Trees

In this section, first we introduce a normalized structure of a kernel tree and isostructural kernel trees. Thereafter, we describe a family of kernel trees based on these two concepts.

Definition 2 (Normalized Structure of a Kernel Tree). *The normalized structure of a kernel tree is a tree the branches of which are reordered according to their weights calculated from the substructures under these branches, in order to perserve structural regularity. The structural regularity includes the order among leaf node types, the order among intermediate node types, and the order between leaf node types and intermediate node types.*

Technically, since a branch in any tree will have a unique corresponding node (the node under the branch), a weight given to a branch implies the weight given

to its corresponding node. In this work, the weight of a node n, denoted by $V(n)$, is calculated with the following formula. This formula enables the preservation of structural regularity.

$$V(n) = W(n)(B \cdot M)^d + \sum_{c \in N_c} V(c) \tag{1}$$

where B is the maximum branching factor in the tree, M is the maximum number of possible types of a node in the tree, d is the depth level of the node n, and N_c is the set of the child nodes of the node n, $W(n)$ is an index value (between 1 and M) representing the type of the node n.

Figure 2 illustrates a kernel tree, its normalized structure and the weight of each node in the tree. In this figure, B is set to 2 (i.e, a binary tree), M is fixed to 5 (pointing to the three leaf node types; linear, RBF, polynomial functions, and two intermediate node types; addition and multiplication operators). Moreover, the index value ($W(n)$) equals to 1, 2 and 3 when the node n expresses a linear, RBF and polynomial function, respectively. It is 4 and 5 when the node represents an addition operator and a multiplication operator, respectively.

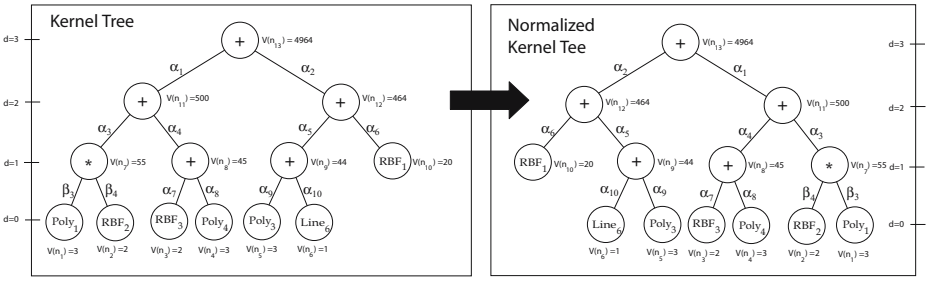


Fig. 2. Normalized structure of a kernel tree

Definition 3 (Isostructural Kernel Trees). *Two kernel trees, T_1 and T_2, are isostructural if and only if the normalized structure of T_1 has the identical structure with the normalized form of T_2.*

It is noted that two isostructural kernel trees will have an identical structure but may have different parameter values at the edges and leaf nodes.

Definition 4 (A Family of Kernel Trees). *A family of kernel trees is defined as a set of isostructural kernel trees.*

3 Searching for Optimal Kernel Trees

In the task of searching the optimal kernel trees, there are two scales of search that are structural search (global search) and parameter search (local search). For structural search, the main target is to find a good kernel tree structure by

evolving with Genetic Programming. On the other hand, the parameter search is done by evolving the parameter vector from the same structure tree.

For a task of global search with GP, crossover and mutation are standard operators for producing a new individual in evolution framework. In GP, crossover operator selects two of parents and randomly switch part of trees between the two selected. Alternatively, mutation operator selects a parent then replace a new structure part to a node randomly. In applied both of these operators, the structure of trees are evolved to find a more efficient tree structure.

Contrast with the structural search, the parameters search focus on tuning the parameters on each kind of selected tree structure. Over the iteration, each family is become more efficient. In this step, the evolution is done from a family mutation operator.

Definition 5 (Mean and Variance of Family). *Suppose that the kernel tree* T_{i,f_j} *,that is a member of tree family* f_j*, has a parameter vector* $v_{T_{i,f_j}}$*. The mean* $,\mu_{f_j}$*, and variance* $,\sigma^2_{f_j}$*, of each family is given by*

$$\mu_{f_j} = \frac{1}{n}\sum_{i=1}^{n} v_{T_{i,f_j}} \tag{2}$$

$$\sigma^2_{f_j} = E[\Big(v_{T_{i,f_j}} - \mu_{f_j}\Big)^2] \tag{3}$$

where n is the number of kernel trees in the family.

Definition 6 (Family Mutation). *Family mutation is a process for generating kernel tree from a group of kernel in a same family. The parameter vector of a new kernel tree is given by*

$$v_{new,f_j} = \mu_{f_j} + \big(D\sigma_{f_j}\big) \tag{4}$$

where D is a random vector that all member is less than 1.

The proposed system integrated a family concept into genetic programming approach. For each generation, the new structure tree can be produced, only a good structure tree will survive. In the same time, the parameter of each family also evolved to find a best set of parameter in each family. The family-based genetic programming is shown in figure 3.

In controlling these two scales of searching methodology, this paper proposes a trade-off strategy to balance the number of structure search and parameter search. The method begins with high number of structure search ratio, then the ratio of the parameter search is increased followed by the number of generation. Both structure and parameter search are applied in each iteration parallely with different ration. With this method, the new structure is discovered in the same time with improvement of the found tree.

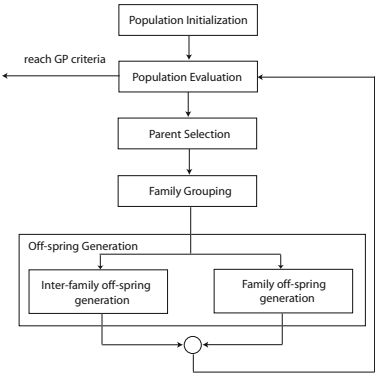


Fig. 3. Family-based Genetic Programming

4 Experimental Evaluation

To evaluate the proposed method, two experiments have been performed. The first experiment compares the classification performance of family-based GP(fGP) with other three methods, grid search(GR), gradient search(GD) and the cascaded of genetic programming and gradient method(gGP) that published in [8]. This experiment evaluated from thirteen binary classes UCI datasets. and Text Classification datasets. In the second experiment, the method is evaluated with text classification dataset consisted of WebKB, 20NewsGroup and Thai Text datasets.

4.1 Evaluation on UCI Datasets

This experiment compares the performance of four kernel selection methods that are grid search, gradient search, cascaded of genetic programming and gradient method and the proposed method. The setting of the experiments is the same as in [8]. The grid and gradient search are searching on a single-RBF kernel function, while the others are multi-kernel function.All results are done with ten-fold cross-validation.

For benchmarking, the proposed method is explored using the thirteen binary-class UCI repository datasets including *Banana, Breast-Cancer, Diabetes, Flare-solar, German, Heart, Image, Ringnorm, Splice, Thyroid, Titanic, Twonorm* and *Waveform*. The data are normalized by zero-mean and unit standard deviation.

In benchmarking with UCI, the average accuracies of ten-fold cross-validation is illustrated in Table 1. In this experiment, the paired t-test is performed for a statistical evaluation the proposed method to the other methods. From the result, the proposed method yields a better performance significantly than GR and GD. Moreover when compare with gGP, fGP still gives a better result from ten datasets out of thirteen datasets.

Table 1. Average accuracy ± standard deviation and statistical test of the thirteen UCI datasets

Dataset	GR(RBF)	GD(RBF)	gGP(Multi)	fGP(Multi)
Banana	$89.32 \pm 0.53^*$	$89.49 \pm 0.46^*$	91.13 ± 1.29	$\mathbf{92.07 \pm 1.21}$
Breast-Cancer	$73.74 \pm 4.99^*$	$74.42 \pm 4.56^*$	$\mathbf{78.63 \pm 1.88}$	79.22 ± 2.43
Diabetes	$76.37 \pm 1.94^*$	$76.75 \pm 3.65^*$	$79.58 \pm 1.05^*$	$\mathbf{81.45 \pm 1.67}$
Flare-solar	$66.79 \pm 2.16^*$	$64.50 \pm 1.79^*$	$68.62 \pm 1.12^*$	$\mathbf{70.32 \pm 1.97}$
German	$75.87 \pm 2.12^*$	$74.88 \pm 2.03^*$	77.57 ± 0.67	$\mathbf{78.94 \pm 1.54}$
Heart	$86.67 \pm 2.83^*$	84.09 ± 3.92	$\mathbf{87.25 \pm 1.91}$	87.04 ± 2.05
Image	$96.74 \pm 0.58^*$	96.25 ± 0.82	$97.03 \pm 0.55^*$	$\mathbf{97.81 \pm 0.53}$
Ringnorm	$98.33 \pm 0.29^*$	98.15 ± 0.16	98.61 ± 0.21	$\mathbf{98.64 \pm 0.29}$
Splice	$89.04 \pm 0.69^*$	$88.95 \pm 0.66^*$	89.94 ± 0.71	$\mathbf{90.21 \pm 0.74}$
Thyroid	$94.92 \pm 2.45^*$	$95.09 \pm 2.18^*$	95.61 ± 0.73	$\mathbf{96.49 \pm 1.41}$
Titanic	$77.42 \pm 0.89^*$	$77.12 \pm 1.53^*$	$79.82 \pm 1.13^*$	$\mathbf{81.25 \pm 1.04}$
Twonorm	$97.42 \pm 0.30^*$	$97.31 \pm 0.27^*$	$98.11 \pm 0.38^*$	$\mathbf{98.52 \pm 0.27}$
Waveform	$91.40 \pm 0.68^*$	$91.73 \pm 1.24^*$	$\mathbf{93.86 \pm 0.84}$	93.42 ± 0.35

* denotes accuracy improvement of Family-based Genetic Programming at $P < 0.1$.

Table 2. Accuracy (%) of RBF kernel and Tree kernel on 20-NewsGroups, WebKB, Thai Medical and Thai News

Dataset	GD(RBF)	gGP(Multi)	fGP(Multi)
20-NewsGroups	88.30	92.67	**92.78**
WebKB1	86.90	90.11	**90.30**
WebKB2	76.84	**83.24**	82.94
Thai Medical	84.75	88.72	**89.32**
Thai News	89.23	91.54	**92.85**

4.2 Evaluation on Text Classification Datasets

In an evaluation with text classification, there are four text datasets are applied. The first two datasets are the well-known English text categorization datasets, WebKB[10] and 20-NewsGroup[11]. The third dataset is a Thai medical text data that contains five topics as follows disease, organ, drug, herb and hospital address. For the forth dataset, Thai News is collected from three newspaper in five topics that are crime, economic, foreign, politic and sport. The number of documents in 20NewsGroup, WebKB, Thai Medical and Thai News are 19997, 8282, 3599 and 1492 respectively. Specific process for Thai text, the word segmentation process is done by using CTTeX program. Unlike binary-class datasets, the performance of the proposed method on these multi-class datasets is performed by the one-against-one technique. For evaluate with multi-class text classification dataset, the average accuracy of five-fold cross-validation also yields a better result four from five datasets compared with other method.

5 Conclusion

This paper introduces family-based genetic programming and a strategy for trading-off between structural search and parameter search. In each of generation, the structure of the population are analyzed then grouping the individual that contain the same structure to be a family. Then, the new generation individuals are constructed with structure crossover, structure mutation and parameter mutation operators. The proposed method a better result compared to other search method both in benchmark datasets and text categorization datasets.

Acknowledgement

This work has been supported by Golden Jubilee Ph.D. Scholarship under Thailand Research Fund (TRF) with the contract No. PHD/0040/2547 and TRF research grant No. BRG50800013.

References

1. Chapelle, O., Vapnik, V., Bousquet, O., Mukherjee, S.: Choosing multiple parameters for support vector machines. Mach. Learn. 46(1-3), 131–159 (2002)
2. Keerthi, S.: Efficient tuning of svm hyperparameters using radius/margin bound and iterative algorithms. IEEE Transactions on Neural Networks 13, 1225–1229 (2002)
3. Lanckriet, G.R.G., Cristianini, N., Bartlett, P., Ghaoui, L.E., Jordan, M.I.: Learning the kernel matrix with semidefinite programming. J. Mach. Learn. Res. 5, 27–72 (2004)
4. Sonnenburg, S., Räatsch, G., Schäafer, C., Schölkopf, B.: Large Scale Multiple Kernel Learning. Journal of Machine Learning Research 7, 1531–1565 (2006)
5. Phienthrakul, T., Kijsirikul, B.: Evolutionary strategies for multi-scale radial basis function kernels in support vector machines. In: Proceedings of the 2005 conference on Genetic and evolutionary computation, pp. 905–911. ACM Press, New York (2005)
6. Sullivan, K.M., Luke, S.: Evolving kernels for support vector machine classification. In: Proceedings of the 9th annual conference on Genetic and evolutionary computation, pp. 1702–1707. ACM, New York (2007)
7. Methasate, I., Theeramunkong, T.: Experiments on kernel tree support vector machines for text categorization. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS, vol. 4426, pp. 720–727. Springer, Heidelberg (2007)
8. Methasate, I., Theeramunkong, T.: Kernel trees for support vector machines. IEICE Transactions 90-D(10), 1550–1556 (2007)
9. Phienthrakul, T., Kijsirikul, B.: GPES: An algorithm for evolving hybrid kernel functions of support vector machines. In: 2007 IEEE Congress on Evolutionary Computation, Singapore, September 25-28, pp. 2636–2643. IEEE Press, Los Alamitos (2007)
10. Ghani, R.: Cmu world wide knowledge base (webkb) project (2001)
11. Lang, K.: Newsweeder: Learning to filter netnews. In: Proceedings of the Twelfth International Conference on Machine Learning, pp. 331–339 (1995)

An Online Incremental Learning Vector Quantization*

Ye Xu[1], Shen Furao[1], Osamu Hasegawa[2], and Jinxi Zhao[1]

[1] The State Key Laboratory for Novel Software Technology, Nanjing University
yexu@smail.nju.edu.cn, {frshen, jxzhao}@nju.edu.cn
[2] Imaging Science and Engineering Lab, Tokyo Institute of Technology
hasegawa@isl.titech.ac.jp

Abstract. As described in this paper, we propose online incremental learning vector quantization (ILVQ) for supervised classification tasks. As a prototype-based classifier, ILVQ needs no prior knowledge of the number of prototypes in the network or their initial value, as do most current prototype-based algorithms. It adopts a threshold-based insertion scheme to determine the number of prototypes needed for each class dynamically according to the distribution of training data. In addition, this insertion policy insures the fulfillment of the incremental learning goal, including both between-class incremental learning and within-class incremental learning. A technique for removing useless prototypes is used to eliminate noise interrupting the input data. Unlike other LVQ-based methods, the learning result won't be affected by the sequence of input patterns that come into the ILVQ. Results of experiments described herein show that the proposed ILVQ can accommodate the non-stationary data environment and can provide good recognition performance and storage efficiency.

1 Introduction

Learning Vector Quantization (LVQ) [1] is a supervised learning technique for nearest prototype classification. During recent years, many variants [2] [3] of LVQ have been proposed. These methods have been used as effective pattern recognition tools in a variety of applications.

However, LVQ and LVQ-based methods have some shortcomings. In the first place, it needs a user to initialize the value of the prototypes. Therefore, the learning performance will be affected by the initial values. What's worse, the online learning can not be indeed realized since many people often initialize them with part of training samples. Additionally, the number of prototypes for every class is arbitrarily predetermined without any valuable prior knowledge. Last but not least, these methods can't handle the incremental data set.

* This work was supported in part by the China NSF grant (#60573157, #60723003, and #60775046) and New Energy and Industrial Technology Development Organization (NEDO) of Japan.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 1046–1053, 2009.
© Springer-Verlag Berlin Heidelberg 2009

As described herein, we propose an autonomous learning method named incremental learning vector quantization (ILVQ) for supervised classification. The goal is to develop a network that can operate autonomously in a non-stationary data environment. We need not predetermine the number or value of prototypes before learning. The proposed ILVQ can grow dynamically, learning the number of prototypes for each class needed to solve a current issue and adjusting the number and value of prototypes during training based on a given application. Actually, ILVQ can realize an incremental learning task in the meanings of within-class incremental learning and between-class incremental learning. Within-class incremental learning means that the network learns new knowledge incrementally within the same class, whereas between-class incremental learning means that it incrementally learns new knowledge that comes from a new incoming class. In other words, ILVQ not only adds prototypes within a class (within-class incremental learning); it also incrementally adds the number of classes during training (between-class incremental learning). Furthermore, ILVQ can eliminate the noise from input data. Consequently, the sequence of input patterns in the training set will impart little influence on the ILVQ performance.

2 Incremental Learning Vector Quantization (ILVQ)

In this section, we introduce the basic idea of the proposed incremental learning vector quantization (ILVQ), and give the detailed algorithm.

Shown in Figure 1 is the overall structure of ILVQ. It's a three-layer network. The first layer is input layer that is used to input patterns $\mathbf{x}$ into the network. The second layer is competitive layer. Prototypes are stored in this layer to conduct competitive learning with each other. The competitive result within a class i is:

$$\mathbf{r}_i = (\|\mathbf{w}_1^i - \mathbf{x}\|, \|\mathbf{w}_2^i - \mathbf{x}\| ..., \|\mathbf{w}_{n_i}^i - \mathbf{x}\|) \tag{1}$$

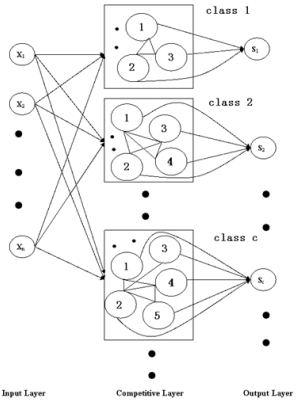


Fig. 1. ILVQ has three layers: input layer, competitive layer and output layer

where n_i is the number of prototypes in class i and $\mathbf{w}_j^i$ is the value of the jth node in class i. The last layer is used to output the result of the network. The competitive result $\mathbf{s}$ that records the winner prototype in each class is

$$\mathbf{s} = (\mathbf{s_1}, \mathbf{s_2}, ..., \mathbf{s_c}) \tag{2}$$

where s_i is the prototype in class i that is nearest to $\mathbf{x}$

$$s_i = \arg \min_{\mathbf{w}_j^i \in classi} r_{ij} = \arg \min_{\mathbf{w}_j^i \in classi} \|\mathbf{w}_j^i - \mathbf{x}\| \tag{3}$$

And we obtain the computed class $label_{\mathbf{x}}$ of the input pattern $\mathbf{x}$

$$label_{\mathbf{x}} = \arg \min_{label_{s_i}} \|\mathbf{w_{s_i}} - \mathbf{x}\| \tag{4}$$

In this network, not only the number of prototypes n_i for each class i but also the class number c are incrementally learned and adjusted. The number and value of prototypes for each class will be reported when the learning is finished. The complete algorithm of ILVQ is given in Algorithm 1.

To fulfill the incremental task, the network should be dynamically growing. Thus we need to sequentially insert proper nodes into the network. However, permanent insertion policy may result in interminable growth of network and overfitting. Therefore, we must decide when and how to insert a new prototype and when insertion is to be stopped. Here, we adopt a threshold method like this: a pattern from new classes that haven't been learned before should be obviously inserted into the network. But for patterns from learned classes, firstly we find the two prototypes "winner" and "runner-up" that are nearest to the input pattern $\mathbf{x}$. Then the insertion operation can be conducted when the distance between $\mathbf{x}$ and $\mathbf{w}_{winner}$ is larger than a certain threshold or distance between $\mathbf{x}$ and $\mathbf{w}_{runnerup}$ is larger than threshold of runner-up. The network may be interrupted by probable noises if we simply take the distance between "winner" and $\mathbf{x}$ into account. So the location of "runner-up" is also considered. This scheme ensures ILVQ to automatically generate proper number of prototypes needed for each class.

However, if the threshold T is too small, all samples will form an isolated prototype. To the contrary, each input pattern will be attributed to only one prototype. To determine T needs some prior knowledge, but we often can not avail them. Furthermore, the threshold should not be constant during training. Otherwise the prototypes in the network may increase endlessly. According to the above discussion, we propose a new method to dynamically compute the threshold distance T in Algorithm 2. We set the average distance between i and other prototypes in the same label with i as i's within-class distance, and set the distance between i and prototypes whose label is different from i's as its between-class distance.

From the discussion in [4], establishing reasonable topological structure is beneficial for us to detect probable noise in the network. We apply the topological representing rule to connect two prototypes when they become the winner and

1. Initialize prototype set G to contain first two input data from the training set, s_1 and s_2.
2. Initialize edge set E storing connection between prototypes to the empty set.
3. Input a new pattern $\xi \in R^n$ to the system.
4. Search set G to find the winner s_1 and runner-up s_2 by $s_1 = argmin_{c \in G} \|\xi - \mathbf{w_c}\|$ and $s_2 = argmin_{c \in G \setminus \{s_1\}} \|\xi - \mathbf{w_c}\|$.
5. **if** $N_{label_\xi} < 2 \bigvee \|\xi - \mathbf{w_{s_1}}\| > T_{s_1} \bigvee \|\xi - \mathbf{w_{s_2}}\| > T_{s_2}$ **then**
6. Insert ξ into set G.
7. Goto Step3 to process the next input pattern.
8. **end if**
9. **if** $(s_1, s_2) \notin E$ **then**
10. Add edge (s_1, s_2) to edge set E and set its age to zero.
11. **end if**
12. **for** s_i in the neighbor area of s_1 **do**
13. Update $age_{(s_1,s_i)} \leftarrow age_{(s_1,s_i)} + 1$
14. **end for**
15. Update winner count $M_{s_1} \leftarrow M_{s_1} + 1$.
16. **if** $label_\xi = label_{s_1}$ **then**
17. Update $\mathbf{w_{s_1}} \leftarrow \mathbf{w_{s_1}} + \eta_1(\xi - \mathbf{w_{s_1}})$
18. **for** s_i in the neighbor area of node s_1 and $label_{s_i} \neq label_\xi$ **do**
19. Update $\mathbf{w_{s_i}} \leftarrow \mathbf{w_{s_i}} - \eta_2(\xi - \mathbf{w_{s_i}})$
20. **end for**
21. **else**
22. Update $\mathbf{w_{s_1}} \leftarrow \mathbf{w_{s_1}} - \eta_1(\xi - \mathbf{w_{s_1}})$
23. **for** s_i in the neighbor area of node s_1 and $label_{s_i} = label_\xi$ **do**
24. Update $\mathbf{w_{s_i}} \leftarrow \mathbf{w_{s_i}} + \eta_2(\xi - \mathbf{w_{s_i}})$
25. **end for**
26. **end if**
27. Delete those edges in set E whose age outstrips the parameter *AgeOld*.
28. **if** the iteration step index is the integer multiple of parameter λ **then**
29. Delete the nodes s_i in set G that have no neighbor node.
30. Delete the nodes s_i whose neighbor node is one and $M_{s_i} < 0.5 \sum_{j=1}^{N_G} \frac{M_j}{N_G}$.
31. **end if**
32. Goto step 3 to continue the online learning process.
33. **return** s et G and set E.

Algorithm 1. Online incremental learning vector quantization

the runner-up of an incoming pattern. In an online environment, the prototypes in the network can not remain their neighbor relationship constantly. The prototypes that are neighboring at an early stage won't be neighboring at a more advanced stage. Thus it is necessary to remove connections which have not been updated recently.

We improve the competitive learning rule to update the value of prototypes. First, we divide the prototypes in the neighbor region of winner into two sets C_1 and C_2 according to their label values. The prototypes that have the same label as $\mathbf{x}$ are put into C_1. And the remaining prototypes are put into C_2. And we update the value of prototypes in this way: when $label_{winner_i} = label_x$, we

1. Compute the within-class distance $T_{i_{within}}$ by $T_{i_{within}} = \frac{1}{N_{label_i}} \sum_{(i,j)\in E \bigwedge label_i = label_j} \|\mathbf{w_i} - \mathbf{w_j}\|$.
2. Find the minimum between-class distance $T_{i_{between}} = \min_{(k_1,i)\in E \bigwedge label_i \neq label_{k_1}} \|\mathbf{w_i} - \mathbf{w_{k_1}}\|$.
3. **if** $T_{i_{between}} < T_{i_{within}}$ **then**
4. Set $T_{i_{between}}$ with the second minimum between-class distance:
5. $T_{i_{between}} = \min_{(k_2,i)\in E \bigwedge label_i \neq label_{k_2} \bigwedge k_1 \neq k_2} \|\mathbf{w_i} - \mathbf{w_{k_2}}\|$.
6. **end if**
7. Go to step2 to update $T_{i_{between}}$ until $T_{i_{between}}$ is no less than $T_{i_{within}}$.
8. Set $T_i = T_{i_{between}}$.
9. **return** T_i

Algorithm 2. Compute the distance threshold T_i of prototype vector i

update winner and prototypes in set C_2. On the contrary, we update winner and those prototypes in C_1. In this case, the directions that these prototypes move are exactly different from the winner: if the winner is moved towards ξ, these prototypes will be moved far from ξ. So the distance between each of the prototype in C_2 and winner will be larger. It is beneficial for us to distinguish those patterns from different classes.

Noise may be interfused in the input data. We need to take some efforts to delete those prototypes in the network that are likely to be noise. Here, we take the strategy in [4]: for every several epochs of learning, we try to remove those nodes with only one or no topological neighbor. In [4], this strategy works well for removing nodes caused by noise without adding computing load.

In Algorithm 1, the learning rate η_1 and η_2 affect the extent the "winner" and its neighbors move towards or far away from the input pattern. In this study, we adopt a method used in [5] to adapt the learning rate over training epoch by $\eta_1 = \frac{1}{M_{winner}}$ and $\eta_2 = \frac{1}{100M_{winner}}$. In this case, the learning rate is time varying and we can make the position of prototype more stable when it becomes a winner for more and more times during training. Due to the divergence of the series $\sum_{n=1}^{\infty} \frac{1}{n}$ and the convergence of $\sum_{n=1}^{\infty} \frac{1}{n^2}$, the learning speed of our network satisfies stochastic approximation conditions.

3 Experiment

3.1 Artificial Data Set

In this section, we conduct our experiment on the two dimensional data set as shown in Fig. 2. Details of the artificial data set can be found in [5]. We apply Fig. 2 to simulate real-world data: 50000 patterns are randomly picked up from all classes, and we add some noise (20% of useful data) into training set. For ILVQ, we set parameters as $\lambda = AgeOld = 16$. During the training, the number and value of prototypes are automatically determined by the algorithm itself. When the training is finished, each class has different number of prototypes:

Table 1. Results for stationary environment

Algorithm	#Prototype	Recognition rate
ILVQ	228	97.5±0.5
LVQ	200	96.7±0.5
LVQ	300	97.0±0.3
LVQ	500	97.3±0.4
G-LVQ	200	96.1±0.2
G-LVQ	300	96.4±0.2
G-LVQ	500	96.5±0.1

Table 2. Results for incremental environment

Algorithm	#Prototype	Recognition rate
ILVQ	291	97.7±0.6
LVQ	200	10.0±0.0
LVQ	300	10.0±0.0
LVQ	500	10.0±0.0
G-LVQ	200	10.0±0.0
G-LVQ	300	10.0±0.0
G-LVQ	500	10.0±0.0

Table 3. Recognition rate and compression ratio of ILVQ under different parameters: Here we always give $AgeOld$ and λ the same value

Parameter	200	300	400	450(optimal)	500	600	700
Recognition rate	97.0	97.2	97.1	97.5	97.4	97.1	97.4
Compression ratio	5.25	6.05	6.07	6.00	6.20	6.57	6.65

class 1, 2, 3, 4 and 5 have 8, 9, 11, 6 and 6 prototypes respectively. It means ILVQ can use fewer prototypes to represent a simple class, and use relative more prototypes to represent a complex class. Such prototypes are connected by edges to represent topological structure of input data, as shown in Fig. 3. We can see not only within-class connections but between-class connections are learned. We highlighted between class connections by black line in Fig. 3.

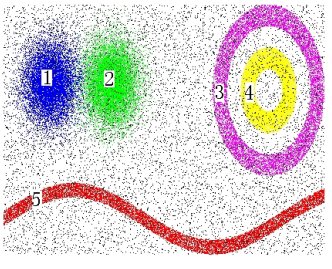


Fig. 2. The artificial data set

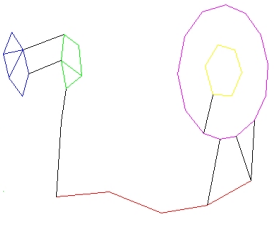


Fig. 3. Topological structure of ILVQ

3.2 Handwritten Digits Recognition

Now we use Optdigits data sets from UCI Machine Learning Repository [6] to test the proposed algorithm. For ILVQ, we apply three times ten-fold cross-validation policy to tune the parameters: $\lambda = AgeOld = 450$. Traditional LVQ and Generalized-Learning Vector Quantization(G-LVQ)[7], an improved version of LVQ, are used to make comparison. For stationary environment and incremental environment, we repeat the algorithms eight times to obtain average results respectively, as shown in Table 1 and Table 2.

Table 4. Recognition rate of experiments: the best and near best performances are highlighted with bold figure

Dataset	ILVQ (Stationary)	ILVQ (Incremental)	LVQ	GLVQ	KMC	NSC	NNC
Glass	**73.3±0.5**	71.4±1.9	68.3±2.0	72.8±0.8	68.8±1.1	70.2±1.5	**72.3±1.2**
Iono	89.5±0.2	89.1±0.2	86.4±0.8	88.6±0.9	87.4±0.6	**91.9±0.8**	86.1±0.7
Iris	**97.1±0.9**	**97.3±0.7**	**96.1±0.6**	**96.7±0.3**	**96.2±0.8**	**96.3±0.4**	**96.7±0.6**
Liver	**67.3±1.3**	**66.6±2.5**	66.3±1.9	**67.4±1.5**	59.3±2.3	62.9±2.3	**67.3±1.6**
Pima	**73.7±1.0**	**73.2±3.3**	**73.5±0.9**	71.1±1.0	58.7±0.9	68.6±1.6	**74.7±0.7**
Wine	73.5±4.1	**73.9±2.5**	72.3±1.5	72.9±4.8	71.9±1.9	**75.3±1.7**	**73.9±1.9**
Average	**79.1±1.3**	**78.6±1.9**	77.2±1.3	**78.3±1.6**	73.7±1.3	77.5±1.4	**78.5±1.1**

Table 5. Parameters and compression ratio (r_c) of experiments

Dataset	ILVQ:Stationary		ILVQ:Incremental		LVQ		GLVQ		KMC		NSC		NNC	
	r_c	(λ,*AgeOld*)	r_c	(λ,*AgeOld*)	r_c	M	r_c	M	r_c	M	r_c	σ^2_{max}	r_c	k
Glass	**25.5**	(786,140)	29.8	(786,140)	45	97	48.7	105	**17**	6	97	0.005	100	1
Iono	25.6	(525,525)	31.7	(530,243)	**6.8**	24	34	120	**4.0**	7	31	1.25	100	2
Iris	19.9	(21,17)	21.9	(66,66)	15	22	22.5	33	**8.0**	4	**7.3**	0.25	100	14
Liver	**6.7**	(16,18)	**4.5**	(5,63)	8.4	29	20.9	72	11	19	**4.9**	600	100	14
Pima	**2.3**	(90,13)	**1.8**	(6,55)	3.4	26	2.6	20	**1.0**	4	**1.7**	2600	100	17
Wine	12	(199,94)	11.3	(124,85)	32	57	10.1	18	29	17	96	4.0	100	1
Average	**15.3**		**16.8**		18.4		23.1		**11.7**		39.7		100	

Table 3 lists recognition rate and compression ratio (the number of prototypes dividing data set size) of ILVQ under different parameters. It means the recognition rate and compression ratio of ILVQ is not sensitive to its parameters. Moreover, we find that in a non-stationary environment LVQ and G-LVQ can only recognize the patterns from the very class (class 10) that are lastly input into the network. The learned patterns (from class 1 to class 9) were destroyed when the LVQ or G-LVQ learned patterns from class 10. But ILVQ can deal with the incremental data environment well because the learned patterns can be always soundly preserved in the network.

3.3 Other UCI Data Sets

Now we sequentially use some other data sets from UCI Machine Learning Repository to test the proposed algorithm. To better evaluate the proposed method, besides LVQ and G-LVQ, we compare it to some other supervised classifiers such as KMC, NNC, and Nearest Subclass Classifier(NSC)[8]. We apply ten-fold cross-validation procedure three times to estimate recognition performance. We list the results of ILVQ in both stationary and incremental environment in Table 4. Because all the above classifiers except NNC can not cope with incremental learning task, we only list the results in stationary environment. The parameters obtained

by ten-fold cross-validation procedure and compression ratio (r_c) are shown in Table 5. We can find that ILVQ realize a best compromise between recognition performance and compression ratio. And the performance of ILVQ in incremental environment is as well as it in stationary environment.

4 Conclusion

In this paper, we propose an online incremental algorithm for supervised learning. The proposed ILVQ applies a threshold-based insertion criterion to grow the network. An innovative method is proposed to adjust the value of the threshold dynamically. Due to the very criterion, the number of prototypes for each class can be generated automatically and adjusted adaptively during learning. Thus users no longer need to predetermine it. ILVQ network can grow gradually and store the learned patterns perfectly so that it can realize the incremental learning well. A proper deletion policy is taken to ensure ILVQ to successfully eliminate the possible noise that comes into the network during learning.

In the experiments, we compare ILVQ with LVQ and other typical prototype-based classifiers. The results show that the recognition rate of ILVQ is better under most conditions. In the respective of compression ratio, the performance of ILVQ is nice as well. Although some algorithms such as KMC and BTS achieve a better compression ratio, we should note ILVQ fulfill the best compromise between classification performance and storage efficiency. Moreover, the experiments demonstrate ILVQ can handle the incremental learning task perfectly.

References

1. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. China Machine Press (2004)
2. De Stefano, C., D'Elia, C., Marcelli, A.: A dynamic approach to learning vector quantization. In: ICPR 2004, vol. 4(4), pp. 601–604 (2004)
3. Pedreira, C.E.: Learning vector quantization with training data selection. IEEE Trans. on PAMI 28(11), 157–162 (2006)
4. Shen, F., Tomotaka, O., Hasegawa, O.: An enhanced self-organizing incremental neural network for online unsupervised learning. Neural Networks 20(8), 893–903 (2007)
5. Shen, F., Hasegawa, O.: An incremental network for on-line unsupervised classification and topology learning. Neural Networks 19, 90–106 (2006)
6. Blake, C.L., Merz, C.J.: Uci repository of machine learning databases. University of California Department of Information, Irvine (1996)
7. Sato, A., Yamada, K.: Generalized learning vector quantization. In: NIPS 1995, pp. 424–429 (1995)
8. Veenman, C.J., Reinders, M.J.T.: The nearest subclass classifier: A compromise between nearest mean and nearest neighbor classifier. IEEE Trans. on PAMI 27(9), 1417–1429 (2005)

On Mining Rating Dependencies in Online Collaborative Rating Networks

Hady W. Lauw[1], Ee-Peng Lim[2], and Ke Wang[3]

[1] School of Computer Engineering, Nanyang Technological University, Singapore
hadylauw@pmail.ntu.edu.sg
[2] School of Information Systems, Singapore Management University, Singapore
eplim@smu.edu.sg
[3] Department of Computing Science, Simon Fraser University, Burnaby, BC, Canada
wangk@cs.sfu.ca

Abstract. The trend of social information processing sees e-commerce and social web applications increasingly relying on user-generated content, such as rating, to determine the quality of objects and to generate recommendations for users. In a rating system, a set of reviewers assign to a set of objects different types of scores based on specific evaluation criteria. In this paper, we seek to determine, for each reviewer and for each object, the dependency between scores on any two given criteria. A reviewer is said to have high dependency between a pair of criteria when his or her rating scores on objects based on the two criteria exhibit strong correlation. On the other hand, an object is said to have high dependency between a pair of criteria when the rating scores it receives on the two criteria exhibit strong correlation. Knowing reviewer dependency and object dependency is useful in various applications including recommendation, customization, and score moderation. We propose a model, called *Interrelated Dependency*, which determines both types of dependency simultaneously, taking into account the interrelatedness between the two types of dependency. We verify the efficacy of this model through experiments on real-life data.

1 Introduction

Rating is one of the key ingredients in social web (as well as many non-web settings). Rating plays an important role in influencing decision making and people's choices. In a rating system, reviewers (users) assign rating scores to objects (products, content items, etc.). Multi-criteria ratings on objects in social web have become very common as users often rate objects based on different evaluation criteria, derived from important object features. For example, to evaluate a digital camera on a product review site, a reviewer may give scores to different camera features (e.g., ease of use, battery life, memory size), in addition to an overall score.

By observing the correlation between criterion-level scores and overall scores (or between scores of two different criteria), we can derive insights about the

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 1054–1061, 2009.
© Springer-Verlag Berlin Heidelberg 2009

dependency behavior of reviewers and objects, which are useful in many applications. For example, a recommender system [1] may want to recommend to a reviewer who exhibit high dependency between the memory size and overall scores a camera that has a large memory size.

Given a set of rating data, we seek to determine the *reviewer dependency* of every reviewer, and the *object dependency* of every object from the rating data.

- *Reviewer Dependency.* A reviewer is said to have high reviewer dependency between a pair of criteria when his or her ratings on various objects based on the two criteria exhibit strong correlation. For example, a camera reviewer may value a camera's memory size so much so that s/he assigns high overall scores to cameras with high memory size scores (also assigned by the reviewer), and low overall scores to cameras with low memory size scores.
- *Object Dependency.* For an object, its object dependency represents the extent to which the object shows correlation in ratings between the two criteria by various reviewers. For example, a well-written conference paper may introduce an idea that is so unique that its reviewers either like it due to the idea being novel (high novelty and overall scores), or dislike it as the idea does not look novel to them (low novelty and overall scores).

A *Naive* approach to determine reviewer dependency and object dependency is to simply apply a standard correlation measure, such as Pearson correlation [2]. For instance, in this approach, a reviewer dependency is equated to the correlation between her scores on the first criterion and her scores on the second criterion. However, this approach fails to recognize the relationship between reviewer dependency and object dependency.

Highly correlated rating scores may be due to reviewer dependency or object dependency. For example, *Naive* may conclude that a reviewer places a premium on a camera's memory size if her memory size scores correlate with her overall scores (*high reviewer dependency*). However, this would be justified only if the other reviewers of the camera do not show similar correlations (*low object dependency*). Otherwise, the correct conclusion should be that memory size is a dominant selling feature of this camera such that any reviewer is bound to judge it mainly by its memory size (*high object dependency*).

Interrelated Dependency Principle. Therefore, we propose the following principle relating reviewer dependency and object dependency.

- A reviewer has *high reviewer dependency* on a pair of evaluation criteria when his or her rating scores on various objects based on the two criteria exhibit *high correlation*, and these rated objects exhibit *low object dependency* on the same two criteria.
- An object has *high object dependency* on a pair of evaluation criteria when the rating scores it receives from various reviewers based on the two criteria exhibit *high correlation*, and these reviewers exhibit *low reviewer dependency* on the same two criteria.

The rest of the paper is organized as follows. In Section 2, we describe our proposed *Interrelated Dependency* (or *ID*) model to determine reviewer

dependency and object dependency between two given criteria. In Section 3, we present experimental results on a real-life dataset. These are followed by an overview of related work in Section 4 and conclusion in Section 5.

2 Interrelated Dependency (ID)Model

Before describing the *Interrelated Dependency* (or *ID*) model, we first review the notations to be used. We model a rating system as a bipartite network with reviewers and objects forming the two distinct sets of entities. A reviewer r_i may assign to an object o_j two rating scores $a_{ij},\ b_{ij} \in [0,1]$ based on two different criteria a and b. Without any loss of generality, the overall score is treated as a criterion. For each r_i, we want to determine r_i's **reviewer dependency** between a and b, denoted by $d_{r_i}(a,b) \in [0,1]$. For each o_j, we want to determine o_j's **object dependency** between a and b, denoted by $d_{o_j}(a,b) \in [0,1]$. When a and b are implicit, we may further simplify the reviewer dependency and object dependency notations as d_{r_i} and d_{o_j} respectively.

Our proposed *ID* model consists of a pair of equations: Equation 1 to determine reviewer dependency d_{r_i} and Equation 2 to determine object dependency d_{o_j}. To determine d_{r_i} with Equation 1, we compute the correlation observed on r_i's scores $[\mathcal{F}(\mathbf{a}_{r_i}, \mathbf{b}_{r_i})]$, and reduce it by the aggregate object dependency of r_i's objects $[\mathcal{A}gg_j\ (1-d_{o_j})]$. $\mathcal{F}$ is a correlation measure, which will be described later in this section. $\mathbf{a}_{r_i}$ and $\mathbf{b}_{r_i}$ are the vectors of r_i's a_{ij} and b_{ij} scores across different o_j's rated by r_i. $\mathcal{F}(\mathbf{a}_{r_i}, \mathbf{b}_{r_i})$ is higher when there is greater correlation between r_i's a_{ij} and b_{ij} scores. $\mathcal{A}gg$ represents a function to aggregate $(1-d_{o_j})$ across various o_j's. $\mathcal{A}gg_j\ (1-d_{o_j})$ is higher when r_i's rated objects generally have low dependency. Thus, a reviewer has high d_{r_i} if she assigns highly correlated scores on objects with low d_{o_j}.

$$d_{r_i} = \mathcal{F}(\mathbf{a}_{r_i}, \mathbf{b}_{r_i}) \times \underset{j}{\mathcal{A}gg}\ (1 - d_{o_j}) \tag{1}$$

$$d_{o_j} = \mathcal{F}(\mathbf{a}_{o_j}, \mathbf{b}_{o_j}) \times \underset{i}{\mathcal{A}gg}\ (1 - d_{r_i}) \tag{2}$$

Equation 2 to determine object dependency is symmetrical to Equation 1. An object has high d_{o_j} if it exhibits highly correlated scores by reviewers with low d_{r_i}. In both Equations 1 and 2, we require $d_{r_i},\ d_{o_j} \in [0,1]$ as well as $\mathcal{F}(\mathbf{a}_{r_i}, \mathbf{b}_{r_i}),\ \mathcal{F}(\mathbf{a}_{o_j}, \mathbf{b}_{o_j}) \in [0,1]$.

In this paper, we use the *average* function for $\mathcal{A}gg$, which is suitable as it takes into account d_{o_j} of all the o_j's concerned. Another possible function is *median*. Examples of unsuitable functions include *summation*, which unfairly penalizes active reviewers (with many objects) and actively-rated objects (with many reviewers); *minimum*, which unfairly penalizes occasional dependency; and *maximum*, which unfairly rewards occasional lack of dependency.

The above equations are based on the principle of interrelated dependency between d_{r_i} and d_{o_j}. To determine d_{r_i}, we need to know d_{o_j} (vice versa). Moreover, d_{r_i} and d_{o_j} are inversely related, with d_{r_i} higher when d_{o_j} is lower. The reviewer

dependency of various reviewers are related by co-rating common objects, and the object dependency of various objects are related by having common reviewers. As reviewers and objects are inter-connected within the rating data, the interrelatedness extends to all reviewers and objects, which means all d_{r_i}'s and d_{o_j}'s need to be solved simultaneously. Given *average* as the $\mathcal{A}gg$ function, the set of Equation 1 for each r_i and Equation 2 for each o_j forms a system of linear equations. Such systems can be solved using either linear algebra or iterative methods, provided the system is uniquely determined [3].

To determine reviewer dependency, we rely more on objects with low dependency. Ideally we should have as many low dependency objects per reviewer as possible. Given that low dependency objects are usually the majority, a reviewer will likely rate more low dependency objects when s/he has rated many objects. Correlation measure is also generally more meaningful for larger "sample" size.

Correlation Measure. $\mathcal{F}$ measures the correlation between two sets of scores. The measure we use in this paper is a slightly modified version of the *Pearson product-moment correlation coefficient* [2]. For two vectors $\mathbf{a}$ and $\mathbf{b}$ of N real-valued elements each, the correlation coefficient $\mathcal{F}(\mathbf{a}, \mathbf{b}) \in [0, 1]$ is determined by Equation 3. $\mathbf{a}[n]$ is the n^{th} element of $\mathbf{a}$, while $\mu_{\mathbf{a}}$ and $\sigma_{\mathbf{a}}$ are the mean and the standard deviation of $\mathbf{a}$'s elements respectively. Similar notations apply to $\mathbf{b}$. Higher $\mathcal{F}(\mathbf{a}, \mathbf{b})$ value indicates greater correlation between $\mathbf{a}$ and $\mathbf{b}$. This measure is also symmetric, i.e., $\mathcal{F}(\mathbf{a}, \mathbf{b}) = \mathcal{F}(\mathbf{b}, \mathbf{a})$.

$$\mathcal{F}(\mathbf{a}, \mathbf{b}) = \left| \frac{\sum_n^N (\mathbf{a}[n] - \mu_{\mathbf{a}})(\mathbf{b}[n] - \mu_{\mathbf{b}})}{(N-1) \times \sigma_{\mathbf{a}} \times \sigma_{\mathbf{b}}} \right| \tag{3}$$

This measure is slightly different from the original *Pearson correlation*. While *Pearson correlation* ranges from -1 to 1, Equation 3 ignores the sign, thus confining the range to 0 to 1. Secondly, *Pearson correlation* is undefined for $\sigma_{\mathbf{a}} = 0$ or $\sigma_{\mathbf{b}} = 0$. We choose to define $\mathcal{F}(\mathbf{a}, \mathbf{b}) = 0$ for these cases. $\sigma_{\mathbf{a}} = 0$ or $\sigma_{\mathbf{b}} = 0$ happens when either $\mathbf{a}$ or $\mathbf{b}$ has uniform elements, in which case the two vectors can reasonably be considered independent. $\mathcal{F}(\mathbf{a}, \mathbf{b})$ is high when elements of $\mathbf{a}$ and $\mathbf{b}$ rise and fall together, and low when they do not vary or vary independently.

Other possible correlation measures include mutual information [4], or rank correlations [5]. We will study these alternatives as part of future work.

Naive Model as a Special Case. A suitable baseline alternative to the *ID* model should be the basic model which does not incorporate the interrelated dependency between d_{r_i} and d_{o_j}. This baseline model, called the *Naive* model, consists of the pair of Equation 4 to determine d_{r_i} and Equation 5 to determine d_{o_j}. *Naive*'s equations compute d_{r_i} and d_{o_j} respectively using only the correlation measure $\mathcal{F}(\mathbf{a}, \mathbf{b})$. Given *average* as the $\mathcal{A}gg$ function, *ID* would degenerate into *Naive* when all d_{o_j}'s (or d_{r_i}'s) are uniform, in which case *ID*'s Equation 1 (or Equation 2) practically degenerates into *Naive*'s Equation 4 (or Equation 5).

$$d_{r_i} = \mathcal{F}(\mathbf{a}_{r_i}, \mathbf{b}_{r_i}) \tag{4}$$

$$d_{o_j} = \mathcal{F}(\mathbf{a}_{o_j}, \mathbf{b}_{o_j}) \tag{5}$$

3 Experiments

Our experimental objective is to verify the effectiveness of the proposed model on a real-life dataset. First, we compare the reviewer dependency and object dependency ranks produced by the *Naive* and *ID* models. Next, we highlight a specific case example that illustrated how the contribution of interrelated dependency principle that makes the *ID* model more effective.

3.1 Dataset

Our dataset was obtained from *Epinions*, a product review Web site. We crawled the site over three days (April 20-23, 2007), beginning from a seed page[1]. The collected dataset consisted of a subset of all products, reviewers, and scores.

In this dataset, each product belongs to a category and each category has its specific rating criteria. Each reviewer may assign one or more scores based on these criteria. These scores, which ranged from 1 to 5, were normalized to a range from 0.2 to 1 by a simple division by 5. We chose to focus on the *Beers* category, as within the collected dataset, this category was relatively large and had more active reviewers and more actively-rated objects. There are four rating criteria for *Beers* category: *Overall*, *Weight*, *Flavor*, and *Complexity*. In the following experiments, we mainly focus on the (*Overall*, *Weight*) pairing. We have also separately carried out other experiments with (*Overall*, *Flavor*) and (*Overall*, *Complexity*) pairings with similar results. For the (*Overall*, *Weight*) pairing, the original data size collected is shown in the first row of Table 1.

Table 1. Data Size

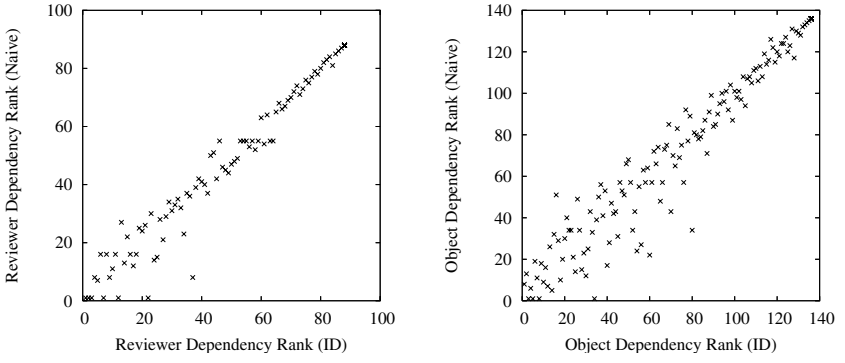
	Reviewers	*Objects*	*Overall Scores*	*Weight Scores*
Original	781	428	2288	2087
Filtered	102	184	1163	1163

We further filtered the dataset to make it more suitable for experiments. First, we considered a reviewer as having rated an object only if the reviewer had assigned both *Overall* and *Weight* scores. Second, we ensured that each reviewer must have at least 3 objects and each object must have at least 3 reviewers, by iteratively removing reviewers and objects not meeting the condition until the condition was met. The final data size is shown in the second row of Table 1.

3.2 Rank Comparison

This part of the experiment compares the ranked lists produced by *Naive* and *ID* for the *(Overall, Weight)* pairing. We construct a ranked list of reviewers and a ranked list of objects for each solution. Reviewers and objects are ranked in decreasing order of d_{r_i} and d_{o_j} respectively. The highest d_{r_i} or d_{o_j} value is given rank 1. Same values share the same rank.

[1] `http://www.epinions.com/rest-Restaurants-All/ViewAll_~1/Grp_~14529`



(a) Reviewer Dependency (b) Object Dependency

Fig. 1. Rank Scatterplots: *ID* vs. *Naive*

Figure 1(a) is a scatterplot of reviewer dependency ranks. A point in each scatterplot represents a reviewer, with the x-value being the rank assigned by *ID* and the y-value the rank assigned by *Naive*. The reviewer dependency ranks assigned by *ID* and *Naive* are positively correlated in general, but are not identical. The positive correlation is expected as both are based on the same $\mathcal{F}$ function. However, *ID* also takes into account object dependency, resulting in a different ranking. Figure 1(a) shows large variances around the diagonal, indicating that *ID* and *Naive*'s rankings are quite different. For example, there are instances where a significant number of reviewers are tied according to *Naive*, but are differentiated by *ID*. In Figure 1(a), 6 reviewers share rank 1 by *Naive*, but are given ranks ranging from 1 to 22 by *ID*.

Figure 1(b) is the corresponding scatterplot for object dependency ranks. As this figure shows much resemblance to the earlier Figure 1(a), similar observations as for reviewer dependency ranks can also be made for object dependency ranks. This itself is an important observation, for it highlights the interrelatedness between reviewer dependency and object dependency. If *ID*'s object dependency ranks were identical to *Naive*'s, so would the reviewer dependency ranks be. Instead, the variance in reviewer dependency ranks results from the variance in object dependency ranks (and vice versa).

3.3 Case Example

Object *boulevard_dry_stout*'s profile is given in Table 2(a). *Naive* assigns this object high object dependency of 1.00 (rank 1). Meanwhile, *ID* assigns it much lower object dependency of 0.33 (rank 34). The rating scores assigned by its three reviewers indeed suggest high object dependency. However, Table 2(a) also shows that *boulevard_dry_stout*'s three reviewers (*lafeet*, *wingdman*, *impydykiechick*) all have high reviewer dependency. For instance, *wingdman* has high reviewer dependency of 0.71 (rank 17). We take a more detailed look at *wingdman*'s rating scores in Table 2(b). Visual inspection reveals a trend whereby *wingdman*'s

Table 2. Case Example

(a) Profile of Object *boulevard_dry_stout*

Object			d_{o_j} (rank)	
			Naive	*ID*
boulevard_dry_stout			1.00 (1)	0.33 (34)
Reviewers	*Score*		d_{r_i} (rank)	
	Ove	*Wei*	*Naive*	*ID*
lafeet	1.0	1.0	0.83 (25)	0.69 (19)
wingdman	1.0	1.0	0.91 (12)	0.71 (17)
impydykiechick	0.8	0.8	0.84 (23)	0.60 (34)

(b) Objects reviewed by *wingdman*

Objects	*Score*	
	Ove	*Wei*
beamish_irish_stout	1.0	1.0
boulevard_bully_porter	1.0	1.0
boulevard_dry_stout	1.0	1.0
dixie_blackened_voodoo_lager	1.0	1.0
sierra_nevada_stout	1.0	1.0
grolsch_premium_lager	0.8	0.6
hacker-pschorr_weisse	0.6	0.6
murphy_s_irish_amber	0.6	0.6
samuel_adams_summer_ale	0.6	0.2
tequiza_beer	0.6	0.2

Overall (*Ove*) score is high (or low) when his Weight (*Wei*) score is high (or low). Given the high reviewer dependency of its reviewers, it is expected that *boulevard_dry_stout* would show high correlation between its *Ove* and *Wei* scores. However, this correlation should be attributed more to its reviewers. Thus, *ID* justifiably assigns lower object dependency (and ranks) to *boulevard_dry_stout*.

4 Related Work

The work on *systematic* dependency uses correlation analysis [6] to test the dependency between a given factor and the assigned scores, which affect all reviewers and objects in general. For example, [7] investigates whether reviewers' scores on peer review submissions are dependent on the reviewers' also being authors; [8] studies whether students tend to assign higher scores to instructors who have given them higher grades; [9] shows that venture capitalists' evaluation of start-up teams is dependent on their similarity to these teams. In contrast, we are concerned with measuring the *specific* dependency of individual reviewers and objects, as reviewers and objects may exhibit varying dependency with respect to two given rating criteria.

The problem of multiple criteria decision making (MCDM) [10] is concerned with making an optimal decision, in view of two or more potentially conflicting criteria. The optimality of a decision necessarily depends on the preferences of the decision-maker. For example, a user may wish to select among several hotels one that scores high on both *luxury* and *affordability*. This may involve comparing in some way the scores assigned to an object on various criteria in aggregate. This problem is different from our problem in that the optimal decision is not necessarily the object with the highest or lowest dependency.

5 Conclusion

In this paper, we address the problem of determining reviewer dependency and object dependency with respect to two rating criteria. The key principle in our *ID* model is modeling the interrelatedness between reviewer dependency and object

dependency, which is ignored by the *Naive* model. Experiments on real-life data show that our approach is more effective than *Naive*. Moreover, although our modeling is based on two given rating criteria, without loss of generality, it can be applied in a pairwise manner for multiple criteria, for instance in learning of the dominant rating criteria affecting the overall scores.

Acknowledgment

This work was supported by A*STAR Public Sector R&D, Singapore, Project Number 062 101 0031.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17(6), 734–749 (2005)
2. Walpole, R.E., Myers, R.H., Myers, S.L., Ye, K.: Probability & Statistics for Engineers & Scientists, 7th edn. Prentice-Hall, Inc., Englewood Cliffs (2002)
3. Anton, H., Rorres, C.: Elementary Linear Algebra with Applications. John Wiley & Sons, Inc., Chichester (1987)
4. Dionisio, A., Menezes, R., Mendes, D.A.: Mutual information: A measure of dependency for nonlinear time series. Physica A 344, 326–329 (2004)
5. Fagin, R., Kumar, R., Sivakumar, D.: Comparing top k lists. SIAM Journal on Discrete Mathematics 17(1), 134–160 (2003)
6. Cohen, J., Cohen, P., West, S.G., Aiken, L.S.: Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences, 3rd edn. Lawrence Erlbaum Associates, Inc., Mahwah (2003)
7. Blackburn, J.L., Hakel, M.D.: An examination of sources of peer-review bias. Psychological Science 17(5), 378–382 (2006)
8. Greenwald, A.G., Gillmore, G.M.: Grading leniency is a removable contaminant of student ratings. American Psychologist 52(11), 1209–1217 (1997)
9. Franke, N., Gruber, M., Harhoff, D., Henkel, J.: What you are is what you like – similarity biases in venture capitalists' evaluations of start-up teams. Journal of Business Venturing 21(6), 802–826 (2006)
10. Figueira, J., Greco, S., Ehrgott, M. (eds.): Multiple Criteria Decision Analysis: State of the Art Surveys. Springer Science+Business Media, Inc. (2005)

Learning to Extract Relations for Relational Classification

Steffen Rendle, Christine Preisach, and Lars Schmidt-Thieme

Machine Learning Lab, University of Hildesheim, Germany
{srendle,preisach,schmidt-thieme}@ismll.uni-hildesheim.de

Abstract. Relational classifiers use relations between objects to predict the class values. In some cases the relations are explicitly given. In other cases the dataset contains implicit relations, e.g. the relation is hidden inside of noisy attribute values. To apply relational classifiers for this task, the relations have to be extracted. Manually extracting relations by a domain expert is an expensive and time consuming task. In this paper we show how extracting relations in datasets with noisy attribute values can be learned. Our method LRE uses a regression model to learn and predict weighted binary relations. We show that LRE is able to extract both equivalence relations and non-constrained relations. Secondly we show that relational classifiers using relations automatically extracted by LRE achieve comparable classification quality as classifiers using manually labeled relations.

1 Introduction

Relational classifiers use relations between instances to predict class values ('entity classification task'). An example is classifying papers by categories using relations like *CommonAuthor*, *CommonCitations* or *SameConference*, where the relations indicate the number of co-authors, co-citations or the same conference given two papers. For problems where such relations are present, relational classifier have shown to outperform state-of-the-art non-relational classifiers like Support Vector Machines [1,2,3]. The drawback of relational classifiers is that they can only be applied in scenarios where the relations are explicitly given. In most real-world scenarios this is not the case. If the relations are not directly given, they have to be annotated manually by domain experts. This is a time consuming and expensive task.

In this paper we propose the method LRE ('Learning Relation Extraction') for automatically extracting relations from a noisy dataset. The method is based on learning a binary regression model for pairs of instances. As the method uses training data, it can adapt itself to different tasks and domains. This makes it superior to handcrafted or heuristic extraction rules.

The overall contributions of our paper are as following: (1) We propose the supervised method LRE for extracting relations. These relations can be used in relational classifiers for the task of entity classification. (2) Our evaluation indicates that learning the relation with our LRE method is successful and results in a comparable performance to manually labeled relations.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 1062–1071, 2009.
© Springer-Verlag Berlin Heidelberg 2009

2 Related Work

Our LRE approach has parallels to the field of record linkage [4], where an equivalence relation should be learned. We share the feature extraction step with work in this field. Here also several heuristic pairwise similarity measures are used to create pairwise features [5,6]. In other components, our approach differs from the work in record linkage as the properties of the relation to predict differs. In record linkage an equivalence relation has to be learned. Thus typically a probabilistic classifier is learned over pairs [5]. Afterwards clustering is performed to create a binary equivalence relation. This approach has been suggested by Preisach et al. [7] to extract relations for relational learning. In contrast to this we learn a regression model to predict weighted relations.

3 Problem Setting

Throughout the paper we use a bibliographic scenario for illustration and evaluation. In this scenario a data set of research papers is given and the category of the paper, e.g. *Machine Learning* or *Artificial Intelligence* should be predicted. We will denote the set of objects – here papers – by $X = \{x_1, \ldots, x_n\}$ and a relation by R. An example of such a data set is given in table 1. Here each paper is described by the attributes *Author*, *Paper title* and *Conference*.

The field values of each attribute can be seen as a relation over a pair of objects – here over two papers. For example the *Conference* attribute values of p_2 and p_3 in table 1 are identical and thus both paper can be linked over a boolean relation SameConference. Similarly, p_1 and p_2 have one author in common and thus CommonAuthors$(p_1, p_2) = 1$. Having such a data set like in figure 1, the relations SameConference and CommonAuthors can easily be constructed. The values of the two binary relations of the given example can be found in table 2. With these two relations any relational classifier can be applied to predict the paper category.

3.1 Relational Classification

A probabilistic relational classifier $P(c|x)$ is a function to predict a class value $c \in C$ for an object $x \in X$ given j relations $R_1, \ldots, R_j : X^2 \to T$. Where T is the range of the relation. We will deal with cases of $T \subseteq \mathbb{R}$.

Table 1. Data from the domain of paper topic classification

ID	Author	Paper title	Conference
p_1	Yoav Freund	Boosting a weak learning algorithm by majority.	COLT
p_2	Yoav Freund, H. Sebastian Seung, Eli Shamir, Naftali Tishby	Prediction, and query by committee	NIPS
p_3	S Rosset, E Segal	Boosting density estimation	NIPS

Table 2. Two binary relations over the papers of table 1. CommonAuthors is a weighted relation, SameConference is a boolean relation.

CommonAuthors			
ID	p_1	p_2	p_3
p_1	1	1	0
p_2	1	4	0
p_3	0	0	2

SameConference			
ID	p_1	p_2	p_3
p_1	true	false	false
p_2	false	true	true
p_3	false	true	true

Table 3. Data sample of table 1 with noise in the Conference and Author attributes

ID	Author	Paper title	Conference
p_1	Freund, Y.	Boosting a weak learning algorithm by majority.	3rd annual workshop on computational learning theory
p_2	Yoav Freund, H. Sebastian Seung, E Shamir, and Naftali Tishby	Prediction, and query by committee	advances in neural information processing systems
p_3	S Rosset, E Segal	Boosting density estimation	NIPS 2003

There are many approaches for relational classifiers. One example is the *probabilistic relational classifier (PRN)* introduced by Macskassy and Provost [2]. It calculates the class probability as the weighted arithmetic mean of the class-membership probabilities of the related objects given a relation R:

$$P(c|x) = \frac{1}{Z} \cdot \sum_{x' \in X} R(x, x') \cdot P(c|x') \tag{1}$$

Where $Z := \sum_{x' \in X} R(x, x')$ is the normalization constant.

Preisach and Schmidt-Thieme [8] have introduced extensions of this algorithm. One extension is not only to take into account direct neighbors but also indirectly linked objects by longer paths. As R is often very sparse, this can be seen as a densification of the relation graph. The model equation of PRN (see (1)) is for a single relation R. PRN can be extended [8] to a multi-relational classifier EPRN by combining several single-relational PRN classifiers by ensembling techniques, e.g. stacking.

3.2 Noisy Data Sets

Relational classifiers are known to result in high quality predictions. However real world data often does not state the relations explicitly. Instead the data set might look like in table 3. Here for example the identity between '*NIPS 2003*' and '*advances in neural information processing systems*' is not obvious. Thus a relational model cannot be run directly on such noisy data sets.

There are many reasons why relations in data sets are not stated explicitly but are hidden. One obvious reason are errors that occur when humans work with an information system, e.g. typos. Secondly, often several abbreviations can be used, e.g. NIPS stands for ***N**eural **I**nformation **P**rocessing **S**ystems*. When writing names the first name can be placed after the family name – e.g. *John Smith* is *Smith, John*. Other places for noise in textual encoded relations are ambiguities – e.g. there are two different people with the same name.

This paper deals with the problem of extracting relations from such noisy data sets. We use an adaptive, supervised method that learns to extract relations given some training data.

3.3 Type of Relations

There are several different types of relations that can be extracted. They can be distinguished by the range T of their target. Important ranges are:

- **Boolean relations** where $T = \{0, 1\}$, e.g SameConference or SamePublisher.
- **Probabilistic relations** where the target is a probability score and thus the range is $T = [0, 1]$.
- **Weighted relations** where a real-valued score is attached to each pair: $T \subseteq \mathbb{R}$, e.g. CommonAuthors.

Our learning method for relations will handle the most general case of weighted relations.

4 Learning Weighted Relations

In the following we describe our method LRE for **L**earning **R**elation **E**xtraction from a noisy data set like in table 3. Our proposed method extracts pairwise features given two objects. Based on these features, a regression model for R is learned and applied to predict $\hat{R}$. The overall algorithm can be found in algorithm 1.

Next we describe the main components of LRE in detail, i.e. extracting pairwise features, learning the regression model and generating candidates for scaling to large data sets.

4.1 Extracting Pairwise Features

The overall goal of our method is to extract relations from noisy attribute values of a data set. Let $a_1, \ldots, a_k : X \to V$ be the noisy attributes of our dataset. E.g. a_1 might return the noisy list of the authors of a specific paper, a_2 the conference string, etc.

As the target relation is defined over a pair of objects, feature extraction aims to create meaningful pairwise information from the attributes. Thus feature

Algorithm 1. LRE: Learning Relation Extraction

procedure LEARNRELATION(X, X_{tr}, R_{tr})
outputs a relation $\hat{R}$ for X

$\forall (x,y) \in \left(X^2 \cup X_{\text{tr}}^2\right) : f(x,y) \leftarrow (f_1(x,y), \ldots, f_l(x,y))$ ▷ extract features
Training:
$C \leftarrow$ GENERATECANDIDATES(X_{tr}) ▷ generate candidate pairs
$D_{\text{tr}} \leftarrow \{(f(x,y), R_{\text{tr}}(x,y)) | (x,y) \in C\}$ ▷ create training-set
TRAIN(D_{tr}) ▷ training the regression model
Prediction:
$\forall (x,y) \in X^2 : \hat{R}(x,y) \leftarrow$ PREDICT($f(x,y)$)

return $\hat{R}$
end procedure

Table 4. Features for learning the relation COMMONAUTHORS from the noisy data of table 3 by using two heuristic pairwise similarity functions

PAIR	OVERLAP(AUTHOR)	TFIDF-SIM(AUTHOR)	TARGET VALUE
(p_1, p_2)	0.5	0.2	1
(p_1, p_3)	0.0	0.0	0
(p_2, p_3)	0.25	0.03	0

extraction can be formulated as a function that generates a real valued feature vector for a pair of objects:

$$f : X^2 \rightarrow \mathbb{R}^l \tag{2}$$

To create a real-valued vector of length l from a pair of objects, several heuristic similarity measures $f_1, \ldots, f_l$ can be applied. Examples of well-known similarity/ distance functions for strings are TFIDF cosine similarity, Overlap index or Levenshtein distance. Details about these and many more similarity measures can be found in [9]. Table 4 shows an example for feature extraction for the COMMONAUTHORS relation.

4.2 Relation Learning

Based on the pairwise features, a regression model is learned to estimate the relation weight between a pair of objects. The model is trained on a labeled subset of objects X_{tr}, where the true relation weight $R(x,y)$ between object pairs $(x,y) \in X_{\text{tr}}^2$ is known. The features of a pair of objects for training the model and later on to predict the unknown weights consist of the elements of the feature vector $f = (f_1, \ldots, f_l)$.

In our implementation we applied a Support Vector Regression (SVR) model using the implementation of libSVM [10]. In general, any other regression method could also be used, e.g. linear regression or ridge regression.

4.3 Scaling

The method proposed so far has a runtime that is quadratic $O(n^2)$ in the number of objects X because the binary relation R has $|X^2|$ possible entries. For large data sets this method is not applicable any more. A solution to the problem relies on the fact that most relations are very sparse, i.e. have many 0 entries. We suggest to use blocking techniques from the field of record linkage to handle such cases. Blocking is a method to reduce the number of pairs, by discarding pairs that are obviously 0. For relations that are equivalence relations any of the proposed blocking techniques from the field of record linkage can be used for GenerateCandidates (see algorithm 1). An overview about blocking for record linkage is given by Baxter et al. [11]. For scaling to the problem of predicting equivalence classes that have both big and many classes, the method proposed by Rendle and Schmidt-Thieme [12] can be applied.

In the general case of predicting an arbitrary weighted relation, we suggest to generate candidates by considering only pairs of objects whose similarity based on a cheap measure exceeds a certain threshold.

5 Evaluation

In our evaluation we investigate if our method successfully learns relations for relational classification. We measure the success in two tasks:

1. **Relation extraction task**: This task measures how good the extracted relation $\hat{R}$ approximates the true (manually) labeled relation R in terms of RMSE over a set S of pairs:

$$E_{\text{RMSE}}(S) = \sqrt{\frac{1}{|S|} \sum_{(x,y)\in S} \left(R(x,y) - \hat{R}(x,y)\right)^2} \tag{3}$$

2. **Entity classification task**: Here the quality of a relation R is measured in terms of how a relational classifier c performs by using the relation. As quality measure we use the classification accuracy:

$$Q^R_{\text{Acc}}(X) = \frac{|\{x \in X : c_R(x) = y_x\}|}{|X|} \tag{4}$$

   Where y_x is the true class label for x and $c_R(x)$ is the prediction of the relational classifier for x using the relation R. We use this measure to compare if a learned relation $\hat{R}$ leads to comparable results as the manually labeled relation R. This means we compare $Q^{\hat{R}}_{\text{Acc}}(X)$ to $Q^R_{\text{Acc}}(X)$.

In all, the entity classification task is the more important task and the goal is to achieve a classification quality with a learned relation as high as with a manually labeled one.

5.1 Dataset

We perform our evaluation on the relational Cora dataset provided by McCallum[1]. This dataset contains papers with their category, author string, citations, conference and journal string. We use a subset with the 12 subcategories of 'Artificial Intelligence', i.e. 'Machine Learning', 'Knowledge Representation', 'Data Mining', etc. We removed papers without authors, citations and neither conference nor journal. In total this subset has 3298 papers.

In the Cora dataset, for the author and citation relation the true relation is already present. For authors the noisy author string is also present. Given this string we try to learn the true author relation. Secondly, we merged the noisy conference and journal attribute. For this merged attribute, we manually labeled a new relation SAMECONFERENCE which states if two papers were published in the same conference proceedings or were published in the same journal. This relation is an equivalence relation. Again in our experiments we try to learn this relation from the noisy string.

5.2 Model Setup

We choose a support vector regression as regression model. The hyperparameters are optimized in each repetition via grid search using cross-validation. As similarity functions we use TFIDF, relative overlap in tokens and absolute overlap in tokens. We use different variants of tokenizers that extract words, 2-grams and 3-grams. Each similarity function is run with each tokenizer. For speedup we use a canopy blocker for the equivalence relation SAMECONFERENCE; for COMMONAUTHOR we use the blocking method described in section 4.3. For relational classification we use an ensemble of PRN models. The ensemble is created by stacking with an SVM as meta classifier [8].

5.3 Learning Relations

In the first experiment we split the dataset X in two disjoint parts of equal size X_{train} and X_{test} with $X_{\text{train}} \cup X_{\text{test}} = X$. We assume that the relations between elements in X_{train} are known – i.e. the relation weights correspond to the manually labeled weights. We then try to predict with LRE all other relation weights, i.e. $\hat{R}$ on $(X \setminus X_{\text{train}})^2$. The results of this relation extraction task is shown in figure 1. We compare the error of LRE to a constant baseline that predicts constantly a weight of 0. As the relations are very sparse – i.e. the weight of most relation pairs is 0 – the constant method results in a low RMSE error. Compared to the constant method the relation extracted by LRE results in an even lower RMSE. Eventhough the relation predicted by the constant method has a low RMSE, this relation is useless for the entity classification task as no objects are linked. With such a relation the relational classifier can only predict the majority class as the relation contains no information. In contrast to this the relations extracted by LRE are also successful in solving the entity classification task.

[1] http://www.cs.umass.edu/∼mccallum/data/cora-classify.tar.gz

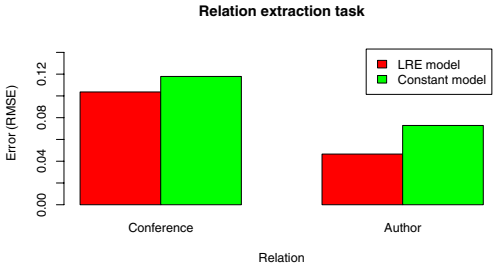


Fig. 1. Error (RMSE) between the true and the predicted relation

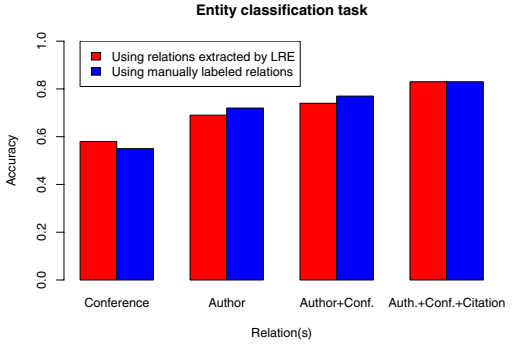


Fig. 2. Comparing the quality of the entity classification task for manually labeled and automatically extracted relations. The goal is to achieve with extracted relations a classification score that is as good as with manually labeled ones

For the entity classification task the entity labels on X_{train} are assumed to be known whereas the labels on X_{test} are unknown and quality is measured on X_{test}. Figure 2 shows the classification accuracy using several combinations of relations. For every combination we report the accuracy with manually labeled relations and the relations automatically extracted by LRE. The citation relation is always manually labeled. As you can see, the classification quality using the learned relations is very similar to the results using manually labeled relations. For the case of the SameConference relation, prediction quality is even slightly better with the learned relation. For this relation the errors that the relation extraction makes seem to be informative for solving the entity classification task.

All results were repeated 10 times with different train/test splits. We report the means; the standard deviation within each experiment is below 0.01 accuracy.

5.4 Amount of Training Data

In the second experiment we vary the amount of training data for learning the relation. This should show how much pairs of a relation have to be labeled before

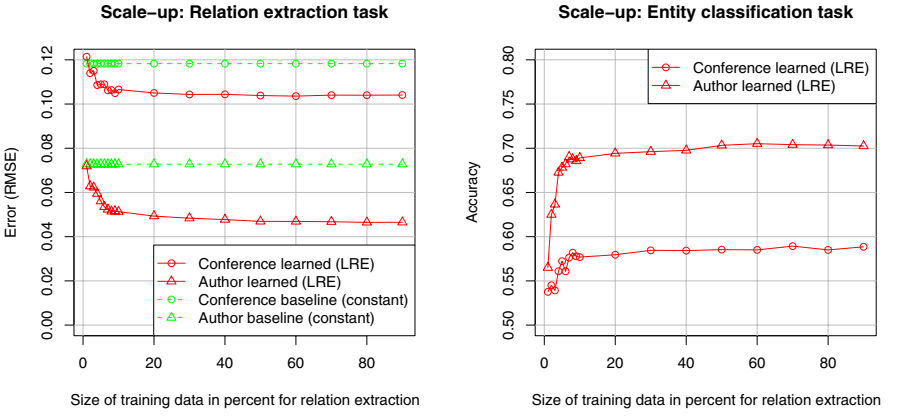


Fig. 3. Quality of the relation extraction task and error (RMSE) of the entity classification using learned relations with increasing amount of labeled data for learning the relation

this relation can be learned successfully and applied to the entity classification task. Our protocol is as following: From the dataset X a test set X_{test} of one third is put aside. Within the remaining 66% we take an increasing proportion $X^{\text{rel}}_{\text{train}}$ from 1 to 90 percent. We assume that the relation weights within $X^{\text{rel}}_{\text{train}}$ are known. We than predict the whole relation on X and evaluate it on X_{test} (see figure 3). We repeated the experiment 8 times with different training data.

Then we use the predicted relation for the task of entity classification. Here we assume that the entity labels on $X \setminus X_{\text{test}}$ are known. We evaluate the quality of the entity classification task on X_{test}. With this evaluation protocol we can measure how many proportions of a relation have to be known (annotated) before it can be successfully learned and applied to the task of entity classification. Please note that with this evaluation protocol the amount of labeled entities is fixed, whereas the amount of labels on relations is varied. Secondly both tasks are measured on a separate set where no information is given.

Figure 3 shows that both task are successfully solved with about 10% of labeles on relations. This means that the proposed LRE method can easily be applied for classification tasks where only implicit relations are given.

6 Conclusion and Future Work

In this paper we have described the LRE method for learning to automatically extract relations from noisy datasets. We have shown that only a small amount of training examples is necessary to learn a relation. We also have shown that the learned relations approximate the manually annotated relations successfully and using them for relational entity classification results in comparable performance to manually annotated relations. Thus our proposed method can replace a domain expert for the task of relation extraction from noisy data.

In future work we would like to investigate active learning techniques to select the training dataset for learning a relation.

Acknowledgements

This work was funded by the X-Media project (www.x-media-project.org) sponsored by the European Commission as part of the Information Society Technologies (IST) programme under EC grant number IST-FP6-026978.

References

1. Lu, Q., Getoor, L.: Link-based text classification. In: Proceedings of IJCAI Workshop on Text Mining and Link Analysis (2003)
2. Macskassy, S.A., Provost, F.: A simple relational classifier. In: Proceedings of the Multi-relational Data Mining Workshop ACM SIGKDD (2003)
3. Neville, J., Jensen, D., Friedland, L., Hay, M.: Learning relational probability trees. In: Proceedings of SIGKDD (2003)
4. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. Journal of the American Statistical Association 64, 1183–1210 (1969)
5. Cohen, W.W., Richman, J.: Learning to match and cluster large high-dimensional data sets for data integration. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2002), Edmonton, Alberta, pp. 475–480 (2002)
6. Bilenko, M., Mooney, R.J.: Adaptive duplicate detection using learnable string similarity measures. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2003), Washington, DC (2003)
7. Preisach, C., Rendle, S., Schmidt-Thieme, L.: Relational classification using automatically extracted relations by record linkage. In: Proceedings of the High Level Information Extraction Workshop at the European Conference on Machine Learning (2008)
8. Preisach, C., Schmidt-Thieme, L.: Ensembles of relational classifiers. Knowledge and Information Systems, 249–272 (2008)
9. Cohen, W.W., Ravikumar, P., Fienberg, S.E.: A comparison of string distance metrics for name-matching tasks. In: Proceedings of the IJCAI 2003 Workshop on Information Integration on the Web, Acapulco, Mexico, pp. 73–78 (August 2003)
10. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines, Software (2001), `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
11. Baxter, R., Christen, P., Churches, T.: A comparison of fast blocking methods for record linkage. In: Proceedings of the 2003 ACM SIGKDD Workshop on Data Cleaning, Record Linkage, and Object Consolidation, Washington, DC (2003)
12. Rendle, S., Schmidt-Thieme, L.: Scaling record linkage to non-uniform distributed class sizes. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS, vol. 5012, pp. 308–319. Springer, Heidelberg (2008)

Author Index

Abolhassani, Hassan 780
Achananuparp, Palakorn 548
Adam, Nabil 16
Afsar, Fayyaz A. 491
Ahmed, Chowdhury Farhan 242, 749
Akram, Muhammad Usman 491
Allison, Lloyd 647
Arif, Muhammad 491
Arimura, Hiroki 172
Attachoo, Boonwat 973

Bailey, James 483
Ben-Tal, Aharon 208
Bennamoun, Mohammed 266
Bernecker, Thomas 673
Bezdek, James 134
Bhadra, Sahely 208
Bhattacharyya, Chiranjib 208
Bheganan, Poramin 74
Bohlscheid, Hans 656
Boughton, Janice 302
Bouguila, Nizar 184, 463
Boulicaut, Jean-François 965
Boutemedjet, Sabri 327
Bringmann, Björn 664
Bunkhumpornpat, Chumphol 475

Cai, Yuanzhe 389
Cai, Yunpeng 401
Cao, Longbing 656
Cao, Minh Duc 647
Chan, Kam Tong 741
Chan, Shing-Kit 86
Chen, Bo 788
Chen, Chun-Ling 147
Chen, Chun Sheng 907
Chen, Depin 764
Chen, Junliang 941
Chen, Xin 867
Chen, Zheng 764
Cheung, Yiu-ming 414
Christen, Peter 507
Chung, Young-joo 339
Cordero Hernandez, Jorge 681

Dan, Ovidiu 556
Daoudi, Khalid 184
Desmarais, Michel C. 883
Dix, Trevor I. 647
Du, Jun 596
Du, Xiaoyong 389
Duan, Jianyong 541

Eftekhar, Milad 780
Eick, Christoph F. 907

Ferri, Cesar 637
Fida, Adnan 491
Fiot, Céline 891
Foitong, Sombut 973
Frank, Eibe 628
Fu, Shunkai 883
Fujimoto, Yu 875
Fukui, Ken-ichi 933
Funes, Ana 637
Furao, Shen 1046

Gao, Jing 363
Gao, Junbin 612
Gao, Yang 278
Gay, Dominique 965
Giles, C. Lee 620
Goodison, Steve 401
Gopal, Madan 52
Grandison, Tyrone 16
Guo, Feng 541
Guo, Jun 788
Guo, Yong Zhen 757

Han, Jiawei 363
Han, Song 883
Hasan, Mohammad Al 122
Hasegawa, Osamu 1046
Hayashi, Kohei 452
He, Hui 788
He, Jun 389
He, Xiaoyun 16
Hernández-Orallo, Jose 637
Hido, Shohei 731
Hino, Hideitsu 875

Hirata, Kouichi 172
Hirayama, Jun-ichiro 452
Ho, Chia-Ling 351
Hoeglinger, Stefan 1006
Hsieh, Nan-Chen 351
Hsu, Kuo-Wei 923
Hu, Derek Hao 278
Hu, Qinmin 949
Hu, Su-Jun 957
Hu, Xiaohua 548, 867
Hu, Xuegang 376
Huang, Anna 628
Huang, Xiangji 949
Hung, Edward 823
Hung, Lun-Ping 351
Hutter, Marcus 813
Hwang, Eenjun 859
Hwang, Seungwon 426

Intarapaiboon, Peerasak 525
Ishii, Shin 452

Jeong, Byeong-Soo 242, 749
Jiang, Yuan 588
Jin, Huidong 813
Jorge, Alípio M. 689

Kantarcioglu, Murat 515
Karypis, George 439
Kashima, Hisashi 1030
Katoh, Takashi 172
Kauffman, Christopher 439
Khan, Latifur 363
Kim, Minkoo 859
Kim, Youngdae 426
Kimura, Masahiro 933
King, Irwin 741
Kitayama, Fumihiko 731
Kitsuregawa, Masaru 339
Koh, Yun Sing 1006
Kriegel, Hans-Peter 673, 831
Kröger, Peer 831
Kumar, Mani Arun 52
Kwee, Agus T. 40

Lallich, Stéphane 705
Lam, Wai 86
Lauw, Hady W. 1054
Le Bras, Yannick 705
Leckie, Christopher 134
Lee, Dongwon 620
Lee, Sang-Hong 723
Lee, Young-Koo 242, 749
Lee, Yue-Shi 232
Lee, Yunsik 604
Lenca, Philippe 705
Leong, Tze-Yun 290
Li, Fang 4
Li, Guoliang 290
Li, Jian 401
Li, Lei 697
Li, Pei 389
Li, Peipei 376
Li, Yun 957
Li, Zhoujun 949
Liang, Qianhui 376
Liang, Tyne 147
Lim, Ee-Peng 1054
Lim, Joon S. 723
Lin, Shuyuan 681
Ling, Charles X. 596
Liu, Bin 302
Liu, Chuanchang 941
Liu, Hongyan 389
Liu, Hua 278
Liu, Ning 764
Liu, Wei 266
Liu, Zhiyuan 254
Loekito, Elsa 483
Lursinsap, Chidchanok 160, 475

Marascu, Alice 891, 899
Marukatat, Sanparith 98
Masseglia, Florent 891, 899
Masud, Mohammad M. 363
Matsuzawa, Hirofumi 731
Meinel, Christoph 772
Meng, Xiangwu 941
Methasate, Ithipan 1038
Milne, David 628
Mizusaki, Junichiro 933
Mocian, Horatiu 556
Moreira, João M. 689
Morimoto, Yasuhiko 839
Muandet, Krikamol 98
Murata, Noboru 875

Nakajima, Shinichi 314
Nantajeewarawat, Ekawit 525
Nath, J. Saketha 208

Nattee, Cholwich 98, 533
Nayak, Richi 74
Ng, Michael 588
Niennattrakul, Vit 981, 1013
Nix, Robert 515
Numao, Masayuki 731, 933

Okumura, Manabu 62
Ouyang, Liang-Yu 232
Oyama, Satoshi 1030

Park, Dong-Chul 604, 915
Park, Jin-Hyeong 196
Park, Laurence A.F. 757
Pears, Russel 1006
Peng, Wen-Chih 997
Peng, Zhiyong 796
Pi, Bingfeng 883
Pinngern, Ouen 973
Poncelet, Pascal 891
Preisach, Christine 1062
Pudjijono, Agus 507
Punyabukkana, Proadpran 580
Pupacdi, Benjarath 122

Qian, Tieyun 796
Qin, Biao 4
Qiu, Qiang 564

Ramamohanarao, Kotagiri 134, 757
Ramírez-Quintana, Maria José 637
Rangwala, Huzefa 439
Rao, Xunrong 883
Ratanamahatana, Chotirat Ann 981, 989, 1013
Reddy, Chandan K. 196
Rendle, Steffen 1062
Renz, Matthias 673
Rho, Seungmin 859
Rinsurongkawong, Vadeerat 907
Rojanavasu, Pornthep 973
Ruengronghirunya, Pongsakorn 981

Saito, Kazumi 933
Sajjipanon, Poat 989
Salem, Saeed 122
Sato, Kazuhisa 933
Schmidt-Thieme, Lars 1062
Schubert, Erich 831
Selmaoui, Nazha 965
Shafiq, Basit 16
She, Qiao-Qiao 588
Shen, Xiajiong 867
Shen, Xiaoyan 941
Shen, Yidong 110
Shen, Zhiyong 110
Sheu, Phillip C.Y. 796
Shi, Yong 1021
Shin, Dong-Kun 723
Siddique, Md. Anisuzzaman 839
Sinapiromsaran, Krung 475
Singh, Goverdhan 891
Soares, Carlos 499, 689
Song, Seheon 859
Sousa, Jorge Freire de 689
Sriphaew, Kritsada 62
Srivastava, Jaideep 796, 923
Su, Bai 110
Suchato, Atiwong 580
Sugiyama, Masashi 314
Suksawatchon, Jakkarin 160
Sun, Guo-Zi 957
Sun, Jun 110
Sun, Maosong 254
Sun, Yijun 401
Suzuki, Einoshin 220, 713
Szeto, Chi-Cheong 823

Takamura, Hiroya 62
Tanbeer, Syed Khairuzzaman 242, 749
Tang, Jie 572
Tang, Wenyin 40
Tangverapong, Akegapon 580
TeCho, Jakkrit 533
Terzi, Evimaria 16
Theeramunkong, Thanaruk 525, 533, 1038
Thuraisingham, Bhavani 363
Tomkins, Andrew 3
Tong, Lijing 541
Tong, Yang 764
Toyoda, Masashi 339
Tsai, Flora S. 40
Tseng, Frank S.C. 147
Tsuda, Koji 1030
Twa, Michael D. 907

Vaidya, Jaideep 16, 515

Wan, Xiaojun 804
Wang, Chiu-Kuang 232

Wang, Hai 620
Wang, Ke 1054
Wang, Liang 134
Wang, Long 772
Wang, Weilei 883
Wattanachon, Ureerat 160
Webb, Geoffrey I. 302
Wei, Ling-Yin 997
Witten, Ian H. 628
Wong, Wilson 266
Woo, Dong-Min 604
Wu, Jung-Wei 232
Wu, Xindong 376, 1021
Wu, Xintao 28

Xia, Yuni 4
Xiao, Jianguo 804
Xie, Zhipeng 849
Xu, Ye 1046
Xu, Yue 74

Yamanishi, Yoshihiro 1030
Yan, Jun 764
Yan, Su 620
Yang, Christopher C. 548
Yang, Geng 957
Yang, Guang-Zhong 1
Yang, Qiang 697
Yang, Wen-Jie 957
Yang, Ying 302
Yao, Fang-Wu 957
Yazdani, Majid 780
Yen, Show-Jane 232
Yin, Xiaoshi 949
Yin, Zhijun 764
Ying, Xiaowei 28
Yu, Hwanjo 426
Yu, Jeffrey Xu 2

Zaki, Mohammed J. 122
Zeng, Hong 414
Zeng, Yifeng 681
Zhang, Chengqi 656
Zhang, Huaifeng 656
Zhang, Jing 572
Zhang, Jun 612
Zhang, Ke 813
Zhang, Mei 541
Zhang, Peng 1021
Zhang, Xian-Xing 278
Zhang, Yang 564
Zhang, Yujie 941
Zhao, Jinxi 1046
Zhao, Yanchang 656
Zheng, Yabin 254
Zhou, Ying 883
Zhou, Zhi-Hua 588, 596
Zhu, Junping 564
Zhu, Xingquan 1021
Zhuo, Hankui 697
Zimek, Arthur 831
Zimmermann, Albrecht 664
Ziou, Djemel 327, 463
Zuefle, Andreas 673