

Lecture Notes in Artificial Intelligence 5254

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

Subseries of Lecture Notes in Computer Science

Yoav Freund László Györfi
György Turán Thomas Zeugmann (Eds.)

Algorithmic Learning Theory

19th International Conference, ALT 2008
Budapest, Hungary, October 13-16, 2008
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

Yoav Freund

University of California at San Diego, La Jolla, USA

E-mail: yfreund@ucsd.edu

László Györfi

Budapest University of Technology and Economics, Hungary

E-mail: gyorfi@szit.bme.hu

György Turán

University of Illinois, Chicago, USA

and University of Szeged, Hungary

E-mail: gyt@uic.edu

Thomas Zeugmann

Hokkaido University, Sapporo, Japan

E-mail: thomas@ist.hokudai.ac.jp

Library of Congress Control Number: 2008936817

CR Subject Classification (1998): I.2.6, F.4, I.7, K.3

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743

ISBN-10 3-540-87986-2 Springer Berlin Heidelberg New York

ISBN-13 978-3-540-87986-2 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 is a part of Springer Science+Business Media

springer.com

© Springer-Verlag Berlin Heidelberg 2008

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper SPIN: 12533824 06/3180 5 4 3 2 1 0

Preface

This volume contains papers presented at the 19th International Conference on Algorithmic Learning Theory (ALT 2008), which was held in Budapest, Hungary during October 13–16, 2008. The conference was co-located with the 11th International Conference on Discovery Science (DS 2008). The technical program of ALT 2008 contained 31 papers selected from 46 submissions, and 5 invited talks. The invited talks were presented in joint sessions of both conferences.

ALT 2008 was the 19th in the ALT conference series, established in Japan in 1990. The series Analogical and Inductive Inference is a predecessor of this series: it was held in 1986, 1989 and 1992, co-located with ALT in 1994, and subsequently merged with ALT. ALT maintains its strong connections to Japan, but has also been held in other countries, such as Australia, Germany, Italy, Singapore, Spain and the USA. The ALT conference series is supervised by its Steering Committee: Naoki Abe (IBM T.J. Watson Research Center, Yorktown, USA), Shai Ben-David (University of Waterloo, Canada), Yoav Freund (University of San Diego, USA), Steffen Lange (Publicity Chair) (FH Darmstadt, Germany), Phil Long (Google, Mountain View, USA), Akira Maruoka (Ishinomaki Senshu University, Japan), Takeshi Shinohara (Kyushu Institute of Technology, Iizuka, Japan), Frank Stephan (National University of Singapore, Republic of Singapore), Einoshin Suzuki (Kyushu University, Fukuoka, Japan), György Turán (University of Illinois at Chicago, USA and University of Szeged, Hungary), Eiji Takimoto (Tohoku University, Japan) Osamu Watanabe (Tokyo University of Technology, Japan) and Thomas Zeugmann (Hokkaido University, Japan; Chair). The ALT Web pages and submission system have been set up (together with Frank Balbach and Jan Poland) and are maintained by Thomas Zeugmann.

The history of conferences in Hungary discussing machine learning goes back, at least, to 1962. The *Proceedings of the Colloquium on the Foundations of Mathematics, Mathematical Machines and Their Applications* (held in Tihany, Hungary during September 11–15, 1962), edited by László Kalmár and published by Akadémiai Kiadó in 1965, contains a paper by S. Watanabe on ‘A Mathematical Explication of Inductive Inference,’ and has a section on artificial intelligence and machine learning, containing a paper by V. M. Glushkov and A. A. Stogny ‘On a Self-Teaching Algorithmic System.’

The present volume contains the texts of the 31 papers presented at ALT 2008, divided into groups of papers on statistical learning, probability and stochastic processes, boosting and experts, active and query learning and inductive inference. The volume also contains abstracts of the invited talks:

- Imre Csiszár (Hungarian Academy of Sciences, Budapest, Hungary): *On Iterative Algorithms with an Information Geometry Background*
- Daniel Keim (Universität Konstanz, Germany): *Visual Analytics: Combining Automated Discovery with Interactive Visualizations*

- László Lovász (Eötvös Loránd University, Budapest, Hungary): *Some Mathematics Behind Graph Property Testing*
- Heikki Mannila (HIIT, Helsinki University of Technology and University of Helsinki, Finland): *Finding Total and Partial Orders from Data for Seriation*
- Tom Mitchell (Carnegie Mellon University, Pittsburgh, PA, USA): *Computational Models of Neural Representations in the Human Brain*

Papers presented at DS 2008 are contained in the DS 2008 proceedings.

The *E. Mark Gold Award* is presented annually at the ALT conferences since 1999, for the most outstanding student contribution. This year, the award was given to Mikhail Dashevskiy for his paper “Aggregating Algorithm for a Space of Analytic Functions.”

We would like to thank the many people and institutions who contributed to the success of the conference. Thanks to the authors of the papers for their submissions, and to the invited speakers for presenting exciting overviews of important recent research developments. We are very grateful to the sponsors of the conference: Aegon Hungary, Fraunhofer IAIS (Sankt Augustin, Germany), Hokkaido University, the Institute of Informatics of the University of Szeged (Hungary), Kyushu University and Tohoku University for their generous financial support.

We are grateful to the members of the Program Committee of ALT 2008. Their hard work in reviewing and discussing the papers made sure that we had an interesting and strong program. We also thank the subreferees assisting the Program Committee. Special thanks go to Local Chair János Csirik (University of Szeged, Hungary), and to Gusztáv Hencsey (Scope Meetings Ltd., Budapest) for the local organization of the conference. We are also grateful to Akira Ishino and Ayumi Shinohara from Tohoku University for their support in preparing ALT/DS 2008. Furthermore, we heartily thank Einoshin Suzuki from Kyushu University for his effort of making travel arrangements for the invited and tutorial speakers. We would like to thank the Discovery Science conference for its ongoing collaboration with ALT, which makes it possible to provide a well-rounded picture of the current theoretical and practical advances in machine learning and the related areas. In particular, we are grateful to Conference Chair Tamás Horváth (University of Bonn and Fraunhofer IAIS, Sankt Augustin, Germany) and Program Committee Chairs Jean-François Boulicaut (INSA Lyon, France) and Michael Berthold (Universität Konstanz, Germany) for their cooperation. Last but not least, we thank Springer for their excellent support in preparing and publishing this volume of the *Lecture Notes in Artificial Intelligence* series.

August 2008

Yoav Freund
László Györfi
György Turán
Thomas Zeugmann

Organization

Conference Chair

László Györfi

Budapest University of Technology and
Economics, Hungary

Program Committee

Naoki Abe

IBM T.J. Watson Research Center, Yorktown

Marta Arias

Universitat Politècnica de Catalunya,
Barcelona

Nicolò Cesa-Bianchi

Università degli Studi di Milano

Koby Crammer

University of Pennsylvania, Philadelphia

Eyal Even-Dar

Google Inc., New York

Yoav Freund

University of California, San Diego (Chair)

Ricard Gavaldà

Universitat Politècnica de Catalunya,
Barcelona

Mark Herbster

University College London

Kouichi Hirata

Kyushu Institute of Technology, Iizuka

Marcus Hutter

Australian National University, Canberra, and
NICTA

Efim Kinber

Sacred Heart University, Fairfield

Gábor Lugosi

Pompeu Fabra University, Barcelona

Ulrike von Luxburg

MPI for Biological Cybernetics, Tübingen

Partha Niyogi

University of Chicago

Csaba Szepesvári

University of Alberta, Edmonton

György Turán

University of Illinois at Chicago and
University of Szeged (Chair)

Nicolas Vayatis

Ecole Normale Supérieure de Cachan

Vladimir Vovk

Royal Holloway, University of London

Manfred K. Warmuth

University of California, Santa Cruz

Akihiro Yamamoto

Kyoto University

Sandra Zilles

University of Alberta, Edmonton

Local Arrangements

János Csirik

University of Szeged and RGAI Szeged,
Hungary

Subreferees

Yohji Akama
Jakob Abernethy
Alberto Bertoni
Giovanni Cavallanti
Kamalika Chaudhuri
Alexey Chernov
Matthew de Brecht
Koichiro Doi
Evan Ettinger
Thomas Gaertner
Karen Glocer
Eiju Hirowatari
Maya Hristakeva
Tamás Horváth
Daniel Hsu
Mayank Kabra

Yuri Kalnishkan
Kevin T. Kelly
Steffen Lange
Guy Lever
Aurelie C. Lozano
Markus Maier
Tetsuhiro Miyahara
Jan Poland
Scott Sanner
Hiroshi Sakamoto
Nathan Srebro
Shinichi Shimosono
Takeshi Shinohara
Rajmonda Sulo
Jie Yang

Sponsoring Institutions

Aegon Hungary
Fraunhofer IAIS, Sankt Augustin, Germany
Institute of Informatics, University of Szeged, Hungary
Department of System Information Science, Tohoku University
Department of Informatics, Kyushu University
Division of Computer Science, Hokkaido University

Erratum

Hutter, M., Servedio, R.A., Takimoto, E. (Eds.) ALT 2007. LNCS (LNAI), vol. 4754. Springer, Heidelberg (2007)

By mistake the following text was not included in the frontmatter of the ALT 2007 proceedings volume LNAI 4754.

Conference Chair

Eiji Takimoto

Tohoku University, Japan

Program Committee

Marcus Hutter

Australian National University, Australia
(Chair)

Rocco Servedio

Columbia University, USA (Chair)

Marta Arias

Columbia University, USA

Nader Bshouty

Technion, Israel

Sanjoy Dasgupta

University of California, San Diego, USA

Ran El-Yaniv

Technion, Israel

Claudio Gentile

Università dell'Insubria, Italy

Lisa Hellerstein

Polytechnic University, USA

Sanjay Jain

National University of Singapore, Singapore

Yuri Kalnishkan

Royal Holloway University of London, UK

Jyrki Kivinen

University of Helsinki, Finland

Adam Klivans

University of Texas, Austin, USA

Phil Long

Google, USA

Jan Poland

Hokkaido University, Japan

Daniil Ryabko

IDSIA, Switzerland

Frank Stephan

National University of Singapore, Singapore

Christino Tamon

Clarkson University, USA

Vladimir Vovk

Royal Holloway University of London, UK

Osamu Watanabe

Tokyo Institute of Technology, Japan

Bob Williamson

Australian National University, Australia

Local Arrangements

Akira Ishino

Tohoku University, Japan

Subreferees

Douglas Aberdeen	Shane Legg
Ron Begleiter	Shie Mannor
John Case	Hanna Mazzawi
Jiang Chen	Dmitry Pechyony
Alexey Chernov	Jim Royer
Alex Clarcck	Tamer Salman
Yoav Freund	Shai Shalev-Shwartz
Alex Fukunaga	Takeshi Shinohara
Bill Gasarch	Etsuji Tomita
Ricard Gavaldá	György Turán
Paul Goldberg	Rolf Wiehagen
Simon Guenter	Yair Wiener
Jeff C. Jackson	Ryo Yoshinaka
Efim Kinber	Sandra Zilles
Gregory Kucherov	
Steffen Lange	

Sponsoring Institutions

Air Force Office of Scientific Research (AFOSR)
Asian Office of Aerospace Research and Development (AOARD)
Computation, IEICE of Japan
Google
GSIS, Tohoku University
New Horizons in Computing (NHC)
RIEC, Tohoku University
Semi-Structured Data Mining Project
Division of Computer Science, Hokkaido University
Institute for Theoretical Computer Science, University at Lübeck

Table of Contents

Invited Papers

On Iterative Algorithms with an Information Geometry Background	1
<i>Imre Csizsár</i>	
Visual Analytics: Combining Automated Discovery with Interactive Visualizations	2
<i>Daniel A. Keim, Florian Mansmann, Daniela Oelke, and Hartmut Ziegler</i>	
Some Mathematics behind Graph Property Testing	3
<i>László Lovász</i>	
Finding Total and Partial Orders from Data for Seriation	4
<i>Heikki Mannila</i>	
Computational Models of Neural Representations in the Human Brain (Extended Abstract)	5
<i>Tom M. Mitchell</i>	

Regular Contributions

Statistical Learning

Generalization Bounds for Some Ordinal Regression Algorithms	7
<i>Shivani Agarwal</i>	
Approximation of the Optimal ROC Curve and a Tree-Based Ranking Algorithm	22
<i>Stéphan Cléménçon and Nicolas Vayatis</i>	
Sample Selection Bias Correction Theory	38
<i>Corinna Cortes, Mehryar Mohri, Michael Riley, and Afshin Rostamizadeh</i>	
Exploiting Cluster-Structure to Predict the Labeling of a Graph	54
<i>Mark Herbster</i>	
A Uniform Lower Error Bound for Half-Space Learning	70
<i>Andreas Maurer and Massimiliano Pontil</i>	
Generalization Bounds for K-Dimensional Coding Schemes in Hilbert Spaces	79
<i>Andreas Maurer and Massimiliano Pontil</i>	

Learning and Generalization with the Information Bottleneck	92
<i>Ohad Shamir, Sivan Sabato, and Naftali Tishby</i>	

Probability and Stochastic Processes

Growth Optimal Investment with Transaction Costs	108
<i>László Györfi and István Vajda</i>	
Online Regret Bounds for Markov Decision Processes with Deterministic Transitions	123
<i>Ronald Ortner</i>	
On-Line Probability, Complexity and Randomness	138
<i>Alexey Chernov, Alexander Shen, Nikolai Vereshchagin, and Vladimir Vovk</i>	
Prequential Randomness	154
<i>Vladimir Vovk and Alexander Shen</i>	
Some Sufficient Conditions on an Arbitrary Class of Stochastic Processes for the Existence of a Predictor	169
<i>Daniil Ryabko</i>	
Nonparametric Independence Tests: Space Partitioning and Kernel Approaches	183
<i>Arthur Gretton and László Györfi</i>	

Boosting and Experts

Supermartingales in Prediction with Expert Advice	199
<i>Alexey Chernov, Yuri Kalnishkan, Fedor Zhdanov, and Vladimir Vovk</i>	
Aggregating Algorithm for a Space of Analytic Functions	214
<i>Mikhail Dashevskiy</i>	
Smooth Boosting for Margin-Based Ranking	227
<i>Jun-ichi Moribe, Kohei Hatano, Eiji Takimoto, and Masayuki Takeda</i>	
Learning with Continuous Experts Using Drifting Games	240
<i>Indraneel Mukherjee and Robert E. Schapire</i>	
Entropy Regularized LPBoost	256
<i>Manfred K. Warmuth, Karen A. Glocer, and S.V.N. Vishwanathan</i>	

Active Learning and Queries

Optimally Learning Social Networks with Activations and Suppressions	272
<i>Dana Angluin, James Aspnes, and Lev Reyzin</i>	
Active Learning in Multi-armed Bandits	287
<i>András Antos, Varun Grover, and Csaba Szepesvári</i>	

Query Learning and Certificates in Lattices	303
<i>M. Arias and J.L. Balcázar</i>	
Clustering with Interactive Feedback	316
<i>Maria-Florina Balcan and Avrim Blum</i>	
Active Learning of Group-Structured Environments	329
<i>Gábor Bartók, Csaba Szepesvári, and Sandra Zilles</i>	
Finding the Rare Cube	344
<i>Shlomo Hoory and Oded Margalit</i>	

Inductive Inference

Iterative Learning of Simple External Contextual Languages	359
<i>Leonor Becerra-Bonache, John Case, Sanjay Jain, and Frank Stephan</i>	
Topological Properties of Concept Spaces	374
<i>Matthew de Brecht and Akihiro Yamamoto</i>	
Dynamically Delayed Postdictive Completeness and Consistency in Learning	389
<i>John Case and Timo Kötzing</i>	
Dynamic Modeling in Inductive Inference	404
<i>John Case and Timo Kötzing</i>	
Optimal Language Learning	419
<i>John Case and Samuel E. Moelius III</i>	
Numberings Optimal for Learning	434
<i>Sanjay Jain and Frank Stephan</i>	
Learning with Temporary Memory	449
<i>Steffen Lange, Samuel E. Moelius III, and Sandra Zilles</i>	

Erratum

Erratum: Constructing Multiclass Learners from Binary Learners: A Simple Black-Box Analysis of the Generalization Errors	464
<i>Jittat Fakcharoenphol and Boonserm Kijsirikul</i>	
Author Index	467

On Iterative Algorithms with an Information Geometry Background

Imre Csiszár

Alfréd Rényi Institute of Mathematics, Hungarian Academy of Sciences,
Budapest, Hungary
`csiszar@renyi.hu`

Abstract. Several extremum problems in Statistics and Artificial Intelligence, e.g., likelihood maximization, are often solved by iterative algorithms such as iterative scaling or the EM algorithm, admitting an intuitive “geometric” interpretation as iterated projections in the sense of Kullback information divergence. Such iterative algorithms, including those using Bregman rather than Kullback divergences, will be surveyed. It will be hinted to that the celebrated belief propagation (or sum-product) algorithm may also admit a similar interpretation.

Visual Analytics: Combining Automated Discovery with Interactive Visualizations*

Daniel A. Keim, Florian Mansmann, Daniela Oelke, and Hartmut Ziegler

University of Konstanz, Germany
`first.lastname@uni-konstanz.de`
`http://infovis.uni-konstanz.de`

Abstract. In numerous application areas fast growing data sets develop with ever higher complexity and dynamics. A central challenge is to filter the substantial information and to communicate it to humans in an appropriate way. Approaches, which work either on a purely analytical or on a purely visual level, do not sufficiently help due to the dynamics and complexity of the underlying processes or due to a situation with intelligent opponents. Only a combination of data analysis and visualization techniques make an effective access to the otherwise unmanageably complex data sets possible.

Visual analysis techniques extend the perceptual and cognitive abilities of humans with automatic data analysis techniques, and help to gain insights for optimizing and steering complicated processes. In the paper, we introduce the basic idea of Visual Analytics, explain how automated discovery and visual analysis methods can be combined, discuss the main challenges of Visual Analytics, and show that combining automatic and visual analysis is the only chance to capture the complex, changing characteristics of the data. To further explain the Visual Analytics process, we provide examples from the area of document analysis.

* The full version of this paper is published in the Proceedings of the 11th International Conference on Discovery Science, Lecture Notes in Artificial Intelligence Vol. 5255.

Some Mathematics behind Graph Property Testing

László Lovász

Department of Computer Science, Eötvös Loránd University, Budapest, Hungary
`lovasz@cs.elte.hu`

Abstract. Graph property testing is the third reincarnation of the same general question, after statistics and learning theory. In its simplest form, we have a huge graph (we don't even know its size), and we draw a sample of the node set of bounded size. What properties of the graph can be deduced from this sample?

The graph property testing model was first introduced by Goldreich, Goldwasser and Ron (but related questions were considered before). In the context of dense graphs, a very general result is due to Alon and Shapira, who proved that every hereditary graph property is testable.

Using the theory of graph limits, Lovász and Szegedy defined an analytic version of the (dense) graph property testing problem, which can be formulated as studying an unknown 2-variable symmetric function through sampling from its domain and studying the random graph obtained when using the function values as edge probabilities. This analytic version allows for simpler formulation of the problems, and leads to various characterizations of testable properties. These results can be applied to the original graph-theoretic property testing. In particular, they lead to a new combinatorial characterization of testable graph properties.

We survey these results, along with analogous results for graphs with bounded degree.

Finding Total and Partial Orders from Data for Seriation*

Heikki Mannila

HIIT

Helsinki University of Technology and
University of Helsinki, Finland

`heikki.mannila@tkk.fi`

Abstract. Ordering and ranking items of different types (observations, web pages, etc.) are important tasks in various applications, such as query processing and scientific data mining. We consider different problems of inferring total or partial orders from data, with special emphasis on applications to the seriation problem in paleontology. Seriation can be viewed as the task of ordering rows of a 0-1 matrix so that certain conditions hold. We review different approaches to this task, including spectral ordering methods, techniques for finding partial orders, and probabilistic models using MCMC methods.

Joint work with Antti Ukkonen, Aris Gionis, Mikael Fortelius, Kai Puolamäki, and Jukka Jernvall.

* The full version of this paper is published in the Proceedings of the 11th International Conference on Discovery Science, Lecture Notes in Artificial Intelligence Vol. 5255.

Computational Models of Neural Representations in the Human Brain

(Extended Abstract)

Tom M. Mitchell

Carnegie Mellon University, Pittsburgh, PA 15213, USA

tom.mitchell@cmu.edu

<http://www.cs.cmu.edu/~tom>

For many centuries scientists have wondered how the human brain represents thoughts in terms of the underlying biology of neural activity. Philosophers, linguists, cognitive scientists and others have proposed theories, for example suggesting that the brain organizes conceptual information in hierarchies of concepts, or that it instead represents different concepts in different local regions of the cortex.

Over the past decade rapid progress has been made on the study of human brain function, driven by the advent of modern brain imaging methods such as functional Magnetic Resonance Imaging (fMRI), which is able to produce three dimensional images of brain activity at a spatial resolution of approximately one millimeter. Using fMRI we have spent several years exploring the question of how the brain represents the meanings of individual words in terms of patterns of neural activity observed with fMRI. The talk accompanying this abstract will present our results, and the use of machine learning methods to analyze this data and to develop predictive computational models. In particular, we ([1],[2]) have explored the following questions:

- *Can one observe differences in neural activity using fMRI, as people think about different items such as “hammer” versus “house”?* Many researchers have now demonstrated that fMRI does indeed reveal differences in neural activity due to considering different items. We present results [1] showing that it is possible to train a machine learning classifier to discover the different patterns of activity associated with different items, and to use this to classify which of several items a person is considering, based on their neural activity.
- *Are neural representations of concepts similar if the stimulus is a word, versus a line drawing of the object?* We tested this question by asking whether a machine learning classifier trained on fMRI data collected when a person reads words, could successfully distinguish which item they were thinking about when the stimuli were line drawings. The classifier performed nearly as accurately classifying fMRI activity generated by line drawing stimuli as by word stimuli, despite being trained on word stimuli. This result suggests that the neural activity captured by the classifier reflects the semantics of the item, and not simply some surface perceptual features associated with the particular form of stimulus.

- *Are neural representations similar across different people?* We tested this question by asking whether a machine learning classifier trained on fMRI data collected from a group of people, could successfully distinguish which item a new person was thinking about, despite the fact that the classifier had never seen data from this new person. These experiments were performed for stimuli corresponding to concrete nouns (i.e., nouns such as “bicycle” and “tomato” which describe physical objects). We found the answer is yes, although accuracies vary by person. This result suggests that despite the fact that individual people are clearly different, our brains use similar neural encodings of semantics of concrete nouns.
- *Can we discover underlying principles of neural representations sufficient to develop a computational model that predicts neural representations for arbitrary words?* We recently developed a computational model that predicts the neural representation for any concrete noun. While imperfect, this model performs well on the 100 words for which we have data to test it. The model is trained using a combination of fMRI data for dozens of words, plus data from a trillion word text corpus that reflects the way in which people typically use words in natural language. This model represents a new approach to computational studies of neural representations in the human brain.

Acknowledgments. The research summarized here is the product of a large team of collaborators, including Marcel Just, Andy Carlson, Kai-Min Chang, Rebecca Hutchinson, Vicente Maleve, Rob Mason, Mark Palatucci, Francisco Pereira, Indra Rustandi, Svetlana Shinkareva, Wei Wang and others. We are grateful for support from the W.M. Keck foundation and the National Science Foundation.

References

- [1] Shinkareva, S., Mason, R., Malave, V., Wang, W., Mitchell, T., Just, M.: Using fMRI Brain Activation to Identify Cognitive States Associated with Perception of Tools and Dwellings. *PLoS ONE* 3(1), 1394 (2008), doi:10.1371/journal.pone.0001394
- [2] Mitchell, T., Shinkareva, S., Carlson, A., Chang, K., Malave, V., Mason, R., Just, M.: Predicting Human Brain Activity Associated with the Meanings of Nouns. *Science* 320, 1191 (2008), doi: 10.1126/science.1152876.

Generalization Bounds for Some Ordinal Regression Algorithms

Shivani Agarwal

Massachusetts Institute of Technology, Cambridge MA 02139, USA
shivani@mit.edu

Abstract. The problem of ordinal regression, in which the goal is to learn a rule to predict labels from a discrete but ordered set, has gained considerable attention in machine learning in recent years. We study generalization properties of algorithms for this problem. We start with the most basic algorithms that work by learning a real-valued function in a regression framework and then rounding off a predicted real value to the closest discrete label; our most basic bounds for such algorithms are derived by relating the ordinal regression error of the resulting prediction rule to the regression error of the learned real-valued function. We end with a margin-based bound for the state-of-the-art ordinal regression algorithm of Chu & Keerthi (2007).

1 Introduction

In addition to the classical problems of classification and regression, several new types of learning problems have emerged in recent years. Among these is the problem of ordinal regression, in which the goal is to learn a rule to predict labels of an ordinal scale, *i.e.*, labels from a discrete but ordered set. Ordinal regression is common in the social sciences where surveys frequently ask users to rate items on an ordinal scale, and has been studied previously in the statistical literature [1]. Recent years have seen a surge of interest in ordinal regression from a machine learning perspective [2,3,4,5,6,7,8,9,10,11,12], partly due to the fact that it is a unique problem which shares characteristics of many other learning problems such as classification, regression, and ranking – and yet is distinct from each – but also due to the fact that ordinal regression increasingly finds applications in diverse areas such as finance, medicine, information retrieval, and user-preference modeling.

Although there has been considerable progress in developing learning algorithms for ordinal regression in the last few years, in most cases, not much is known about the theoretical properties of these algorithms: how well they generalize, and at what rate (if at all) they converge to an optimal solution. In this paper, we begin an attempt to fill this gap. Our focus is on the question of generalization properties of these algorithms.

1.1 Previous Results

In the ordinal regression problem, described in greater detail in Section 2, the learner is given a sequence of labeled training examples $S = ((x_1, y_1), \dots, (x_m, y_m))$, the x_i being instances in some instance space X and the y_i being labels in a discrete, ordered

set, which we take to be $[r] = \{1, \dots, r\}$ for some $r \in \mathbb{N}$, and the goal is to learn a rule $g : X \rightarrow [r]$ that predicts accurately labels of future instances. The penalty incurred for a wrong prediction is larger for predictions farther from the true label: in the setting we consider, the penalty incurred by g on an example (x, y) is proportional to $|g(x) - y|$.

Barring some work on neural network models in the 1990s [13] (which was inspired largely by the statistical models of [1]), among the earliest studies of ordinal regression in machine learning was that of Herbrich et al. [2], in which a large-margin algorithm similar to support vector machines (SVMs) was proposed. This work included a margin-based generalization bound for the zero-training-error case¹. However, the setting of [2] differs from the setting described above, in that the error of a prediction rule is measured in terms of pairs of examples for which the relative order between the predicted labels differs from the relative order between the true labels; indeed, in their setting, it is possible for a rule that predicts the wrong labels on two instances to incur no loss at all, as long as the relative order of those labels is correct. In this sense, the problem studied in [2] is more similar to some ranking problems than what has now come to be commonly accepted as the problem of ordinal regression.

The years following [2] saw several developments in ordinal regression. Crammer & Singer [14] proposed an algorithm for ordinal regression in the online learning model, motivated by the perceptron algorithm for classification, and provided a mistake bound for their algorithm. This was followed by an extension of their algorithm by Harrington [7], in which an online approximation to the Bayes point was sought, as well as extensions in [5] which included a multiplicative update algorithm. All of these came with mistake bounds; indeed, it can safely be said that these online algorithms for ordinal regression are among the better understood theoretically.

In the traditional offline (or ‘batch’) learning model, there have been four broad approaches to developing ordinal regression algorithms. The first approach treats the labels y_i as real values, uses a standard regression algorithm to learn a real-valued function $f : X \rightarrow \mathbb{R}$, and then predicts the label of a new instance x by rounding the predicted real value $f(x)$ to the closest discrete label. This approach can be used with any regression learning algorithm, and was discussed specifically in the context of regression trees by Kramer et al. [3]; Kramer et al. also discussed some simple methods to modify the regression tree learning algorithm to directly predict labels for ordinal regression.

The second approach consists of reducing an ordinal regression problem to one or more binary classification problems, which can then be solved using a standard binary classification algorithm. For example, Frank & Hall [4] proposed a method for reducing an r -label ordinal regression problem to a series of $r - 1$ binary classification problems, each of which could be solved using any classifier capable of producing probability estimates; the method was somewhat ad-hoc as it required certain independence assumptions in order to compute probabilities needed for making label predictions. More recently, Cardoso & da Costa [11] have proposed an algorithm for transforming an ordinal regression problem in a Euclidean space into a single binary classification problem in a higher-dimensional space.

¹ We note that the bound in [2] contains a mistake, although the mistake is easily corrected: the article incorrectly claims that a sample of m independent instances gives $m - 1$ independent pairs of instances; this can be corrected by replacing $m - 1$ in the bound with $m/2$.

In the third approach, algorithms are designed to directly learn prediction rules for ordinal regression; as in the case of [2] or [14], this usually consists of learning a real-valued function $f : X \rightarrow \mathbb{R}$ together with a set of thresholds $b^1 \leq \dots \leq b^{r-1} \leq b^r = \infty$, the resulting prediction rule being given by $g(x) = \min_{j \in \{1, \dots, r\}} \{j : f(x) < b^j\}$. Going back full circle to the large-margin framework used in [2], Shashua & Levin [6] proposed two large-margin algorithms, also motivated by SVMs, to directly learn prediction rules for ordinal regression; unlike [2], the problem setting in this case corresponds to the setting described above, in which the error of a prediction rule is measured in terms of the difference between the true and predicted labels. However, as pointed out by Chu & Keerthi [10], one of the algorithms in [6] contains a fundamental flaw in that it fails to guarantee the necessary inequalities among the thresholds; Chu & Keerthi offer a modification that corrects this, as well as a second large-margin algorithm that is among the current state of the art.

Finally, there has also been some work on Bayesian learning algorithms for ordinal regression, such as that by Chu & Ghahramani [8].

Among all the (offline) algorithms discussed above, only two – the classification-based algorithm of Cardoso & da Costa [11] and the large-margin algorithm of Shashua & Levin [6] – have been accompanied by some form of generalization analysis; unfortunately, in both cases, the analysis is either incorrect or incomplete. Cardoso & da Costa (in an appendix of [11]) attempt to derive a margin-based bound for their algorithm by applying a bound for binary classifiers; however it is not clear to what function class the bound is applied, and on closer inspection it becomes clear that the analysis is, in fact, incorrect. Shashua & Levin (in [15]) also attempt to derive a margin-based bound for their algorithm; again, the quantities involved are not clearly defined, and furthermore the analysis claims to bound the ‘probability that a test example will not be separated correctly’ which, as we argue in Section 2, is not the natural quantity of interest in ordinal regression, and so this analysis too appears, at best, to be incomplete. These failed attempts – as well as the lack of any theoretical analysis for the other algorithms – all point to the need for a more careful analysis of the generalization properties of ordinal regression algorithms. This is what we aim to achieve in this paper.

1.2 Our Results

We formalize the mathematical setting involved in studying generalization properties of ordinal regression algorithms, including clear definitions of the quantities involved (Section 2), and then proceed to derive generalization bounds for some of these algorithms. We start with the most basic algorithms that work by learning a real-valued function in a regression framework and then rounding off a predicted real value to the closest discrete label; we relate the ordinal regression error of the resulting prediction rule to the regression error of the learned real-valued function, and use this to derive some basic ‘black-box’ generalization bounds for such algorithms (Section 3). Next we employ a direct stability analysis for such algorithms (Section 4); this gives better bounds in some cases. We also investigate the use of stability analysis for more general algorithms for ordinal regression, and outline some difficulties involved in achieving this goal (Section 5). Finally, we derive a margin-based bound for the state-of-the-art ordinal regression algorithm of Chu & Keerthi [10] (Section 6).

2 The Ordinal Regression Problem

The setting of the ordinal regression problem can be described as follows. There is an instance space X from which instances are drawn, and a finite set of discrete labels that have a total order relation among them; we take this set to be $[r] = \{1, \dots, r\}$ for some $r \in \mathbb{N}$, with the usual ‘greater than’ order relation among the labels. The learner is given a sequence of labeled training examples $S = ((x_1, y_1), \dots, (x_m, y_m)) \in (X \times [r])^m$, and the goal is to learn a rule $g : X \rightarrow [r]$ that predicts accurately labels of future instances. For example, consider a user-preference modeling task in which a user gives ratings to the books she reads – ranging from 1 to 5 stars – and the goal is to predict her ratings on new books. In this case the ratings can be viewed as a discrete set of labels, but these labels clearly have an ordering among them, and so this is an instance of an ordinal regression problem (with $r = 5$).

Ordinal regression shares properties of both classification and regression: as in (multiclass) classification, the goal is to assign one of r different labels to a new instance; but as in regression, the labels are ordered, suggesting that predictions farther from the true label should incur a greater penalty than predictions closer to the true label. Indeed, if a book is rated by a user as having 5 stars, then a prediction of 4 stars would clearly be preferable to a prediction of 1 star (and should therefore incur a smaller penalty). As in classification and regression, it is generally assumed that all examples (x, y) (both training examples and future, unseen examples) are drawn randomly and independently according to some (unknown) distribution $\mathcal{D}$ on $X \times [r]$.

There are many ways to evaluate the quality of a prediction rule $g : X \rightarrow [r]$; indeed, some recent work has focused on comparing different evaluation criteria for ordinal regression [12]. In applications where the relative ranking of instances is important, it may be appropriate to consider the performance of g on pairs of examples $(x, y), (x', y')$, and count a mistake if the relative order of the predicted labels $g(x), g(x')$ differs from the relative order of the true labels y, y' , *i.e.*, if $(y - y')(g(x) - g(x')) < 0$. This leads to the following ‘pair-wise’ error for evaluating g :

$$L_{\mathcal{D}}^{\text{pairs}}(g) = \mathbf{E}_{((x,y),(x',y')) \sim \mathcal{D} \times \mathcal{D}} [\mathbf{I}_{\{(y-y')(g(x)-g(x')) < 0\}}] , \quad (1)$$

where $\mathbf{I}_{\phi}$ denotes the indicator variable whose value is 1 if ϕ is true and 0 otherwise; this is simply the probability that g incurs a mistake on a random pair of examples, each drawn independently according to $\mathcal{D}$. As discussed in Section 1, this is the evaluation criterion used by Herbrich et al. [2]. This criterion focuses on the relative order of instances in the ranking induced by g , and is similar to the criterion used in a form of ranking problem termed r -partite ranking (see, for example, [16]).

In the setting we consider, however, the goal is not to produce a ranking of instances, but rather to predict a label for each instance that is as close as possible to the true label; in other words, we are interested in the performance of g on *individual* examples. Again, there are several ways of measuring the loss of a prediction rule g on an example (x, y) depending on the particular application and the semantics associated with the labels. A common approach, which has been used explicitly or implicitly by a majority of the more

recent papers on ordinal regression, is to use the absolute loss $\ell_{\text{ord}}(g, (x, y)) = |g(x) - y|$ – henceforth referred to as the *ordinal regression loss* – which simply measures the absolute difference between the predicted and true labels.² This is the loss we use.

Thus, in the setting considered in this paper, the quality of a prediction rule $g : X \rightarrow [r]$ is measured by its *expected ordinal regression error* with respect to $\mathcal{D}$:

$$L_{\mathcal{D}}^{\text{ord}}(g) = \mathbf{E}_{(x,y) \sim \mathcal{D}} [|g(x) - y|]. \quad (2)$$

In practice, since the distribution $\mathcal{D}$ is not known, the expected ordinal regression error of a prediction rule g cannot be computed exactly; instead, it must be estimated using an empirically observable quantity, such as the *empirical ordinal regression error* of g with respect to a finite sample $S = ((x_1, y_1), \dots, (x_m, y_m)) \in (X \times [r])^m$:

$$\hat{L}_S^{\text{ord}}(g) = \frac{1}{m} \sum_{i=1}^m |g(x_i) - y_i|. \quad (3)$$

Our goal in this paper is to derive generalization bounds for ordinal regression algorithms; in particular, we wish to derive probabilistic bounds on the expected ordinal regression error of a learned prediction rule in terms of an empirical quantity, such as the empirical error of the rule, measured with respect to the training sample from which it is learned.

3 Black-Box Bounds for Regression-Based Algorithms

We start with the most basic algorithms that learn a real-valued function $f : X \rightarrow \mathbb{R}$ in a standard regression setting, treating the labels y_i simply as real values, and then round off a predicted real value $f(x)$ to the closest label in $[r]$; the prediction rule for such an algorithm is given by

$$g_f(x) = \begin{cases} 1, & \text{if } f(x) < 1 + \frac{1}{2} \\ j, & \text{if } j - \frac{1}{2} \leq f(x) < j + \frac{1}{2}, \quad j \in \{2, \dots, r-1\} \\ r, & \text{if } f(x) \geq r - \frac{1}{2}, \end{cases} \quad (4)$$

which can also be written as

$$g_f(x) = \min_{j \in \{1, \dots, r\}} \{j : f(x) < b^j\}, \quad (5)$$

with $b^j = j + \frac{1}{2}$ for $j \in \{1, \dots, r-1\}$ and $b^r = \infty$. In this section, we relate the ordinal regression error of such a prediction rule g_f to the regression error of the underlying real-valued function f ; this allows us to use established generalization bounds for regression algorithms to obtain some basic black-box bounds for the resulting ordinal regression algorithms.

² While many of the ordinal regression papers discussed in Section 1 (including [14,6,5] which, despite the the term ‘ranking’ in their titles, are on ordinal regression) explicitly employ the absolute loss, several others (such as [7,11]) use this loss implicitly – in the form of the mean absolute error (MAE) or mean absolute distance (MAD) criterion – when measuring performance empirically on benchmark data sets.

The loss of a real-valued function $f : X \rightarrow \mathbb{R}$ on an example $(x, y) \in X \times \mathbb{R}$ in the regression setting is usually measured either by the absolute loss $\ell_{\text{abs}}(f, (x, y)) = |f(x) - y|$, or more frequently, by the squared loss $\ell_{\text{sq}}(f, (x, y)) = (f(x) - y)^2$. The quality of f with respect to a distribution $\mathcal{D}$ on $X \times \mathbb{R}$ is then measured by either its expected absolute error or its expected squared error:

$$L_{\mathcal{D}}^{\text{abs}}(f) = \mathbf{E}_{(x,y) \sim \mathcal{D}} [|f(x) - y|] ; \quad L_{\mathcal{D}}^{\text{sq}}(f) = \mathbf{E}_{(x,y) \sim \mathcal{D}} [(f(x) - y)^2] . \quad (6)$$

The corresponding empirical quantities with respect to $S = ((x_1, y_1), \dots, (x_m, y_m)) \in (X \times \mathbb{R})^m$ are defined analogously:

$$\widehat{L}_S^{\text{abs}}(f) = \frac{1}{m} \sum_{i=1}^m |f(x_i) - y_i| ; \quad \widehat{L}_S^{\text{sq}}(f) = \frac{1}{m} \sum_{i=1}^m (f(x_i) - y_i)^2 . \quad (7)$$

The following simple lemma provides a connection between the respective errors of a real-valued function f and the corresponding prediction rule g_f .

Lemma 1. *For all $f : X \rightarrow \mathbb{R}$ and for all $(x, y) \in X \times [r]$, we have:*

1. $|g_f(x) - y| \leq \min(|f(x) - y| + \frac{1}{2}, 2|f(x) - y|)$.
2. $|g_f(x) - y| \leq \min(2(f(x) - y)^2 + \frac{1}{2}, 4(f(x) - y)^2)$.

Proof. If $|g_f(x) - y| = 0$, both results clearly hold. Therefore assume $|g_f(x) - y| \neq 0$. Then $|g_f(x) - y| \in \{1, \dots, r-1\}$, and from the definition of g_f , it follows that

$$|f(x) - y| \geq \frac{1}{2} . \quad (8)$$

Part 1. The definition of g_f easily yields the first inequality:

$$|g_f(x) - y| \leq |f(x) - y| + \frac{1}{2} . \quad (9)$$

Combining this with Eq. (8) gives the second inequality:

$$|g_f(x) - y| \leq 2|f(x) - y| . \quad (10)$$

Part 2. From Eq. (8), we have $2|f(x) - y| \geq 1$. Since $a \geq 1 \Rightarrow a \leq a^2$, this gives

$$2|f(x) - y| \leq 4(f(x) - y)^2 .$$

Combining this with Eqs. (9) and (10) yields the desired inequalities. $\square$

Theorem 1. *For all $f : X \rightarrow \mathbb{R}$ and for all distributions $\mathcal{D}$ on $X \times [r]$, we have:*

1. $L_{\mathcal{D}}^{\text{ord}}(g_f) \leq \phi(L_{\mathcal{D}}^{\text{abs}}(f))$, where $\phi(L) = \min(L + \frac{1}{2}, 2L)$.
2. $L_{\mathcal{D}}^{\text{ord}}(g_f) \leq \psi(L_{\mathcal{D}}^{\text{sq}}(f))$, where $\psi(L) = \min(2L + \frac{1}{2}, 4L)$.

Proof. Immediate from Lemma 1. $\square$

As a consequence of Theorem 1, any generalization result that provides a bound on the expected (absolute or squared) error of the real-valued function f learned by a regression algorithm immediately provides a bound also on the expected ordinal regression error of the corresponding prediction rule g_f . Below we provide two specific examples of such black-box bounds: the first uses a standard uniform convergence bound for regression algorithms that is expressed in terms of covering numbers; the second uses a stability bound for regression algorithms due to Bousquet & Elisseeff [17].

Theorem 2 (Covering number bound). *Let $\mathcal{F}$ be a class of real-valued functions on X , and let $\mathcal{A}$ be an ordinal regression algorithm which, given as input a training sample $S \in (X \times [r])^m$, learns a real-valued function $f_S \in \mathcal{F}$ and returns as output the prediction rule $g_S \equiv g_{f_S}$. Then for any $\varepsilon > 0$ and for any distribution $\mathcal{D}$ on $X \times [r]$,*

$$\mathbf{P}_{S \sim \mathcal{D}^m} \left[L_{\mathcal{D}}^{\text{ord}}(g_S) \leq \psi \left(\widehat{L}_S^{\text{sq}}(f_S) + \varepsilon \right) \right] \geq 1 - 4 \mathcal{N}_1(\varepsilon/16, \mathcal{F}, 2m) \cdot \exp(-m\varepsilon^2/32),$$

where $\psi(\cdot)$ is as defined in Theorem 1, and $\mathcal{N}_1$ refers to d_1 covering numbers.

Proof. The following bound on the expected squared error of the learned real-valued function is well known (cf. the uniform convergence result in Theorem 17.1 of [18]):

$$\mathbf{P}_{S \sim \mathcal{D}^m} \left[L_{\mathcal{D}}^{\text{sq}}(f_S) \leq \widehat{L}_S^{\text{sq}}(f_S) + \varepsilon \right] \geq 1 - 4 \mathcal{N}_1(\varepsilon/16, \mathcal{F}, 2m) \cdot \exp(-m\varepsilon^2/32).$$

The result then follows from Part 2 of Theorem 1. $\square$

Next we review some notions needed to present the stability bound.

Definition 1 (Loss stability). *Let $\ell(f, (x, y))$ be a loss function defined for $f : X \rightarrow \mathbb{R}$ and $(x, y) \in X \times Y$ for some $Y \subseteq \mathbb{R}$. A regression algorithm whose output on a training sample $S = ((x_1, y_1), \dots, (x_m, y_m)) \in (X \times Y)^m$ we denote by $f_S : X \rightarrow \mathbb{R}$ is said to have loss stability β with respect to ℓ (where $\beta : \mathbb{N} \rightarrow \mathbb{R}$) if for all $m \in \mathbb{N}$, $S \in (X \times Y)^m$, $1 \leq i \leq m$ and $(x'_i, y'_i) \in X \times Y$, we have for all $(x, y) \in X \times Y$,*

$$|\ell(f_S, (x, y)) - \ell(f_{S^i}, (x, y))| \leq \beta(m),$$

where S^i denotes the sequence obtained from S by replacing (x_i, y_i) with (x'_i, y'_i) .

Theorem 3 ([17]³). *Let $\ell(f, (x, y))$ be a loss function defined for $f : X \rightarrow \mathbb{R}$ and $(x, y) \in X \times Y$ for some $Y \subseteq \mathbb{R}$. Let $\mathcal{A}$ be a regression algorithm which, given as input a training sample $S \in (X \times Y)^m$, returns as output a real-valued function $f_S : X \rightarrow \mathbb{R}$, such that $0 \leq \ell(f_S, (x, y)) \leq M$ for all S and all $(x, y) \in X \times Y$. If $\mathcal{A}$ has loss stability β with respect to ℓ , then for any $0 < \delta < 1$ and for any distribution $\mathcal{D}$ on $X \times Y$, with probability at least $1 - \delta$ over the draw of S (according to $\mathcal{D}^m$),*

$$L_{\mathcal{D}}^{\ell}(f_S) \leq \widehat{L}_S^{\ell}(f_S) + \beta(m) + (2m\beta(m) + M) \sqrt{\frac{\ln(1/\delta)}{2m}},$$

where $L_{\mathcal{D}}^{\ell}$, $\widehat{L}_S^{\ell}$ denote expected and empirical ℓ -errors, defined analogously to (6–7).

³ The version presented here differs slightly (only in constants) from the result of [17]. This is due to a slight difference in definitions of stability: our definitions are in terms of changes to a training sample that consist of replacing one element in the sample with a new one, while those in [17] are in terms of changes that consist of removing one element from the sample.

Theorem 4 (Stability bound). *Let $\mathcal{A}$ be an ordinal regression algorithm which, given as input a training sample $S \in (X \times [r])^m$, learns a real-valued function $f_S : X \rightarrow [c, d]$ using a regression algorithm that has loss stability β with respect to the squared loss ℓ_{sq} (defined for $(x, y) \in X \times [r]$), and returns as output the prediction rule $g_S \equiv g_{f_S}$. Then for any $0 < \delta < 1$ and for any distribution $\mathcal{D}$ on $X \times [r]$, with probability at least $1 - \delta$ over the draw of S (according to $\mathcal{D}^m$),*

$$L_{\mathcal{D}}^{\text{ord}}(g_S) \leq \psi \left(\widehat{L}_S^{\text{sq}}(f_S) + \beta(m) + (2m\beta(m) + M) \sqrt{\frac{\ln(1/\delta)}{2m}} \right),$$

where $M = (\max(d, r) - \min(c, 1))^2$, and $\psi(\cdot)$ is as defined in Theorem 1.

Proof. Follows from Theorem 3 applied to ℓ_{sq} (note that $0 \leq \ell_{\text{sq}}(f, (x, y)) \leq M$ for all $f : X \rightarrow [c, d]$ and all $(x, y) \in X \times [r]$), and Part 2 of Theorem 1. $\square$

Example 1 (Bound 1 for SVR-based algorithm). As a further example of how Theorem 1 can be applied, consider an ordinal regression algorithm which, given a training sample $S \in (X \times [r])^m$, uses the support vector regression (SVR) algorithm to learn a real-valued function $f_S \in \mathcal{F}$ in some reproducing kernel Hilbert space (RKHS) $\mathcal{F}$, and returns the prediction rule $g_S \equiv g_{f_S}$. The SVR algorithm minimizes a regularized version of the empirical ℓ_ϵ -error $\widehat{L}_S^\epsilon(f) = \frac{1}{m} \sum_{i=1}^m \ell_\epsilon(f, (x_i, y_i))$ for some $\epsilon > 0$, where ℓ_ϵ is the ϵ -insensitive loss defined by $\ell_\epsilon(f, (x, y)) = (|f(x) - y| - \epsilon)_+$ (here $a_+ = \max(a, 0)$). If the kernel K associated with $\mathcal{F}$ satisfies $K(x, x) \leq \kappa^2 \forall x \in X$, and a regularization parameter λ is used, then the SVR algorithm has loss stability $2\kappa^2/\lambda m$ with respect to ℓ_ϵ [17], and furthermore, with $M = \max(r + \kappa\sqrt{r/\lambda}, 2\kappa\sqrt{r/\lambda})$, satisfies $0 \leq \ell_\epsilon(f_S, (x, y)) \leq M$. Therefore, applying Theorem 3 to ℓ_ϵ , observing that $\ell_{\text{abs}} \leq \ell_\epsilon + \epsilon$, and finally, using Part 1 of Theorem 1, we get that for any $0 < \delta < 1$ and for any distribution $\mathcal{D}$ on $X \times [r]$, with probability at least $1 - \delta$ over $S \sim \mathcal{D}^m$,

$$L_{\mathcal{D}}^{\text{ord}}(g_S) \leq \phi \left(\widehat{L}_S^\epsilon(f_S) + \epsilon + \frac{2\kappa^2}{\lambda m} + \left(\frac{4\kappa^2}{\lambda} + M \right) \sqrt{\frac{\ln(1/\delta)}{2m}} \right), \quad (11)$$

where $\phi(\cdot)$ is as defined in Theorem 1.

4 Direct Stability Analysis for Regression-Based Algorithms

The stability bounds for regression-based algorithms discussed above – in Theorem 4 and in Example 1 – make use of existing stability bounds for regression algorithms in a black-box manner. In this section, we directly analyze the stability of these algorithms in the context of the ordinal regression error of the final prediction rule; this allows us to obtain alternative bounds which in some cases are tighter than those obtained through the above black-box analysis. We start with an alternative definition of the stability of a regression algorithm (note that the algorithms we consider here are the same as before, *i.e.*, algorithms that learn a real-valued function f in a regression setting and then make label predictions according to g_f ; the difference will lie in the manner in which we analyze these algorithms). The approach we use is similar to that used by Bousquet & Elisseeff [17] to analyze binary classification algorithms that learn a real-valued function f and then make class predictions according to $\text{sgn}(f)$.

Definition 2 (Score stability). A regression algorithm whose output on a training sample $S = ((x_1, y_1), \dots, (x_m, y_m)) \in (X \times \mathbb{R})^m$ we denote by $f_S : X \rightarrow \mathbb{R}$ is said to have score stability ν (where $\nu : \mathbb{N} \rightarrow \mathbb{R}$) if for all $m \in \mathbb{N}$, $S \in (X \times Y)^m$, $1 \leq i \leq m$ and $(x'_i, y'_i) \in X \times Y$, we have for all $x \in X$,

$$|f_S(x) - f_{S^i}(x)| \leq \nu(m),$$

where S^i denotes the sequence obtained from S by replacing (x_i, y_i) with (x'_i, y'_i) .

The following loss, defined for $f : X \rightarrow \mathbb{R}$ and, crucially, for $(x, y) \in X \times [r]$, will play an important role in our analysis; we refer to it as the ‘clipped’ loss:

$$\begin{aligned} \ell_{\text{clip}}(f, (x, 1)) &= \begin{cases} 0, & \text{if } f(x) < 1 \\ 2(f(x) - 1), & \text{if } 1 \leq f(x) < \frac{r+1}{2} \\ r - 1, & \text{if } f(x) \geq \frac{r+1}{2}; \end{cases} \\ \ell_{\text{clip}}(f, (x, y)) &= \begin{cases} y - 1, & \text{if } f(x) < \frac{y+1}{2} \\ 2(y - f(x)), & \text{if } \frac{y+1}{2} \leq f(x) < y \\ 2(f(x) - y), & \text{if } y \leq f(x) < r - \frac{y+r}{2} \\ r - y, & \text{if } f(x) \geq \frac{y+r}{2} \end{cases} \quad \text{for } y \in \{2, \dots, r-1\}; \\ \ell_{\text{clip}}(f, (x, r)) &= \begin{cases} r - 1, & \text{if } f(x) < \frac{r+1}{2} \\ 2(r - f(x)), & \text{if } \frac{r+1}{2} \leq f(x) < r \\ 0, & \text{if } f(x) \geq r. \end{cases} \end{aligned}$$

Figure 1 shows plots of this loss for $r = 4$. A crucial property of this loss, which is immediate from the definitions (see Figure 1), is the following:

Lemma 2. For all $f : X \rightarrow \mathbb{R}$ and for all $(x, y) \in X \times [r]$, we have:

$$|g_f(x) - y| \leq \ell_{\text{clip}}(f, (x, y)) \leq 2|f(x) - y|.$$

The following lemma shows that a regression algorithm that has good score stability also has good loss stability with respect to ℓ_{clip} . The proof is similar to the proof of Lemma 2 of [19], and is omitted for lack of space.

Lemma 3. If a real-valued function learning algorithm has score stability ν (where $\nu : \mathbb{N} \rightarrow \mathbb{R}$), then it has loss stability $\beta = 2\nu$ with respect to the clipped loss ℓ_{clip} .

We are now ready for the main result of this section:

Theorem 5 (Direct stability bound). Let $\mathcal{A}$ be an ordinal regression algorithm which, given as input a training sample $S \in (X \times [r])^m$, learns a real-valued function $f_S : X \rightarrow \mathbb{R}$ using a regression algorithm that has score stability ν , and returns as output the prediction rule $g_S \equiv g_{f_S}$. Then for any $0 < \delta < 1$ and for any distribution $\mathcal{D}$ on $X \times [r]$, with probability at least $1 - \delta$ over the draw of S (according to $\mathcal{D}^m$),

$$L_{\mathcal{D}}^{\text{ord}}(g_S) \leq \widehat{L}_S^{\text{clip}}(f_S) + 2\nu(m) + (4m\nu(m) + r - 1) \sqrt{\frac{\ln(1/\delta)}{2m}},$$

where $\widehat{L}_S^{\text{clip}}$ denotes the empirical ℓ_{clip} -error with respect to S .

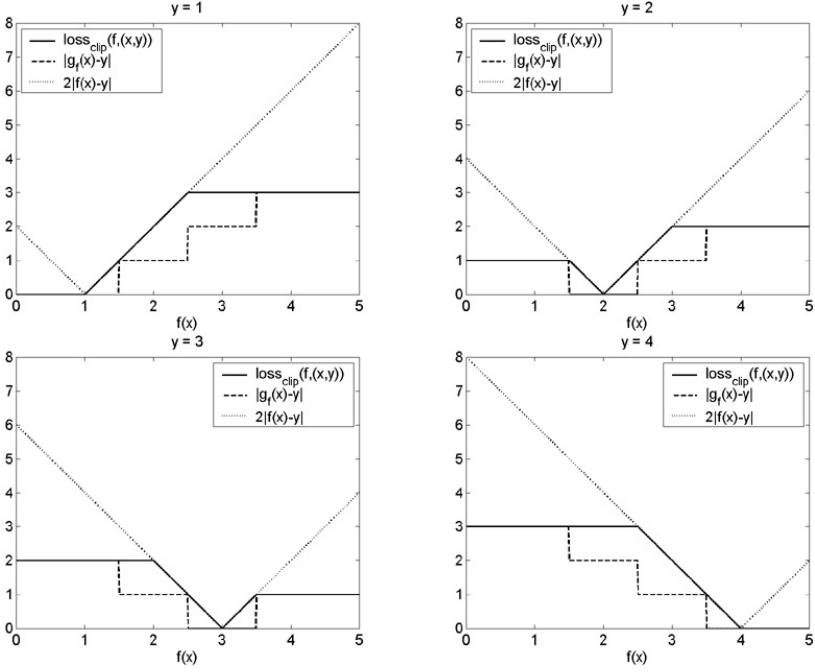


Fig. 1. Plots of the clipped loss $\ell_{\text{clip}}(f, (x, y))$, together with $\ell_{\text{ord}}(g_f, (x, y)) = |g_f(x) - y|$, and $2\ell_{\text{abs}}(f, (x, y)) = 2|f(x) - y|$, for $y = 1, 2, 3$ and 4 (each plotted as a function of $f(x)$), for an ordinal regression problem with $r = 4$

Proof. By Lemma 3, we have that $\mathcal{A}$ has loss stability 2ν with respect to ℓ_{clip} . Furthermore, we have $0 \leq \ell_{\text{clip}}(f, (x, y)) \leq r - 1$ for all $f : X \rightarrow \mathbb{R}$ and all $(x, y) \in X \times [r]$. The result then follows from Theorem 3 applied to ℓ_{clip} (with $\beta = 2\nu$ and $M = r - 1$), and from Lemma 2. $\square$

Example 2 (Bound 2 for SVR-based algorithm). For a comparison of the above bound with the black-box stability bounds of Section 3, consider again the SVR-based ordinal regression algorithm of Example 1 which, given a training sample $S \in (X \times [r])^m$, uses the SVR algorithm to learn a real-valued function $f_S \in \mathcal{F}$ in an RKHS $\mathcal{F}$, and returns the prediction rule $g_S \equiv g_{f_S}$. Under the conditions of Example 1, the SVR algorithm is known to have score stability $2\kappa^2/\lambda m$ [17]. Therefore, by Theorem 5, we get that for any $0 < \delta < 1$ and for any distribution $\mathcal{D}$ on $X \times [r]$, with probability at least $1 - \delta$ over $S \sim \mathcal{D}^m$,

$$L_{\mathcal{D}}^{\text{ord}}(g_S) \leq \widehat{L}_S^{\text{clip}}(f_S) + \frac{4\kappa^2}{\lambda m} + \left(\frac{8\kappa^2}{\lambda} + r - 1 \right) \sqrt{\frac{\ln(1/\delta)}{2m}}. \quad (12)$$

Comparing this with the bound of Example 1, we find that if the term in the argument of $\phi(\cdot)$ in Eq. (11) is smaller than $1/2$, then the direct bound above is guaranteed to give a tighter generalization result.

5 Stability Analysis for More General Algorithms

So far we have considered regression-based algorithms for ordinal regression that learn a real-valued function $f : X \rightarrow \mathbb{R}$ and then make label predictions according to g_f . We now consider more general algorithms that learn both a real-valued function $f : X \rightarrow \mathbb{R}$ and a set of thresholds $b^1 \leq \dots \leq b^{r-1} \leq b^r = \infty$; as discussed previously, the prediction rule in this case is given by

$$g_{f,b}(x) = \min_{j \in \{1, \dots, r\}} \{j : f(x) < b^j\}, \quad (13)$$

where $b \equiv (b^1, \dots, b^{r-1})$ denotes the vector of $r - 1$ thresholds (note that b^r is fixed to ∞). Recall that regression-based algorithms can be viewed as using a fixed threshold vector given by $b^j = j + \frac{1}{2}$ for $j \in \{1, \dots, r - 1\}$. In what follows, the term threshold vector will always refer to a vector of thresholds $(b^1, \dots, b^{r-1})$ that satisfies the inequalities $b_1 \leq \dots \leq b_{r-1}$.

The ordinal regression loss of a prediction rule $g_{f,b}$ on an example $(x, y) \in X \times [r]$ effectively counts the number of thresholds b^j such that $f(x)$ falls to the wrong side of b^j :

$$|g_{f,b}(x) - y| = \sum_{j=1}^{y-1} \mathbf{I}_{\{f(x) < b^j\}} + \sum_{j=y}^{r-1} \mathbf{I}_{\{f(x) \geq b^j\}}. \quad (14)$$

In general, the loss on an example $(x, y) \in X \times [r]$ in this more general setting is determined by both f and b , and as before, given a loss function $\ell(f, b, (x, y))$ in this setting, we can define the expected and empirical ℓ -errors of a function/threshold-vector pair (f, b) with respect to a distribution $\mathcal{D}$ on $X \times [r]$ and a sample $S \in (X \times [r])^m$, respectively, as follows:

$$L_{\mathcal{D}}^{\ell}(f, b) = \mathbf{E}_{(x,y) \sim \mathcal{D}}[\ell(f, b, (x, y))]; \quad \hat{L}_S^{\ell}(f, b) = \frac{1}{m} \sum_{i=1}^m \ell(f, b, (x_i, y_i)). \quad (15)$$

In order to analyze ordinal regression algorithms in this more general setting, we can extend the notion of loss stability in a straightforward manner to loss functions $\ell(f, b, (x, y))$. It is then possible to show the following result, which states that an algorithm with good loss stability with respect to such a loss ℓ has good generalization properties with respect to the error induced by ℓ . We omit the proof, which follows the proofs of similar results for classification/regression and ranking in [17, 19].

Theorem 6 (Stability bound for (f, b) -learners). *Let $\mathcal{A}$ be an ordinal regression algorithm which, given as input a training sample $S \in (X \times [r])^m$, learns a real-valued function $f_S : X \rightarrow \mathbb{R}$ and a threshold vector $b_S \equiv (b_S^1, \dots, b_S^{r-1})$, and returns as output the prediction rule $g_S \equiv g_{f_S, b_S}$. Let ℓ be any loss function in this setting such that $0 \leq \ell(f_S, b_S, (x, y)) \leq M$ for all training samples S and all $(x, y) \in X \times [r]$, and let $\beta : \mathbb{N} \rightarrow \mathbb{R}$ be such that $\mathcal{A}$ has loss stability β with respect to ℓ . Then for any $0 < \delta < 1$ and for any distribution $\mathcal{D}$ on $X \times [r]$, with probability at least $1 - \delta$ over the draw of S (according to $\mathcal{D}^m$),*

$$L_{\mathcal{D}}^{\ell}(f_S, b_S) \leq \hat{L}_S^{\ell}(f_S, b_S) + \beta(m) + (2m\beta(m) + M) \sqrt{\frac{\ln(1/\delta)}{2m}}.$$

In the case of classification and regression, and also of ranking, once a stability-based generalization result of the above form was established, it was quickly shown that there were practical learning algorithms for those problems that satisfied the desired stability conditions, and hence the result could be applied to these algorithms to obtain generalization bounds for them (indeed, we used such bounds for the support vector regression (SVR) algorithm in our study of regression-based algorithms in the previous two sections). Unfortunately, this has proved to be more difficult in our setting.

Given that many of the classification, regression and ranking algorithms for which stability analysis has been successful are regularization-based algorithms – with particular success among algorithms that perform regularization in an RKHS (such as the SVR algorithm for regression or SVMs for classification) – a natural candidate for stability analysis in our setting is the ordinal regression algorithm of Chu & Keerthi [10], which is inspired by SVMs and also performs regularization in an RKHS. However, our attempts to show stability of this algorithm have so far had limited success.

The algorithm of [10] learns a real-valued function f and a threshold vector b by minimizing a regularized upper bound on the empirical ordinal regression error. Specifically, if we associate with each label $y \in [r]$ the sign vector $(y^1, \dots, y^{r-1}) \in \{-1, +1\}^{r-1}$ defined by

$$y^j = \begin{cases} +1, & \text{if } j \in \{1, \dots, y-1\} \\ -1, & \text{if } j \in \{y, \dots, r-1\}, \end{cases} \quad (16)$$

then the loss ℓ_{CK} used by Chu & Keerthi is given by

$$\ell_{\text{CK}}(f, b, (x, y)) = \sum_{j=1}^{r-1} (1 - y^j(f(x) - b^j))_+. \quad (17)$$

Comparing with Eq. (14), it is easily verified that

$$|g_{f,b}(x) - y| \leq \ell_{\text{CK}}(f, b, (x, y)). \quad (18)$$

Given a training sample $S \in (X \times [r])^m$, the Chu-Keerthi algorithm returns a real-valued function $f_S \in \mathcal{F}$ and a threshold vector $b_S \in \mathbb{R}^{r-1}$ that satisfy⁴

$$(f_S, b_S) = \arg \min_{(f,b) \in \mathcal{F} \times \mathbb{R}^{r-1}} \widehat{L}_S^{\text{CK}}(f, b) + \lambda(\|f\|_K^2 + \|b\|^2), \quad (19)$$

where $\mathcal{F}$ is an RKHS with kernel K , $\|f\|_K$ denotes the RKHS norm of f , and $\lambda > 0$ is a regularization parameter; as discussed in [10], the vector b_S returned by the above algorithm always satisfies the necessary inequalities $b_S^1 \leq \dots \leq b_S^{r-1}$. The difficulty in showing stability of the above algorithm seems to stem from the lack of a satisfactory notion of the loss ℓ_{CK} being ‘jointly convex’ in $f(x)$ and the b_j ; in the corresponding analysis for classification/regression and ranking algorithms, convexity of the relevant loss functions in $f(x)$ allowed the desired stability conditions to be established. This difficulty appears to apply also in considering other notions of stability, such as possible extensions of score stability to the above setting.

⁴ The version here includes the regularization term for b suggested in a footnote of [10].

6 Margin Bound for Chu & Keerthi's Algorithm

We consider now a different approach to analyzing ordinal regression algorithms that learn both a real-valued function $f : X \rightarrow \mathbb{R}$ and a set of thresholds $b^1 \leq \dots \leq b^{r-1} \leq b^r = \infty$, and then make label predictions according to the prediction rule $g_{f,b}$ defined in Eq. (13) (recall that b is the threshold vector $(b^1, \dots, b^{r-1})$, and that the term threshold vector always refers to a vector of thresholds satisfying the above inequalities). In particular, we define a notion of margin for prediction rules of this form, and use this notion to derive generalization bounds for such algorithms. Our approach generalizes the margin-based analysis used in the study of classification algorithms, and results in a margin-based bound for the ordinal regression algorithm of Chu & Keerthi [10] discussed in Section 5.

Definition 3 (Margin). *Let $f : X \rightarrow \mathbb{R}$ and let $b \equiv (b^1, \dots, b^{r-1})$ be a threshold vector. Then for each $j \in \{1, \dots, r-1\}$, we define the margin of f with respect to b^j on an example $(x, y) \in X \times [r]$ as follows:*

$$\rho^j(f, b^j, (x, y)) = y^j(f(x) - b^j),$$

where y^j is as defined in Eq. (16).

Next, for $\gamma > 0$, we define the γ -margin loss of a real-valued function f and a threshold vector b on an example $(x, y) \in X \times [r]$ as follows:

$$\ell_\gamma(f, b, (x, y)) = \sum_{j=1}^{r-1} \mathbf{I}_{\{\rho^j(f, b^j, (x, y)) \leq \gamma\}}. \quad (20)$$

The ℓ_γ loss counts the number of thresholds b^j for which the corresponding margin $\rho^j(f, b^j, (x, y))$ is smaller than (or equal to) γ ; thus, comparing with Eq. (14), we immediately have that for all $\gamma > 0$,

$$|g_{f,b}(x) - y| \leq \ell_\gamma(f, b, (x, y)). \quad (21)$$

We then have the following margin-based generalization bound for (f, b) -learners. The proof makes use of a margin-based bound for binary classifiers (cf. Theorem 10.1 of [18]), applied separately to each of the $r-1$ classification tasks of predicting y^j through $\text{sgn}(f(x) - b^j)$; a union bound argument then leads to the result below. We omit the details due to lack of space.

Theorem 7 (Margin bound). *Let $\mathcal{F}$ be a class of real-valued functions on X , and let A be an ordinal regression algorithm which, given as input a training sample $S \in (X \times [r])^m$, learns a real-valued function $f_S \in \mathcal{F}$ and a threshold vector $b_S \in [-B, B]^{r-1}$, and returns as output the prediction rule $g_S \equiv g_{f_S, b_S}$. Let $\gamma > 0$. Then for any $0 < \delta < 1$ and for any distribution $\mathcal{D}$ on $X \times [r]$, with probability at least $1 - \delta$ over the draw of S (according to $\mathcal{D}^m$),*

$$L_{\mathcal{D}}^{\text{ord}}(g_S) \leq \widehat{L}_S^\gamma(f_S, b_S) + (r-1) \sqrt{\frac{8}{m} \left(\ln \mathcal{N}_\infty(\gamma/2, \mathcal{F}, 2m) + \ln \left(\frac{4B(r-1)}{\delta\gamma} \right) \right)},$$

where $\widehat{L}_S^\gamma$ denotes the empirical ℓ_γ -error, and $\mathcal{N}_\infty$ refers to d_∞ covering numbers.

Example 3 (Bound for Chu-Keerthi algorithm). Recall that the ordinal regression algorithm of Chu & Keerthi [10], described in Section 5, performs regularization in an RKHS $\mathcal{F}$ with kernel K as follows: given a training sample $S \in (X \times [r])^m$, the algorithm selects a function $f_S \in \mathcal{F}$ and a threshold vector b_S that minimize a regularized upper bound on the ordinal regression error of the resulting prediction rule $g_S \equiv g_{f_S, b_S}$. It is easy to show that the output of the Chu-Keerthi algorithm always satisfies

$$\|f_S\|_K^2 + \|b_S\|^2 \leq \frac{r-1}{\lambda},$$

where λ is the regularization parameter. Thus we have that

$$b_S \in \left[-\sqrt{\frac{r-1}{\lambda}}, \sqrt{\frac{r-1}{\lambda}} \right]^{r-1}; \quad f_S \in \mathcal{F}_{r,\lambda} \equiv \left\{ f \in \mathcal{F} \mid \|f\|_K^2 \leq \frac{r-1}{\lambda} \right\}.$$

By Theorem 7, it follows that if the covering numbers $\mathcal{N}_\infty(\gamma/2, \mathcal{F}_{r,\lambda}, 2m)$ of the effective function class $\mathcal{F}_{r,\lambda}$ can be upper bounded appropriately, then we have a generalization bound for the Chu-Keerthi algorithm. Such covering number bounds are known in a variety of settings. For example, if the kernel K satisfies $K(x, x) \leq \kappa^2 \forall x \in X$, then using a covering number bound of Zhang [20], we get that there is a constant C such that for any $\gamma > 0$, any $0 < \delta < 1$ and any distribution $\mathcal{D}$ on $X \times [r]$, with probability at least $1 - \delta$ over $S \sim \mathcal{D}^m$,

$$L_{\mathcal{D}}^{\text{ord}}(g_S) \leq \hat{L}_S^\gamma(f_S, b_S) + (r-1) \sqrt{\frac{C}{m} \left(\frac{\kappa^2(r-1)}{\lambda\gamma^2} \ln(m) + \ln \left(\frac{r-1}{\lambda\delta\gamma} \right) \right)}.$$

7 Conclusion

Our goal in this paper has been to study generalization properties of ordinal regression algorithms that learn to predict labels in a discrete but ordered set. We have focused on the absolute loss $|g(x) - y|$, for which we have obtained bounds in a variety of settings; other losses such as the squared loss $(g(x) - y)^2$ can also be useful and should be explored. Note that all such losses that measure the performance of a prediction rule g on a single example (x, y) must necessarily assume a metric on the set of labels y ; in our case, we assume the labels are in $\{1, \dots, r\}$, with the absolute distance metric (such labels are referred to as having an *interval* scale in [1]). In applications where the labels are ordered but cannot be associated with a metric, it may be more appropriate to consider losses that measure the ranking performance of g on pairs of examples [2, 16].

Another important question concerns the consistency properties of ordinal regression algorithms: whether they converge to an optimal solution, and if so, at what rate. It would be particularly interesting to study the consistency properties of algorithms that minimize a convex upper bound on the ordinal regression error, as has been done recently for classification [21, 22].

Acknowledgments

We would like to thank Yoram Singer for discussions on ordinal regression and for pointing us to the need for generalization bounds for this problem. This research was supported in part by NSF award DMS-0732334.

References

1. McCullagh, P., Nelder, J.A.: Generalized Linear Models, 2nd edn. Chapman and Hall, Boca Raton (1989)
2. Herbrich, R., Graepel, T., Obermayer, K.: Large margin rank boundaries for ordinal regression. In: *Advances in Large Margin Classifiers*, pp. 115–132. MIT Press, Cambridge (2000)
3. Kramer, S., Pfahringer, B., Widmer, G., Groeve, M.D.: Prediction of ordinal classes using regression trees. *Fundamenta Informaticae* 47, 1001–1013 (2001)
4. Frank, E., Hall, M.: A simple approach to ordinal classification. In: *Proceedings of the 12th European Conference on Machine Learning*, pp. 145–156 (2001)
5. Crammer, K., Singer, Y.: Online ranking by projecting. *Neural Computation* 17(1), 145–175 (2005)
6. Shashua, A., Levin, A.: Ranking with large margin principle: Two approaches. In: *Advances in Neural Information Processing Systems*, vol. 15, pp. 937–944. MIT Press, Cambridge (2003)
7. Harrington, E.F.: Online ranking/collaborative filtering using the perceptron algorithm. In: *Proceedings of the 20th International Conference on Machine Learning*, pp. 250–257 (2003)
8. Chu, W., Ghahramani, Z.: Gaussian processes for ordinal regression. *Journal of Machine Learning Research* 6, 1019–1041 (2005)
9. Rennie, J.D.M., Srebro, N.: Loss functions for preference levels: Regression with discrete ordered labels. In: *Proc. IJCAI Multidisciplinary Workshop on Advances in Preference Handling* (2005)
10. Chu, W., Keerthi, S.S.: Support vector ordinal regression. *Neural Computation* 19(3), 792–815 (2007)
11. Cardoso, J.S., da Costa, J.F.P.: Learning to classify ordinal data: The data replication method. *Journal of Machine Learning Research* 8, 1393–1429 (2007)
12. Waegeman, W., De Baets, B., Boullart, L.: ROC analysis in ordinal regression learning. *Pattern Recognition Letters* 29(1), 1–9 (2008)
13. Mathieson, M.J.: Ordinal models for neural networks. In: *Neural Networks in Financial Engineering*, pp. 523–536. World Scientific, Singapore (1996)
14. Crammer, K., Singer, Y.: Pranking with ranking. In: *Advances in Neural Information Processing Systems*, vol. 14, pp. 641–647. MIT Press, Cambridge (2002)
15. Shashua, A., Levin, A.: Taxonomy of large margin principle algorithms for ordinal regression problems. Technical Report 2002-39, Leibniz Center for Research, School of Computer Science and Engg., The Hebrew University of Jerusalem (2002)
16. Rajaram, S., Agarwal, S.: Generalization bounds for k -partite ranking. In: *Proceedings of the NIPS-2005 Workshop on Learning to Rank* (2005)
17. Bousquet, O., Elisseeff, A.: Stability and generalization. *Journal of Machine Learning Research* 2, 499–526 (2002)
18. Anthony, M., Bartlett, P.L.: *Learning in Neural Networks: Theoretical Foundations*. Cambridge University Press, Cambridge (1999)
19. Agarwal, S., Niyogi, P.: Stability and generalization of bipartite ranking algorithms. In: *Proceedings of the 18th Annual Conference on Learning Theory* (2005)
20. Zhang, T.: Covering number bounds of certain regularized linear function classes. *Journal of Machine Learning Research* 2, 527–550 (2002)
21. Zhang, T.: Statistical behavior and consistency of classification methods based on convex risk minimization. *The Annals of Statistics* 32, 56–85 (2004)
22. Bartlett, P.L., Jordan, M.I., McAuliffe, J.D.: Convexity, classification, and risk bounds. *Journal of the American Statistical Association* 101(473), 138–156 (2006)

Approximation of the Optimal ROC Curve and a Tree-Based Ranking Algorithm

Stéphan Cléménçon¹ and Nicolas Vayatis²

¹ LTCI, Telecom Paristech (TSI) - UMR Institut Telecom/CNRS 5141

`stephan.clemencon@telecom-paristech.fr`

² CMLA, ENS Cachan & UniverSud - UMR CNRS 8536

61, avenue du Président Wilson - 94235 Cachan cedex, France

`nicolas.vayatis@cmla.ens-cachan.fr`

Abstract. We consider the extension of standard decision tree methods to the *bipartite ranking* problem. In ranking, the goal pursued is global: define an order on the whole input space in order to have positive instances on top with maximum probability. The most natural way of ordering all instances consists in projecting the input data x onto the real line using a real-valued *scoring function* s and the accuracy of the ordering induced by a candidate s is classically measured in terms of the AUC. In the paper, we discuss the design of tree-structured scoring functions obtained by maximizing the AUC criterion. In particular, the connection with recursive piecewise linear approximation of the optimal ROC curve both in the L_1 -sense and in the L_∞ -sense is discussed.

1 Introduction

The statistical ranking problem can broadly be considered as the problem of ordering instances from a high-dimensional space. A natural approach consists in "projecting" these instances onto the real line through some real-valued scoring function. Such a function would allow to rank any list of instances in the initial space. We focus here on the setup where a binary label characterizing each instance is given. This is known as the *bipartite ranking problem* ([FISS03], [AGH⁺05], [CLV05]). A standard performance measure for a scoring function in the presence of classification data is the Receiver Operating Characteristic (ROC) curve, together with the Area Under the ROC Curve, known as the AUC (see [Ega75], [HM82]). But, since their introduction, ROC curves and the AUC have served mostly for validation and not as the target for optimization principles. More recently, several aspects of AUC maximization have been discussed in the machine learning literature ([CM04], [Rak04], [YDMW03]) and also from a statistical learning perspective ([AGH⁺05], [CLV05], [CLV08]). A particular class of learning algorithms will be at the center of the present paper, namely decision trees in the spirit of CART [BFOS84]. The investigation of decision trees in the context of ranking was initiated only recently in the field of machine learning ([FFHO02], [PD03], [XZW06]). In the present work, we propose a tree-based ranking algorithm and exhibit statistical results which

guarantee its performance. The method builds adaptively a scoring function from training data with an ROC curve close to the optimal one. The approach relies on linear-by-parts approximations of the ROC curve which correspond to finite-dimensional (piecewise constant) approximations of optimal scoring functions. As the ROC curve provides a performance measure of functional nature, the approximation can be conceived in a variety of ways depending on the topology equipping the space of ROC curves. For instance, the AUC is related to the L_1 -distance but we will also consider convergence to the optimal ROC curve in a stronger sense described by the L_∞ -distance. A recursive implementation of the approximation procedure naturally leads to a tree-like structure for underlying scoring functions. We suggest that such a tree-based ranker could serve as a weak learner and feed a boosting-type algorithm such as RankBoost ([FISS03]). We also provide mathematical results in terms of the approximation error, statistical consistency, and convergence rates.

The paper is organized as follows. In Section 2, we present a general approach for assessing optimality in the bipartite ranking problem. We also recall the main concepts and discuss the issue of AUC maximization. In Section 3, we discuss the approximation of the optimal ROC curve with piecewise constant scoring functions and provide an adaptive tree-structured recursive procedure for which an approximation error result is established. This approximation scheme can be carried out over empirical data by the means of the TREERANK algorithm described in Section 4. The statistical consistency of the method is also studied.

2 Setup and Optimal ROC Curves

We study the ranking problem for classification data with binary labels. This is also known as the bipartite ranking problem ([FISS03]). The data are assumed to be generated as copies of a random pair $(X, Y) \in \mathcal{X} \times \{-1, +1\}$ where X is a random descriptor living in the measurable space $\mathcal{X}$ and Y represents its binary label (relevant vs. irrelevant, healthy vs. sick, ...). We denote by $P = (\mu, \eta)$ the distribution of (X, Y) , where μ is the marginal distribution of X and η is the *regression function* (up to an affine transformation): $\eta(x) = \mathbb{P}\{Y = 1 \mid X = x\}$, $x \in \mathcal{X}$. We will also denote by $p = \mathbb{P}\{Y = 1\}$ the proportion of positive labels. In the sequel, we assume that the distribution μ is absolutely continuous with respect to Lebesgue measure. The goal of a ranking procedure is to provide an ordering of the elements of $\mathcal{X}$ based on their labels. We expect to end up with a list with positive labels at the top and negative labels at the bottom. However, label information does not permit to derive a total order on $\mathcal{X}$ and among relevant (positively labelled) objects in $\mathcal{X}$, some might be more relevant than others. In short, a good ranking should preserve the ordering induced by the likelihood of having a positive label, namely the regression function η . We consider the approach where the ordering can be derived by the means of a *scoring function* $s : \mathcal{X} \rightarrow \mathbb{R}$. The following definition sets the goal of learning methods in the setup of bipartite ranking.

Definition 1 (Optimal scoring functions). A scoring function $s^* : \mathcal{X} \rightarrow \mathbb{R}$ is said to be optimal if it induces the same ordering over $\mathcal{X}$ as the function $\eta(x) = \mathbb{P}\{Y = 1 \mid X = x\}$, $\forall x \in \mathcal{X}$. In other words:

$$\forall x, x' \in \mathcal{X}, \quad s^*(x) - s^*(x') > 0 \Rightarrow \eta(x) - \eta(x') > 0.$$

According to the previous definition, the next proposition is a trivial characterization of the class of optimal scoring functions.

Proposition 1. The class of optimal scoring functions is given by the set

$$\mathcal{S}^* = \{ s^* = T \circ \eta \mid T : [0, 1] \rightarrow \mathbb{R} \text{ strictly increasing} \}.$$

We now recall the concept of ROC curve and explain why it is a natural choice of performance measure for the ranking problem with classification data. In this section, we only consider *true* ROC curves which correspond to the situation where the underlying distribution is known. Before recalling the definition, we need to introduce some notations. For a given scoring rule s , the conditional cumulative distribution functions (cdf) of the random variable $s(X)$ given $Y = +1$ and $Y = -1$ are denoted respectively by G_s and H_s . We also denote by $\bar{G}_s(z) = 1 - G_s(z)$ and $\bar{H}_s(z) = 1 - H_s(z)$ the residual conditional cdfs of the r.v. $s(X)$. When $s = \eta$, we shall denote the previous functions by G^* , H^* , $\bar{G}^*$, $\bar{H}^*$ respectively. We will also use the notation, for all t :

$$\alpha(t) = \bar{H}^*(t) \text{ and } \beta(t) = \bar{G}^*(t).$$

We introduce the notation $Q(Z, \alpha)$ to denote the quantile of order $1 - \alpha$ for the distribution of a random variable Z conditioned on the event $Y = -1$. In particular, the following quantile will be of interest: $Q^*(\alpha) = Q(\eta(X), \alpha) = \bar{H}^{*-1}(\alpha)$, where we have used here the notion of generalized inverse F^{-1} of a c  dl  g function F : $F^{-1}(z) = \inf\{t \in \mathbb{R} \mid F(t) \geq z\}$.

A classical way to assess the performance of a scoring function s in separating the two populations (positive vs. negative labels) is the *Receiver Operating Characteristic* known as the ROC curve ([Ega75]).

Definition 2 (True ROC curve). The ROC curve of a scoring function s is the parametric curve: $z \mapsto (\bar{H}_s(z), \bar{G}_s(z))$ for thresholds $z \in \mathbb{R}$. It can also be defined as the plot of the function:

$$\alpha \in [0, 1] \mapsto \bar{G}_s \circ \bar{H}_s^{-1}(\alpha) = \bar{G}_s(Q(s(X), \alpha)) = \text{ROC}(s, \alpha).$$

By convention, points of the curve corresponding to possible jumps (due to possible degenerate points for H_s or G_s) are connected by line segments, in order that the ROC curve is always continuous. For $s = \eta$, we take the notation $\text{ROC}^*(\alpha) = \text{ROC}(\eta, \alpha)$.

The residual cdf $\bar{G}_s$ is also called the true positive rate and $\bar{H}_s$ is the false positive rate, so that the ROC curve is the plot of the true positive rate against

the false positive rate. The ROC curve provides a visual tool for comparing the ranking performance of two scoring rules. Optimal scoring functions are those for which the ROC curve dominates all the others for all $\alpha \in (0, 1)$. The next proposition highlights the fact that the ROC curve is relevant when evaluating performance in the bipartite ranking problem.

Proposition 2. *The class $\mathcal{S}^*$ provides the best possible ranking with respect to the ROC curve criterion. As a matter of fact, for any scoring function s , we have: $\forall \alpha \in (0, 1)$, $\text{ROC}^*(\alpha) \geq \text{ROC}(s, \alpha)$, and $\forall s^* \in \mathcal{S}^*$, $\text{ROC}(s^*, \alpha) = \text{ROC}^*(\alpha)$. Moreover, if we set:*

$$R_\alpha^* = \{x \in \mathcal{X} \mid \eta(x) > Q^*(\alpha)\} \text{ and } R_{s,\alpha} = \{x \in \mathcal{X} \mid s(x) > Q(s(X), \alpha)\}$$

then, for any s and any α such that $Q^*(\alpha) < 1$:

$$\text{ROC}^*(\alpha) - \text{ROC}(s, \alpha) = \frac{\mathbb{E}(|\eta(X) - Q^*(\alpha)| \mathbb{I}\{X \in R_\alpha^* \Delta R_{s,\alpha}\})}{p(1 - Q^*(\alpha))}$$

where Δ denotes the symmetric difference between sets.

The last statement reveals that the pointwise difference between the dominating ROC curve and the one related to a candidate scoring function s may be interpreted as the error made in recovering the specific level set R_α^* through $R_{s,\alpha}$. A simple consequence of the previous result (and its proof) is that the one-dimensional statistic $\eta(X)$ (instead of the supposedly high-dimensional observation X) suffices to recover the optimal ROC curve. In other words, projecting the original data onto $(0, 1)$ using the regression function leaves the ROC curve untouched. The following result will be needed later.

Proposition 3 (Derivative of the ROC). *We assume that the optimal ROC curve is differentiable. Then, we have, for any α such that $Q^*(\alpha) < 1$:*

$$\frac{d}{d\alpha} \text{ROC}^*(\alpha) = \frac{1-p}{p} \cdot \frac{Q^*(\alpha)}{1 - Q^*(\alpha)}.$$

Although the ROC curve is a useful graphical tool for evaluating the performance of a scoring function, its use as the target of an optimization strategy to estimate ROC-optimal scoring functions turns out to be quite challenging. Indeed, selecting a scoring function by empirical maximization of the ROC curve over a class $\mathcal{S}$ of scoring functions is a highly complex task because of the functional nature of the ROC curve criterion. Of course, the closer to ROC^* the ROC curve of a candidate scoring function $s \in \mathcal{S}$, the more pertinent the ranking induced by s . However, various metrics can be considered for measuring the distance between curves. We focus on two essential cases:

$$\begin{aligned} \text{the } L_1 \text{ metric: } d_1(s^*, s) &= \int_0^1 (\text{ROC}(s^*, \alpha) - \text{ROC}(s, \alpha)) \, d\alpha \\ \text{the } L_\infty \text{ metric: } d_\infty(s^*, s) &= \sup_{\alpha \in (0,1)} (\text{ROC}(s^*, \alpha) - \text{ROC}(s, \alpha)) \end{aligned}$$

Remark 1. In order to avoid a possible confusion due to the notation, we bring to the reader's attention the fact that d_1 and d_∞ do not denote metrics on the space of scoring functions $\mathcal{S}$, but on the set of ROC curves.

As far as we know, the L_∞ metric has not been considered in the literature yet, while the weaker L_1 -metric actually corresponds to a very popular criterion which is at the heart of most practical ranking methods. This is known as the Area Under an ROC Curve (or AUC, see [HM82]).

Definition 3 (AUC). *For any scoring function s , define the AUC as:*

$$\text{AUC}(s) = \int_0^1 \text{ROC}(s, \alpha) \, d\alpha ,$$

and set $\text{AUC}^* = \text{AUC}(\eta)$. We then have: $d_1(s^*, s) = \text{AUC}^* - \text{AUC}(s)$.

When it comes to finding a scoring function, based on empirical data, which will perform well with respect to the AUC criterion, various strategies can be considered. A possible angle is the *plug-in* approach ([DGL96]). The idea of plug-in consists in using an estimate $\hat{\eta}$ of the regression function as a scoring function. It is expected that, whenever $\hat{\eta}$ is close to η in a certain sense, then $\text{ROC}(\hat{\eta}, \cdot)$ and ROC^* are also close.

Proposition 4. *Consider $\hat{\eta}$ an estimator of η . We have:*

$$\text{AUC}^* - \text{AUC}(\hat{\eta}) \leq \frac{1}{p(1-p)} \mathbb{E} (|\hat{\eta}(X) - \eta(X)|) .$$

Moreover if H^* has a density which is bounded by below on $[0, 1]$: $\exists c > 0$ such that $\forall \alpha \in [0, 1]$, $\frac{dH^*}{d\alpha}(\alpha) \geq c^{-1}$. Then, we have: $\forall \alpha \in [0, 1]$ such that $Q^*(\alpha) < 1$,

$$\text{ROC}^*(\alpha) - \text{ROC}(\hat{\eta}, \alpha) \leq \frac{c \mathbb{E} (|H^*(\eta(X)) - H_{\hat{\eta}}(\hat{\eta}(X))|)}{p(1 - Q^*(\alpha))} .$$

However, plug-in rules face difficulties when dealing with high-dimensional data ([GKKW02]). Another drawback of plug-in rules is that they are not consistent with respect to the supremum norm. This observation provides an additional motivation for exploring algorithms based on empirical AUC maximization. A nice feature of the AUC performance measure is that it may be interpreted in a probabilistic fashion and we refer to [CLV08] for a systematic study of related empirical and convex risk minimization strategies which involve U -statistics. From a machine learning perspective, there is a growing literature in which existing algorithms are adapted in order to perform AUC optimization (such as, for instance: [CM04], [Rak04], [YDMW03]). The tree-based method we propose in the sequel consists in an adaptive recursive strategy for building a piecewise constant scoring function with nearly maximum AUC.

3 Piecewise Linear Approximation of the Optimal ROC Curve

In this section, we assume that the distribution, and hence the optimal ROC curve, are known. We also assume that the optimal ROC curve is differentiable and concave (see [CV08] for a discussion on these assumptions). We consider the problem of building, in a stepwise manner, a scoring function whose ROC curve is a piecewise linear approximation/interpolation of the optimal curve ROC^* .

3.1 Piecewise Constant Scoring Functions

When it comes to approximations of the optimal s^* , a natural idea is to introduce discrete versions and to replace the expectation by a finite sum. We now introduce D -representation of a piecewise constant scoring function where the ' D ' stands for 'disjoint'.

Definition 4 (D -representation). *Let $N \geq 1$. The D -representation of a piecewise constant scoring function s_N taking values in $\{a_1, \dots, a_N\}$ is given by: $\forall x \in \mathcal{X}$, $s_N(x) = \sum_{j=1}^N a_j \mathbb{I}\{x \in C_j\}$, for some decreasing sequence $(a_j)_{j \geq 1}$ and some partition $\mathcal{C}_N = (C_j)_{1 \leq j \leq N}$ of $\mathcal{X}$. We define $\mathcal{S}_N$ to be the class of such scoring functions.*

We now list some obvious properties of piecewise constant scoring function.

Proposition 5. *Consider some piecewise constant scoring function $s_N \in \mathcal{S}_N$.*

- (i) *The ROC curve of s_N is piecewise linear with N linear parts.*
- (ii) *The ROC curve of s_N does not depend on the particular values of the sequence $(a_j)_{j \geq 1}$ appearing in its D -representation but only on their ordering.*

Our purpose in this section is to design an iterative procedure which outputs a piecewise constant scoring function $s_N \in \mathcal{S}_N$ whose ROC curve is as close as possible to the optimal ROC^* . Closeness between ROC curves will be measured both in terms of AUC and in the L_∞ -sense. The iterative procedure described in the sequel satisfies the following approximation error result.

Proposition 6. *Assume that the optimal ROC curve is twice differentiable and concave and that its second derivative takes its values in a bounded interval which does not contain zero. There exists a sequence of piecewise constant scoring functions $(s_N)_{N \geq 1}$ such that, for any $N \geq 1$, $s_N \in \mathcal{S}_N$ and:*

$$d_1(s^*, s_N) \leq C \cdot N^{-2} \text{ and } d_\infty(s^*, s_N) \leq C \cdot N^{-2},$$

where the constant C depends only on the distribution.

The approximation rate of $O(N^{-2})$ is actually reached by any piecewise linear approximation provided that the mesh length is of order $O(N^{-1})$. This result is well-known folklore in approximation theory, see [DL93]. We underline that the

piecewise linear approximation method we describe next is adaptive in the sense that breakpoints are not fixed in advance and strongly depend on the target curve (which suggests that this scheme possibly yields a sharper constant C). It highlights the explicit relationship between the approximation of the optimal ROC curve and the corresponding piecewise constant scoring function. The ranking algorithm proposed in the sequel (Section 4) will appear as a statistical version of this variable knot approximation, where the unknown quantities driving the recursive partitioning will be replaced by their empirical counterparts.

3.2 One-Step Approximation to the Optimal ROC Curve

We now provide some insights on the general construction by describing the one-step modification of a given piecewise constant scoring function s_N . Here, modifications are picked up in the class $\mathcal{G}$ of level sets of the regression function η : $\mathcal{G} = \{\{x \in \mathcal{X} : \eta(x) > t\} : t \in (0, 1)\}$.

Definition 5 (One-step L_1 approximation). *Given $s_N \in \mathcal{S}_N$, we define:*

$$\sigma_N = \arg \max_{\sigma \in \mathcal{G}} d_1(s_N, s_N + \sigma).$$

Then, the one-step approximation sequence to some optimal scoring function s^* is defined as the sequence $(s_N)_{N \geq 1}$ of scoring functions such that:

$$s_1 = \mathbb{I}_{\mathcal{X}} \text{ and } \forall N \geq 1, s_{N+1} = s_N + \sigma_N.$$

Now we consider a different representation of piecewise constant scoring functions based on increasing subsets. A constructive procedure will rely on a particular choice of subsets $(R_j)_{j \geq 1}$. We focus on partitions with sets of the form: $R_j = \{x \in \mathcal{X} : \eta(x) > u_j\}$ for some positive decreasing sequence $(u_j)_{j \geq 1}$ with $u_1 > 0$.

First iteration. We initialize the procedure for $N = 1$ with the scoring function:

$$\forall x \in \mathcal{X}, \quad s_1(x) = \mathbb{I}\{x \in \mathcal{X}\} \equiv 1,$$

which ranks all instances equally. It is clear that adding up the indicator of any region of the form $\{\eta(x) > t\}$ for some $t \in (0, 1)$ would provide a piecewise linear approximation of the optimal ROC curve. We choose the one which maximizes the AUC criterion. It is easy to see that the one-step L_1 approximation at the first iteration is given by the piecewise constant scoring function:

$$\forall x \in \mathcal{X}, \quad s_2(x) = \mathbb{I}\{x \in \mathcal{X}\} + \mathbb{I}\{\eta(x) > p\}.$$

where $p = \mathbb{P}\{Y = 1\}$. We also have: $(d\beta/d\alpha)(p) = 1$.

It is noteworthy that the one-step L_1 approximation obtained by optimization of the AUC criterion is the same as the one obtained through optimization of the sup-norm. One can easily check that

$$\arg \max_{\sigma \in \mathcal{G}} d_1(s_1, s_1 + \sigma) = \arg \max_{\sigma \in \mathcal{G}} d_\infty(s_1, s_1 + \sigma).$$

N -th iteration. Now consider a piecewise constant scoring function $s_N \in \mathcal{S}_N$. The ROC curve of s_N is a broken line with N linear pieces defined by the sequence of points $((\alpha_j, \beta_j))_{0 \leq j \leq N}$ where $(\alpha_0, \beta_0) = (0, 0)$ and $(\alpha_N, \beta_N) = (1, 1)$. For a fixed j , we look for the optimal splitting which would increase the AUC by adding a knot $(\alpha(t), \beta(t))$ such that $\alpha(t)$ is between α_j and α_{j+1} . We take the notation

$$s_{N+1,t}^{(j)}(x) = s_N(x) + \mathbb{I}\{\eta(x) > t\}, \text{ with } t \in (Q^*(\alpha_{j+1}), Q^*(\alpha_j)).$$

The AUC can then be written, for some constant c_j , as:

$$A_{N+1}(t) = \text{AUC}(s_{N+1,t}^{(j)}) = c_j + \frac{1}{2}(\alpha_{j+1} - \alpha_j)\beta(t) - \frac{1}{2}\alpha(t)(\beta_{j+1} - \beta_j),$$

which is maximized at t^* such that:

$$d\beta(t^*) = \left(\frac{\beta_{j+1} - \beta_j}{\alpha_{j+1} - \alpha_j} \right) d\alpha(t^*) .$$

Set $\alpha_j^* = \alpha(t^*)$ and get, thanks to Proposition 3, the following relationship:

$$\frac{1-p}{p} \cdot \frac{Q^*(\alpha_j^*)}{1 - Q^*(\alpha_j^*)} = \frac{\beta_{j+1} - \beta_j}{\alpha_{j+1} - \alpha_j} .$$

This leads to a one-step optimal splitting point (α_j^*, β_j^*) on the ROC curve such that: $\alpha_j^* = \bar{H}^*(\Delta_j)$ and $\beta_j^* = \bar{G}^*(\Delta_j)$ where

$$\Delta_j = \frac{p(\beta_{j+1} - \beta_j)}{(1-p)(\alpha_{j+1} - \alpha_j) + p(\beta_{j+1} - \beta_j)} = t^* .$$

Remark 2 (INTERPRETATION IN TERMS OF PARTITIONS). The insertion of the new knot (α_j^*, β_j^*) is materialized by the splitting of subset R_{j+1} with a subset R_j^* containing R_j and we have $R_j^* = \{x \in \mathcal{X} : \eta(x) > Q^*(\alpha_j^*)\}$, while $R_j = \{x \in \mathcal{X} : \eta(x) > Q^*(\alpha_j)\}$. In terms of D -representations, we can write: $s_N = \sum_{j=1}^N (N-j+1) \mathbb{I}_{C_j}$ where $C_j = \{x \in \mathcal{X} : Q^*(\alpha_{j+1}) < \eta(x) \leq Q^*(\alpha_j)\}$. After the splitting, in the new partition, the set C_{j+1} is replaced by C_j^* and $C_{j+1} \setminus C_j^*$ where $C_{j+1} = \{x \in \mathcal{X} : Q^*(\alpha_{j+1}) < \eta(x) \leq Q^*(\alpha_j^*)\}$.

The previous computations quantify the improvement in terms of AUC after adding one knot for each linear part of the ROC curve at step N . Instead of sticking to one-step approximations, we can introduce an approximation scheme which will add 2^N knots after the N -th iteration.

3.3 A Tree-Structured Recursive Approximation Scheme

We now turn to the full recursive procedure. At each step, an adaptively chosen knot is added between all consecutive points of the current meshgrid. We take $N = 2^D$ with $D \geq 0$ and we describe iterations over D for constructing a

sequence of piecewise constant scoring functions. It will be easier to work with D -representations of the form: $\forall x \in \mathcal{X}$, $s_D(x) = \sum_{k=0}^{2^D-1} (2^D - k) \mathbb{I}\{x \in C_{D,k}\}$, where, for fixed D , the class of sets $(C_{D,k})_{0 \leq k \leq 2^D-1}$ is a disjoint partition of $\mathcal{X}$. We will use the following notations:

$$\alpha(C) = \mathbb{P}\{X \in C \mid Y = -1\} \text{ and } \beta(C) = \mathbb{P}\{X \in C \mid Y = 1\}.$$

The iterative procedure goes as follows.

Initialization ($d = 0$ and $d = 1$). For the extremal points, we set:

$$\forall d \in \mathbb{N}, \alpha_{d,0}^* = \beta_{d,0}^* = 0 \quad \text{and} \quad \alpha_{d,2^d}^* = \beta_{d,2^d}^* = 1,$$

and for the first iteration points ($d = 1$): $\alpha_{1,1}^* = \bar{H}^*(p)$ and $\beta_{1,1}^* = \bar{G}^*(p)$.

From d to $d + 1$, for $d \geq 1$. Let the collection $\{(\alpha_{d,k}^*, \beta_{d,k}^*)\}_{k=0, \dots, 2^d-1}$ be given. On each interval $(\alpha_{d,k}^*, \alpha_{d,k+1}^*)$, we apply the one-step approximation. Hence, the new point is given by: $\alpha_{d+1,2k+1}^* = \bar{H}^*(\Delta_{d+1,2k+1}^*)$ and $\beta_{d+1,2k+1}^* = \bar{G}^*(\Delta_{d+1,2k+1}^*)$, where

$$\Delta_{d+1,2k+1}^* = \frac{p(\beta_{d,k+1}^* - \beta_{d,k}^*)}{(1-p)(\alpha_{d,k+1}^* - \alpha_{d,k}^*) + p(\beta_{d,k+1}^* - \beta_{d,k}^*)}.$$

Moreover, the previous cut-off point is renamed: $\alpha_{d+1,2k}^* = \alpha_{d,k}^*$ and $\beta_{d+1,2k}^* = \beta_{d,k}^*$. Set also $\Delta_{d+1,2k}^* = \Delta_{d,k}^*$. Note that, for each level d , the resulting partition is given by the class of sets: $C_{d,k}^* = \{x \in \mathcal{X} : \Delta_{d,k}^* < \eta(x) \leq \Delta_{d,k+1}^*\}$, for all $k = 0, \dots, 2^d - 1$ with the convention that $\Delta_{d,0}^* = 0$ and $\Delta_{d,2^d}^* = 1$ for all $d \geq 0$. We also define the sets $R_{d,k}^*$ by: $R_{d,k}^* = C_{d,k}^* \cup R_{d,k-1}^*$ with $R_{d,0}^* = C_{d,0}^*$.

Remark 3 (A TREE-STRUCTURED RECURSIVE INTERPOLATION SCHEME). A nice feature of the recursive approximation procedure is its binary-tree structure. Owing to their crucial practical advantages regarding implementation and interpretation, tree-structured decision rules have been proved useful for a wide range of statistical tasks and are in particular among the most popular methods for regression and classification (we refer to Chapter 20 in [DGL96] for an excellent account of tree decision rules in the context of classification).

4 A Tree-Structured Weak Ranker

It is time to exploit the theory developed in the previous sections to deal with empirical data. We formulate a practical algorithm which implements a top-down strategy to build a binary tree-structured scoring function. This algorithm mimics the ideal recursive approximation procedure of the optimal ROC curve from Section 3, where probabilities are replaced by their empirical counterparts.

4.1 The TREERANK Algorithm

Assume now that a training dataset $\mathcal{D}_n = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$ of n independent copies of the pair (X, Y) is available. Set $n_+ = \sum_{i=1}^n \mathbb{I}\{Y_i = 1\} = n - n_-$. We introduce the following data-based quantities, for any subset C :

$$\hat{\alpha}(C) = \frac{1}{n_-} \sum_{i=1}^n \mathbb{I}\{X_i \in C, Y_i = -1\} \text{ and } \hat{\beta}(C) = \frac{1}{n_+} \sum_{i=1}^n \mathbb{I}\{X_i \in C, Y_i = +1\}$$

which correspond respectively to the empirical false positive rate and the empirical true positive rate of a classifier predicting $+1$ on the set C . For notational convenience, we set $\alpha_{d,0} = \beta_{d,0} = 0$ and $\alpha_{d,2^d} = \beta_{d,2^d} = 1$ for all $d \geq 0$. We assume that we are given a class $\mathcal{C}$ of subsets of $\mathcal{X}$.

TREERANK ALGORITHM

1. **Initialization.** Set $C_{0,0} = \mathcal{X}$.
2. **Iterations.** For $d = 0, \dots, D-1$ and for $k = 0, \dots, 2^d - 1$:
 - (a) (OPTIMIZATION STEP.) Set the entropy measure:

$$A_{d,k+1}(C) = (\alpha_{d,k+1} - \alpha_{d,k})\hat{\beta}(C) - (\beta_{d,k+1} - \beta_{d,k})\hat{\alpha}(C).$$

Find the best subset $C_{d+1,2k}$ of rectangle $C_{d,k}$ in the AUC sense:

$$C_{d+1,2k} = \arg \max_{C \in \mathcal{C}, C \subset C_{d,k}} A_{d,k+1}(C).$$

Then, set $C_{d+1,2k+1} = C_{d,k} \setminus C_{d+1,2k}$.

- (b) (UPDATE.) Set

$$\alpha_{d+1,2k+1} = \alpha_{d,k} + \hat{\alpha}(C_{d+1,2k}) \text{ and } \beta_{d+1,2k+1} = \beta_{d,k} + \hat{\beta}(C_{d+1,2k})$$

as well as $\alpha_{d+1,2k+2} = \alpha_{d,k+1}$ and $\beta_{d+1,2k+2} = \beta_{d,k+1}$.

3. **Output.** After D iterations, get the piecewise constant scoring function:

$$s_D(x) = \sum_{k=0}^{2^D-1} (2^D - k) \mathbb{I}\{x \in C_{D,k}\}$$

The main features of the TREERANK algorithm are listed in the following remarks.

Remark 4 (READING THE RANKS). The resulting ranking induced by the scoring function s_D may be read from the left to the right looking at the terminal nodes

Remark 5 (A SIMPLE STOPPING CRITERION). If there is more than one subrectangle solution in the OPTIMIZATION STEP, take the larger. Hence, if there is no improvement in terms of AUC maximization when splitting the current rectangle $C_{d,k}$, set $C_{d+1,2k} = C_{d,k}$, so that $C_{d+1,2k+1} = \emptyset$.

Remark 6 (ON THE SPLITTING CRITERION). The splitting criterion $A_{d,k}$ comes from the expression of $A_{N+1}(t)$ in Subsection 3.2. In the context of classification,

this splitting rule has been considered previously in [FFHO02]. We point out that, in contrast to tree-based classification methods, such as CART, the splitting criterion depends on the node through the parent's false and true positive rates $\hat{\alpha}(C)$ and $\hat{\beta}(C)$. This can be explained by the fact that the goal pursued in the ranking problem is global: one attempts to order all input data with respect to each other.

4.2 Consistency of TREERANK and Rate Bounds

We now provide a consistency result for the class of partitions induced by the TREERANK algorithm. The formulation (and the proof) mimics Theorem 21.2 from [DGL96].

Theorem 1. *We consider scoring functions s_n corresponding to partitions $\mathcal{F}_n$ of $\mathcal{X}$. We assume that the $\mathcal{F}_n$'s are random partitions of $\mathcal{X}$ resulting from runs of TREERANK with training sets of size n . We also assume that $\mathcal{X}$ is bounded and that the partitions $\mathcal{F}_n$ belong to a VC class of sets with VC dimension V , for any n and any training set. If the diameter of any cell of $\mathcal{F}_n$ goes to 0 when n tends to infinity, then we have that:*

$$\text{AUC}(s^*) - \text{AUC}(s_n) = d_1(s^*, s_n) \rightarrow 0$$

almost surely, as n goes to ∞ .

If we have, in addition that H^* has a density which is bounded by below on $[0, 1]$ and that, for any α , $Q^*(\alpha) < 1 - \epsilon$, for some $\epsilon > 0$, then:

$$d_\infty(s^*, s_n) \rightarrow 0$$

almost surely, as n goes to ∞ .

Remark 7 (BOUNDEDNESS OF $\mathcal{X}$). This assumption is a simplification which can be removed at the cost of a longer proof (the core of the argument can be found in [DGL96]).

Remark 8 (COMPLEXITY ASSUMPTION). Instead of assuming a finite VC dimension, a weaker assumption on the combinatorial entropy of the class of partitions may be provided (again check [DGL96] for this refinement).

Under additional assumptions, a rate bound can be established for the scoring function produced by TREERANK.

Theorem 2. *Assume that conditions of Proposition 6 are fulfilled. Suppose that the class $\mathcal{C}$ of subset candidates contains all level sets R_α^* , $\alpha \in [0, 1]$ and is stable under intersections, i.e. $\forall (C, C') \in \mathcal{C}^2: C \cap C' \in \mathcal{C}$. Assume furthermore that $\mathcal{C}$ has finite VC dimension V .*

- (i) *For all $\delta > 0$, there exists a constant c_0 and universal constants c_1, c_2 such that, with probability at least $1 - \delta$, we have for all $D \geq 1, n \in \mathbb{N}$:*

$$d_1(\hat{s}_D, s_D) \leq c_0^D \left\{ \left(\frac{c_1^2 V}{n} \right)^{\frac{1}{2D}} + \left(\frac{c_2^2 \log(1/\delta)}{n} \right)^{\frac{1}{2D}} \right\},$$

and

$$d_\infty(\hat{s}_D, s_D) \leq c_0^D \left\{ \left(\frac{c_1^2 V}{n} \right)^{\frac{1}{2(D+1)}} + \left(\frac{c_2^2 \log(\frac{1}{\delta})}{n} \right)^{\frac{1}{2(D+1)}} \right\}.$$

(ii) Choosing $D = D_n$ so that $D_n \sim \sqrt{\log n}$, as $n \rightarrow \infty$. Then, for all $\delta > 0$, there exist constants c and κ such that, with probability at least $1 - \delta$, we have for all $n \in \mathbb{N}$:

$$d_i(\hat{s}_{D_n}, s^*) \leq c \exp(-\kappa \sqrt{\log n}), \quad i \in \{1, \infty\}.$$

References

- [AGH⁺05] Agarwal, S., Graepel, T., Herbrich, R., Har-Peled, S., Roth, D.: Generalization bounds for the area under the ROC curve. *Journal of Machine Learning Research* 6, 393–425 (2005)
- [BFOS84] Breiman, L., Friedman, J., Olshen, R., Stone, C.: *Classification and Regression Trees*. Wadsworth and Brooks (1984)
- [CLV05] Cléménçon, S., Lugosi, G., Vayatis, N.: Ranking and scoring using empirical risk minimization. In: Auer, P., Meir, R. (eds.) *COLT 2005*. LNCS (LNAI), vol. 3559, pp. 1–15. Springer, Heidelberg (2005)
- [CLV08] Cléménçon, S., Lugosi, G., Vayatis, N.: Ranking and empirical risk minimization of U-statistics. *The Annals of Statistics* 36, 844–874 (2008)
- [CM04] Cortes, C., Mohri, M.: AUC optimization vs. error rate minimization. In: Thrun, S., Saul, L., Schölkopf, B. (eds.) *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge (2004)
- [CV08] Cléménçon, S., Vayatis, N.: Tree-structured ranking rules and approximation of the optimal ROC curve. Technical Report hal-00268068, HAL (2008)
- [DGL96] Devroye, L., Györfi, L., Lugosi, G.: *A Probabilistic Theory of Pattern Recognition*. Springer, Heidelberg (1996)
- [DL93] Devore, R., Lorentz, G.: *Constructive Approximation*. Springer, Heidelberg (1993)
- [Ega75] Egan, J.P.: *Signal Detection Theory and ROC Analysis*. Academic Press, London (1975)
- [FFHO02] Ferri, C., Flach, P.A., Hernández-Orallo, J.: Learning decision trees using the area under the roc curve. In: *ICML 2002: Proceedings of the Nineteenth International Conference on Machine Learning*, pp. 139–146. Morgan Kaufmann Publishers Inc., San Francisco (2002)
- [FISS03] Freund, Y., Iyer, R.D., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research* 4, 933–969 (2003)
- [GKKW02] Györfi, L., Köhler, M., Krzyzak, A., Walk, H.: *A Distribution-Free Theory of Nonparametric Regression*. Springer, Heidelberg (2002)

- [HM82] Hanley, J.A., McNeil, J.: The meaning and use of the area under a ROC curve. *Radiology* 143, 29–36 (1982)
- [PD03] Provost, F., Domingos, P.: Tree induction for probability-based ranking. *Machine Learning* 52(3), 199–215 (2003)
- [Rak04] Rakotomamonjy, A.: Optimizing area under roc curve with svms. In: *Proceedings of the First Workshop on ROC Analysis in AI* (2004)
- [XZW06] Xia, F., Zhang, W., Wang, J.: An effective tree-based algorithm for ordinal regression. *IEEE Intelligent Informatics Bulletin* 7(1), 22–26 (2006)
- [YDMW03] Yan, L., Dodier, R.H., Mozer, M., Wolniewicz, R.H.: Optimizing classifier performance via an approximation to the wilcoxon-mann-whitney statistic. In: Fawcett, T., Mishra, N. (eds.) *Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003)*, pp. 848–855 (2003)

Appendix – Proofs

Due to space limitations, we only provide some of the proofs of our results. More details can be found in the extended version of the paper [CV08].

Proof of Proposition 2. The first part of the proposition is a simple consequence of Neyman-Pearson’s lemma formulated in the following setting: given the observation X , test the null assumption $H_0 : Y = -1$ against the alternative $H_1 : Y = +1$. Denote by $p = \mathbb{P}\{Y = 1\}$. The optimal test statistic is then given by the likelihood ratio test:

$$\phi^*(x) = \frac{\mathbb{P}\{X = x \mid Y = 1\}}{\mathbb{P}\{X = x \mid Y = -1\}} = \frac{1-p}{p} \cdot \frac{\eta(x)}{1-\eta(x)}.$$

Indeed, since the function $u \mapsto \frac{1-p}{p} \cdot \frac{u}{1-u}$ is strictly increasing on $(0, 1)$, the test based on the statistic $\phi^*(X)$ is obviously equivalent to the one based $\eta(X)$. Hence η is an optimal scoring function in the sense of the ROC curve. Any element of the class $\mathcal{S}^*$ will also maximize the ROC curve thanks to the invariance property under strictly increasing transforms. The last statement of Proposition 2 is proved as follows. First, we use the fact that, for any measurable function h , we have:

$$\mathbb{E}(h(X) \mid Y = +1) = \frac{1-p}{p} \mathbb{E}\left(\frac{\eta(X)}{1-\eta(X)} h(X) \mid Y = -1\right).$$

We apply this with $h(X) = \mathbb{I}\{X \in R_\alpha^*\} - \mathbb{I}\{X \in R_{s,\alpha}\}$ to get:

$$\text{ROC}^*(\alpha) - \text{ROC}(s, \alpha) = \frac{1-p}{p} \mathbb{E}\left(\frac{\eta(X)}{1-\eta(X)} h(X) \mid Y = -1\right).$$

Then we add and subtract $\frac{Q^*(\alpha)}{1-Q^*(\alpha)}$ and using the fact that $1 - \alpha = \mathbb{P}\{X \in R_{s,\alpha}\} = \mathbb{P}\{X \in R_\alpha^*\}$, we get:

$$\text{ROC}^*(\alpha) - \text{ROC}(s, \alpha) = \left(\frac{1-p}{p}\right) \mathbb{E}\left(\left(\frac{\eta(X)}{1-\eta(X)} - \frac{Q^*(\alpha)}{1-Q^*(\alpha)}\right) h(X) \mid Y = -1\right).$$

We remove the conditioning with respect to $Y = -1$ and using then conditioning on X , we obtain:

$$\text{ROC}^*(\alpha) - \text{ROC}(s, \alpha) = \frac{1}{p} \mathbb{E} \left(\left(\frac{\eta(X) - Q^*(\alpha)}{1 - Q^*(\alpha)} \right) \mathbf{1}_{h(X)} \right).$$

It is then easy to see that this expression corresponds to the statement in the Proposition. $\blacksquare$

Proof of Proposition 4. We recall (see [CLV08]) that:

$$\text{AUC}^* - \text{AUC}(\hat{\eta}) = \frac{\mathbb{E}(|\eta(X) - \eta(X')| \mathbb{I}\{(X, X') \in \Gamma\})}{2p(1-p)}.$$

where $\Gamma = \{(x, x') : \text{sgn}(\hat{\eta}(X) - \hat{\eta}(X')) \neq \text{sgn}(\eta(X) - \eta(X'))\}$. But, one may easily check that: if $\text{sgn}(\hat{\eta}(X) - \hat{\eta}(X')) \neq \text{sgn}(\eta(X) - \eta(X'))$, then

$$|\eta(X) - \eta(X')| \leq |\eta(X) - \hat{\eta}(X)| + |\eta(X') - \hat{\eta}(X')|,$$

which gives the first part of the result.

Turning to the second assertion, consider the event $\mathcal{E} = \{X \in R_{\alpha}^* \Delta R_{\hat{\eta}, \alpha}\}$. Notice first that, after Proposition 2, we have:

$$\text{ROC}^*(\alpha) - \text{ROC}(\hat{\eta}, \alpha) = \frac{\mathbb{E}(|\eta(X) - Q^*(\alpha)| \mathbb{I}_{\mathcal{E}})}{p(1 - Q^*(\alpha))} \leq \frac{c \mathbb{E}(|H^*(\eta(X)) - 1 + \alpha| \mathbb{I}_{\mathcal{E}})}{p(1 - Q^*(\alpha))}$$

by virtue of the finite increments theorem. Now, observing that

$$\mathcal{E} = \{\text{sgn}(H^*(\eta(X)) - 1 + \alpha) \neq \text{sgn}(H_{\hat{\eta}}(\hat{\eta}(X)) - 1 + \alpha)\},$$

we have in a similar fashion as above: if $X \in R_{\alpha}^* \Delta R_{\hat{\eta}, \alpha}$, then

$$|H^*(\eta(X)) - 1 + \alpha| \leq |H^*(\eta(X)) - H_{\hat{\eta}}(\hat{\eta}(X))|,$$

which, combined to the previous bound, proves the second part. $\blacksquare$

Partial proof of Theorem 2. We consider here the case of the L_1 . The proof for the L_{∞} -metric is similar.

The general version can be established by recurrence. Here we only detail the transition from $D = 1$ to $D = 2$. Let us introduce the notation: for all $C \in \mathcal{C}$, $d \in \mathbb{N}$ and $k \in \{1, \dots, 2^d\}$,

$$\begin{aligned} A_{d,k}^*(C) &= \alpha(C_{d,k-1}^*)\beta(C) - \beta(C_{d,k-1}^*)\alpha(C), \\ \tilde{A}_{d,k}(C) &= \alpha(C_{d,k-1})\beta(C) - \beta(C_{d,k-1})\alpha(C). \end{aligned}$$

Equipped with this notation, we can bound the deviation $2|\text{AUC}(s_D^*) - \text{AUC}(s_D)|$ by

$$\sum_{k=0}^{2^{D-1}-1} |A_{D-1,k+1}^*(C_{D,2k}^*) - \tilde{A}_{D-1,k+1}(C_{D,2k})|.$$

We have

$$\begin{aligned} 2\text{AUC}(s_1^*) - 1 &= \Lambda_{0,1}^*(C_{1,0}^*) = \beta_{1,1}^* - \alpha_{1,1}^*, \\ 2\text{AUC}(s_1) - 1 &= \Lambda_{0,1}^*(C_{1,0}) = \tilde{\Lambda}_{0,1}(C_{1,0}). \end{aligned}$$

In the first place, notice that

$$\text{AUC}(s_1^*) - \text{AUC}(s_1) \geq 0.$$

Indeed, since ROC^* dominates any true ROC curve everywhere, observe that

$$\beta(C_{1,0}) - \alpha(C_{1,0}) \leq \text{ROC}^*(\alpha(C_{1,0})) - \alpha(C_{1,0}).$$

and recall that $\alpha_{1,1}^* = \arg \max_{\alpha \in (0,1)} \{\text{ROC}^*(\alpha) - \alpha\}$.

Now, write

$$2\{\text{AUC}(s_1^*) - \text{AUC}(s_1)\} = (\text{I}) + (\text{II}) + (\text{III}),$$

where

$$\begin{aligned} (\text{I}) &= \Lambda_{0,1}^*(C_{1,0}^*) - \Lambda_{0,1}(C_{1,0}^*) \\ (\text{II}) &= \Lambda_{0,1}(C_{1,0}^*) - \Lambda_{0,1}(C_{1,0}) \\ (\text{III}) &= \Lambda_{0,1}(C_{1,0}) - \Lambda_{0,1}^*(C_{1,0}). \end{aligned}$$

By definition, one has $(\text{II}) \leq 0$, while (I) and (III) are both bounded by

$$\sup_{C \in \mathcal{C}} |\alpha(C) - \hat{\alpha}(C)| + \sup_{C \in \mathcal{C}} |\beta(C) - \hat{\beta}(C)|.$$

Let $\delta > 0$. Consequently, using twice the VC inequality for the expectation of a supremum, we obtain that, with probability at least $1 - \delta$: $\forall n \in \mathbb{N}$,

$$\text{AUC}(s_1^*) - \text{AUC}(s_1) \leq c_1 \sqrt{\frac{V}{n}} + c_2 \sqrt{\frac{\log(1/\delta)}{n}} = B(1, n, \delta),$$

Using a Taylor-Lagrange expansion of $\alpha \mapsto \text{ROC}^*(\alpha) - \alpha$ around $\alpha_{1,1}^*$ at the second order, we get that $\{\text{ROC}^*(\alpha_{1,1}^*) - \alpha_{1,1}^*\} - \{\text{ROC}^*(\alpha(C_{1,0})) - \alpha(C_{1,0})\}$ is equal to

$$-\frac{1}{2}(\text{ROC}^*)''(\tilde{\alpha})(\alpha_{1,1}^* - \alpha(C_{1,0}))^2,$$

for a certain $\tilde{\alpha}$ between $\alpha(C_{1,0})$ and $\alpha_{1,1}^*$. Besides, using again that ROC^* dominates any other true ROC curve (so that $\beta(C_{1,0}) \leq \text{ROC}^*(\alpha(C_{1,0}))$), it is also bounded by the deviation $2\{\text{AUC}(s_1^*) - \text{AUC}(s_1)\}$. We set $m = -\sup_{\alpha \in [0,1]} (\text{ROC}^*)'(\alpha) > 0$. Combined with the bound previously established, we obtain that, for all $\delta > 0$, we have with probability larger than $1 - \delta$: $\forall n \in \mathbb{N}$,

$$|\alpha_{1,1}^* - \alpha(C_{1,0})| \leq \frac{2}{\sqrt{m}} \sqrt{B(1, n, \delta)} \leq \frac{2}{\sqrt{m}} B(2, n, \delta).$$

By triangular inequality again, we have with probability larger than δ : $\forall n \in \mathbb{N}$,

$$|\beta_{1,1}^* - \beta(C_{1,0})| \leq \frac{2}{\sqrt{m}}B(2, n, \delta) + B(1, n, \delta).$$

Hence, $\forall n \geq n_\delta = \max\{c_1^2 V, c_2^2 \log(1/\delta)\}$, with probability at least $1 - \delta$, we have

$$|\alpha(C_{1,1}^*) - \alpha(C_{1,0})| + |\beta(C_{1,1}^*) - \beta(C_{1,0})| \leq \kappa_2 B(2, n, \delta),$$

with $\kappa_2 = 6/\sqrt{m}$. This suggests that the deviation at the next iteration should be of order $O_{\mathbb{P}}(n^{-1/4})$. With some additional technicalities, this argument can be iterated. The recurrence leads to the next lemma which is the key for the proof of the theorem.

Lemma 1. *Under the assumptions of Theorem 2, there exist constants κ_1 , κ_2 , c_1 and c_2 such that, for all $\delta > 0$, we have with probability at least $1 - \delta$: $\forall d \in \mathbb{N}$, $\forall n \in \mathbb{N}$,*

$$|\text{AUC}(s_d^*) - \text{AUC}(s_d)| \leq \kappa_1^{d-1} B(d, n, \delta),$$

and $\forall k \in \{0, \dots, 2^{d-1} - 1\}$,

$$|\alpha(C_{d,2k}^*) - \alpha(C_{d,2k})| + |\beta(C_{d,2k}^*) - \beta(C_{d,2k})| \leq \kappa_2^d B(d+1, n, \delta),$$

where: $\forall (d, n, \delta) \in \mathbb{N} \times \mathbb{N} \times]0, 1[$,

$$B(d, n, \delta) = \left(\frac{c_1^2 V}{n} \right)^{\frac{1}{2d}} + \left(\frac{c_2^2 \log(1/\delta)}{n} \right)^{\frac{1}{2d}}.$$

Sample Selection Bias Correction Theory

Corinna Cortes¹, Mehryar Mohri^{1,2}, Michael Riley¹, and Afshin Rostamizadeh²

¹ Google Research,

76 Ninth Avenue, New York, NY 10011

² Courant Institute of Mathematical Sciences,

251 Mercer Street, New York, NY 10012

Abstract. This paper presents a theoretical analysis of sample selection bias correction. The sample bias correction technique commonly used in machine learning consists of reweighting the cost of an error on each training point of a biased sample to more closely reflect the unbiased distribution. This relies on weights derived by various estimation techniques based on finite samples. We analyze the effect of an error in that estimation on the accuracy of the hypothesis returned by the learning algorithm for two estimation techniques: a cluster-based estimation technique and kernel mean matching. We also report the results of sample bias correction experiments with several data sets using these techniques. Our analysis is based on the novel concept of *distributional stability* which generalizes the existing concept of point-based stability. Much of our work and proof techniques can be used to analyze other importance weighting techniques and their effect on accuracy when using a distributionally stable algorithm.

1 Introduction

In the standard formulation of machine learning problems, the learning algorithm receives training and test samples drawn according to the same distribution. However, this assumption often does not hold in practice. The training sample available is *biased* in some way, which may be due to a variety of practical reasons such as the cost of data labeling or acquisition. The problem occurs in many areas such as astronomy, econometrics, and species habitat modeling.

In a common instance of this problem, points are drawn according to the test distribution but not all of them are made available to the learner. This is called the *sample selection bias problem*. Remarkably, it is often possible to correct this bias by using large amounts of unlabeled data.

The problem of sample selection bias correction for linear regression has been extensively studied in econometrics and statistics (Heckman, 1979; Little & Rubin, 1986) with the pioneering work of Heckman (1979). Several recent machine learning publications (Elkan, 2001; Zadrozny, 2004; Zadrozny et al., 2003; Fan et al., 2005; Dudík et al., 2006) have also dealt with this problem. The main correction technique used in all of these publications consists of reweighting the cost of training point errors to more closely reflect that of the test distribution. This is in fact a technique commonly used in statistics and machine learning for

a variety of problems of this type (Little & Rubin, 1986). With the exact weights, this reweighting could optimally correct the bias, but, in practice, the weights are based on an estimate of the sampling probability from finite data sets. Thus, it is important to determine to what extent the error in this estimation can affect the accuracy of the hypothesis returned by the learning algorithm. To our knowledge, this problem has not been analyzed in a general manner.

This paper gives a theoretical analysis of sample selection bias correction. Our analysis is based on the novel concept of *distributional stability* which generalizes the point-based stability introduced and analyzed by previous authors (Devroye & Wagner, 1979; Kearns & Ron, 1997; Bousquet & Elisseeff, 2002). We show that large families of learning algorithms, including all kernel-based regularization algorithms such as Support Vector Regression (SVR) (Vapnik, 1998) or kernel ridge regression (Saunders et al., 1998) are distributionally stable and we give the expression of their stability coefficient for both the l_1 and l_2 distance.

We then analyze two commonly used sample bias correction techniques: a cluster-based estimation technique and kernel mean matching (KMM) (Huang et al., 2006b). For each of these techniques, we derive bounds on the difference of the error rate of the hypothesis returned by a distributionally stable algorithm when using that estimation technique versus using perfect reweighting. We briefly discuss and compare these bounds and also report the results of experiments with both estimation techniques for several publicly available machine learning data sets. Much of our work and proof techniques can be used to analyze other importance weighting techniques and their effect on accuracy when used in combination with a distributionally stable algorithm.

The remaining sections of this paper are organized as follows. Section 2 describes in detail the sample selection bias correction technique. Section 3 introduces the concept of distributional stability and proves the distributional stability of kernel-based regularization algorithms. Section 4 analyzes the effect of estimation error using distributionally stable algorithms for both the cluster-based and the KMM estimation techniques. Section 5 reports the results of experiments with several data sets comparing these estimation techniques.

2 Sample Selection Bias Correction

2.1 Problem

Let X denote the input space and Y the label set, which may be $\{0, 1\}$ in classification or any measurable subset of $\mathbb{R}$ in regression estimation problems, and let $\mathcal{D}$ denote the *true distribution* over $X \times Y$ according to which test points are drawn. In the sample selection bias problem, some pairs $z = (x, y)$ drawn according to $\mathcal{D}$ are not made available to the learning algorithm. The learning algorithm receives a training sample S of m labeled points $z_1, \dots, z_m$ drawn according to a *biased distribution* $\mathcal{D}'$ over $X \times Y$. This sample bias can be represented by a random variable s taking values in $\{0, 1\}$: when $s = 1$ the point is sampled, otherwise it is not. Thus, by definition of the sample selection bias, the support of the biased distribution $\mathcal{D}'$ is included in that of the true distribution $\mathcal{D}$.

As in standard learning scenarios, the objective of the learning algorithm is to select a hypothesis h out of a hypothesis set H with a small generalization error $R(h)$ with respect to the true distribution $\mathcal{D}$, $R(h) = \mathbb{E}_{(x,y) \sim \mathcal{D}}[c(h, z)]$, where $c(h, z)$ is the cost of the error of h on point $z \in X \times Y$.

While the sample S is collected in some biased manner, it is often possible to derive some information about the nature of the bias. This can be done by exploiting large amounts of unlabeled data drawn according to the true distribution $\mathcal{D}$, which is often available in practice. Thus, in the following let U be a sample drawn according to $\mathcal{D}$ and $S \subseteq U$ a labeled but biased sub-sample.

2.2 Weighted Samples

A *weighted sample* S_w is a training sample S of m labeled points, $z_1, \dots, z_m$ drawn i.i.d. from $X \times Y$, that is augmented with a non-negative weight $w_i \geq 0$ for each point z_i . This weight is used to emphasize or de-emphasize the cost of an error on z_i as in the so-called *importance weighting* or *cost-sensitive learning* (Elkan, 2001; Zadrozny et al., 2003). One could use the weights w_i to derive an equivalent but larger unweighted sample S' where the multiplicity of z_i would reflect its weight w_i , but most learning algorithms, e.g., decision trees, logistic regression, AdaBoost, Support Vector Machines (SVMs), kernel ridge regression, can directly accept a weighted sample S_w . We will refer to algorithms that can directly take S_w as input as *weight-sensitive algorithms*.

The empirical error of a hypothesis h on a weighted sample S_w is defined as

$$\hat{R}_w(h) = \sum_{i=1}^m w_i c(h, z_i). \quad (1)$$

Proposition 1. *Let $\mathcal{D}'$ be a distribution whose support coincides with that of $\mathcal{D}$ and let S_w be a weighted sample with $w_i = \Pr_{\mathcal{D}}(z_i) / \Pr_{\mathcal{D}'}(z_i)$ for all points z_i in S . Then,*

$$\mathbb{E}_{S \sim \mathcal{D}'}[\hat{R}_w(h)] = R(h) = \mathbb{E}_{z \sim \mathcal{D}}[c(h, z)]. \quad (2)$$

Proof. Since the sample points are drawn i.i.d.,

$$\mathbb{E}_{S \sim \mathcal{D}'}[\hat{R}_w(h)] = \frac{1}{m} \sum_z \mathbb{E}_{S \sim \mathcal{D}'}[w_i c(h, z_i)] = \mathbb{E}_{z_1 \sim \mathcal{D}'}[w_1 c(h, z_1)]. \quad (3)$$

By definition of w and the fact that the support of $\mathcal{D}$ and $\mathcal{D}'$ coincide, the right-hand side can be rewritten as follows

$$\sum_{\mathcal{D}'(z_1) \neq 0} \frac{\Pr_{\mathcal{D}}(z_1)}{\Pr_{\mathcal{D}'}(z_1)} \Pr_{\mathcal{D}'}(z_1) c(h, z_1) = \sum_{\mathcal{D}(z_1) \neq 0} \Pr_{\mathcal{D}}(z_1) c(h, z_1) = \mathbb{E}_{z_1 \sim \mathcal{D}}[c(h, z_1)]. \quad (4)$$

This last term is the definition of the generalization error $R(h)$. $\square$

2.3 Bias Correction

The probability of drawing $z = (x, y)$ according to the true but unobserved distribution $\mathcal{D}$ can be straightforwardly related to the observed distribution $\mathcal{D}'$. By definition of the random variable s , the observed biased distribution $\mathcal{D}'$ can be expressed by $\Pr_{\mathcal{D}'}[z] = \Pr_{\mathcal{D}}[z|s = 1]$. We will assume that all points z in the support of $\mathcal{D}$ can be sampled with a non-zero probability so the support of $\mathcal{D}$ and $\mathcal{D}'$ coincide. Thus for all $z \in X \times Y$, $\Pr[s = 1|z] \neq 0$. Then, by the Bayes formula, for all z in the support of $\mathcal{D}$,

$$\Pr_{\mathcal{D}}[z] = \frac{\Pr[z|s = 1] \Pr[s = 1]}{\Pr[s = 1|z]} = \frac{\Pr[s = 1]}{\Pr[s = 1|z]} \Pr_{\mathcal{D}'}[z]. \quad (5)$$

Thus, if we were given the probabilities $\Pr[s = 1]$ and $\Pr[s = 1|z]$, we could derive the true probability $\Pr_{\mathcal{D}}$ from the biased one $\Pr_{\mathcal{D}'}$ exactly and correct the sample selection bias.

It is important to note that this correction is only needed for the training sample S , since it is the only source of selection bias. With a weight-sensitive algorithm, it suffices to reweight each sample z_i with the weight $w_i = \frac{\Pr[s=1]}{\Pr[s=1|z_i]}$. Thus, $\Pr[s = 1|z]$ need not be estimated for all points z but only for those falling in the training sample S . By Proposition 1, the expected value of the empirical error after reweighting is the same as if we were given samples from the true distribution and the usual generalization bounds hold for $\widehat{R}(h)$ and $R(h)$.

When the sampling probability is independent of the labels, as it is commonly assumed in many settings (Zadrozny 2004; 2003), $\Pr[s = 1|z] = \Pr[s = 1|x]$, and Equation 5 can be re-written as

$$\Pr_{\mathcal{D}}[z] = \frac{\Pr[s = 1]}{\Pr[s = 1|x]} \Pr_{\mathcal{D}'}[z]. \quad (6)$$

In that case, the probabilities $\Pr[s = 1]$ and $\Pr[s = 1|x]$ needed to reconstitute $\Pr_{\mathcal{D}}$ from $\Pr_{\mathcal{D}'}$ do not depend on the labels and thus can be estimated using the unlabeled points in U . Moreover, as already mentioned, for weight-sensitive algorithms, it suffices to estimate $\Pr[s = 1|x_i]$ for the points x_i of the training data; no generalization is needed.

A simple case is when the points are defined over a discrete set.¹ $\Pr[s = 1|x]$ can then be estimated from the frequency m_x/n_x , where m_x denotes the number of times x appeared in $S \subseteq U$ and n_x the number of times x appeared in the full data set U . $\Pr[s = 1]$ can be estimated by the quantity $|S|/|U|$. However, since $\Pr[s = 1]$ is a constant independent of x , its estimation is not even necessary.

If the estimation of the sampling probability $\Pr[s = 1|x]$ from the unlabeled data set U were exact, then the reweighting just discussed could correct the sample bias optimally. Several techniques have been commonly used to estimate the reweighting quantities. But, these estimate weights are not guaranteed to be exact. The next section addresses how the error in that estimation affects the error rate of the hypothesis returned by the learning algorithm.

¹ This can be as a result of a quantization or clustering technique as discussed later.

3 Distributional Stability

Here, we will examine the effect on the error of the hypothesis returned by the learning algorithm in response to a change in the way the training points are weighted. Since the weights are non-negative, we can assume that they are normalized and define a distribution over the training sample. This study can be viewed as a generalization of stability analysis where a single sample point is changed (Devroye & Wagner, 1979; Kearns & Ron, 1997; Bousquet & Elisseeff, 2002) to the more general case of *distributional stability* where the sample's weight distribution is changed.

Thus, in this section the sample weight $\mathcal{W}$ of $S_{\mathcal{W}}$ defines a distribution over S . For a fixed learning algorithm L and a fixed sample S , we will denote by $h_{\mathcal{W}}$ the hypothesis returned by L for the weighted sample $S_{\mathcal{W}}$. We will denote by $d(\mathcal{W}, \mathcal{W}')$ a divergence measure for two distributions $\mathcal{W}$ and $\mathcal{W}'$. There are many standard measures for the divergences or distances between two distributions, including the relative entropy, the Hellinger distance, and the l_p distance.

Definition 1 (Distributional β -Stability). *A learning algorithm L is said to be distributionally β -stable for the divergence measure d if for any two weighted samples $S_{\mathcal{W}}$ and $S_{\mathcal{W}'}$,*

$$\forall z \in X \times Y, \quad |c(h_{\mathcal{W}}, z) - c(h_{\mathcal{W}'}, z)| \leq \beta d(\mathcal{W}, \mathcal{W}'). \quad (7)$$

Thus, an algorithm is distributionally stable when small changes to a weighted sample's distribution, as measured by a divergence d , result in a small change in the cost of an error at any point. The following proposition follows directly from the definition of distributional stability.

Proposition 2. *Let L be a distributionally β -stable algorithm and let $h_{\mathcal{W}}$ ($h_{\mathcal{W}'}$) denote the hypothesis returned by L when trained on the weighted sample $S_{\mathcal{W}}$ (resp. $S_{\mathcal{W}'}$). Let $\mathcal{W}_T$ denote the distribution according to which test points are drawn. Then, the following holds*

$$|R(h_{\mathcal{W}}) - R(h_{\mathcal{W}'})| \leq \beta d(\mathcal{W}, \mathcal{W}'). \quad (8)$$

Proof. By the distributional stability of the algorithm,

$$\mathbb{E}_{z \sim \mathcal{W}_T} [|c(z, h_{\mathcal{W}}) - c(z, h_{\mathcal{W}'})|] \leq \beta d(\mathcal{W}, \mathcal{W}'), \quad (9)$$

which implies the statement of the proposition. $\square$

3.1 Distributional Stability of Kernel-Based Regularization Algorithms

Here, we show that kernel-based regularization algorithms are distributionally β -stable. This family of algorithms includes, among others, Support Vector Regression (SVR) and kernel ridge regression. Other algorithms such as those based on

the relative entropy regularization can be shown to be distributionally β -stable in a similar way as for point-based stability. Our results also apply to classification algorithms such as Support Vector Machine (SVM) (Cortes & Vapnik, 1995) using a margin-based loss function l_γ as in (Bousquet & Elisseeff, 2002).

We will assume that the cost function c is σ -admissible, that is there exists $\sigma \in \mathbb{R}_+$ such that for any two hypotheses $h, h' \in H$ and for all $z = (x, y) \in X \times Y$,

$$|c(h, z) - c(h', z)| \leq \sigma |h(x) - h'(x)|. \quad (10)$$

This assumption holds for the quadratic cost and most other cost functions when the hypothesis set and the set of output labels are bounded by some $M \in \mathbb{R}_+$: $\forall h \in H, \forall x \in X, |h(x)| \leq M$ and $\forall y \in Y, |y| \leq M$. We will also assume that c is differentiable. This assumption is in fact not necessary and all of our results hold without it, but it makes the presentation simpler.

Let $N: H \rightarrow \mathbb{R}_+$ be a function defined over the hypothesis set. Regularization-based algorithms minimize an objective of the form: $F_{\mathcal{W}}(h) = \widehat{R}_{\mathcal{W}}(h) + \lambda N(h)$, where $\lambda \geq 0$ is a trade-off parameter. We denote by B_F the Bregman divergence associated to a convex function F , $B_F(f||g) = F(f) - F(g) - \langle f - g, \nabla F(g) \rangle$, and define Δh as $\Delta h = h' - h$.

Lemma 1. *Let the hypothesis set H be a vector space. Assume that N is a proper closed convex function and that N is differentiable. Assume that $F_{\mathcal{W}}$ admits a minimizer $h \in H$ and $F_{\mathcal{W}'}$ a minimizer $h' \in H$. Then, the following bound holds,*

$$B_N(h'||h) + B_N(h||h') \leq \frac{\sigma l_1(\mathcal{W}, \mathcal{W}')}{\lambda} \sup_{x \in S} |\Delta h(x)|. \quad (11)$$

Proof. Since $B_{F_{\mathcal{W}}} = B_{\widehat{R}_{\mathcal{W}}} + \lambda B_N$ and $B_{F_{\mathcal{W}'}} = B_{\widehat{R}_{\mathcal{W}'}} + \lambda B_N$, and a Bregman divergence is non-negative, $\lambda(B_N(h'||h) + B_N(h||h')) \leq B_{F_{\mathcal{W}}}(h'||h) + B_{F_{\mathcal{W}'}}(h||h')$. By the definition of h and h' as the minimizers of $F_{\mathcal{W}}$ and $F_{\mathcal{W}'}$,

$$B_{F_{\mathcal{W}}}(h'||h) + B_{F_{\mathcal{W}'}}(h||h') = \widehat{R}_{F_{\mathcal{W}}}(h') - \widehat{R}_{F_{\mathcal{W}}}(h) + \widehat{R}_{F_{\mathcal{W}'}}(h) - \widehat{R}_{F_{\mathcal{W}'}}(h'). \quad (12)$$

Thus, by the σ -admissibility of the cost function c , using the notation $\mathcal{W}_i = \mathcal{W}(x_i)$ and $\mathcal{W}'_i = \mathcal{W}'(x_i)$,

$$\begin{aligned} \lambda(B_N(h'||h) + B_N(h||h')) &\leq \widehat{R}_{F_{\mathcal{W}}}(h') - \widehat{R}_{F_{\mathcal{W}}}(h) + \widehat{R}_{F_{\mathcal{W}'}}(h) - \widehat{R}_{F_{\mathcal{W}'}}(h') \\ &= \sum_{i=1}^m \left[c(h', z_i) \mathcal{W}_i - c(h, z_i) \mathcal{W}_i + c(h, z_i) \mathcal{W}'_i - c(h', z_i) \mathcal{W}'_i \right] \\ &= \sum_{i=1}^m \left[(c(h', z_i) - c(h, z_i)) (\mathcal{W}_i - \mathcal{W}'_i) \right] = \sum_{i=1}^m \left[\sigma |\Delta h(x_i)| (\mathcal{W}_i - \mathcal{W}'_i) \right] \\ &\leq \sigma l_1(\mathcal{W}, \mathcal{W}') \sup_{x \in S} |\Delta h(x)|, \end{aligned} \quad (13)$$

which establishes the lemma. $\square$

Given $x_1, \dots, x_m \in X$ and a positive definite symmetric (PDS) kernel K , we denote by $\mathbf{K} \in \mathbb{R}^{m \times m}$ the kernel matrix defined by $\mathbf{K}_{ij} = K(x_i, x_j)$ and by $\lambda_{\max}(\mathbf{K}) \in \mathbb{R}_+$ the largest eigenvalue of $\mathbf{K}$.

Lemma 2. *Let H be a reproducing kernel Hilbert space with kernel K and let the regularization function N be defined by $N(\cdot) = \|\cdot\|_K^2$. Then, the following bound holds,*

$$B_N(h'\|h) + B_N(h\|h') \leq \frac{\sigma \lambda_{\max}^{\frac{1}{2}}(\mathbf{K}) l_2(\mathcal{W}, \mathcal{W}')}{\lambda} \|\Delta h\|_2. \quad (14)$$

Proof. As in the proof of Lemma 1,

$$\lambda(B_N(h'\|h) + B_N(h\|h')) \leq \sum_{i=1}^m \left[(c(h', z_i) - c(h, z_i))(\mathcal{W}_i - \mathcal{W}'_i) \right]. \quad (15)$$

By definition of a reproducing kernel Hilbert space H , for any hypothesis $h \in H$, $\forall x \in X, h(x) = \langle h, K(x, \cdot) \rangle$ and thus also for any $\Delta h = h' - h$ with $h, h' \in H$, $\forall x \in X, \Delta h(x) = \langle \Delta h, K(x, \cdot) \rangle$. Let $\Delta \mathcal{W}_i$ denote $\mathcal{W}'_i - \mathcal{W}_i$, $\Delta \mathcal{W}$ the vector whose components are the $\Delta \mathcal{W}_i$'s, and let V denote $B_N(h'\|h) + B_N(h\|h')$. Using σ -admissibility, $V \leq \sigma \sum_{i=1}^m |\Delta h(x_i) \Delta \mathcal{W}_i| = \sigma \sum_{i=1}^m |\langle \Delta h, \Delta \mathcal{W}_i K(x_i, \cdot) \rangle|$. Let $\epsilon_i \in \{-1, +1\}$ denote the sign of $\langle \Delta h, \Delta \mathcal{W}_i K(x_i, \cdot) \rangle$. Then,

$$\begin{aligned} V &\leq \sigma \left\langle \Delta h, \sum_{i=1}^m \epsilon_i \Delta \mathcal{W}_i K(x_i, \cdot) \right\rangle \leq \sigma \|\Delta h\|_K \left\| \sum_{i=1}^m \epsilon_i \Delta \mathcal{W}_i K(x_i, \cdot) \right\|_K \\ &= \sigma \|\Delta h\|_K \left(\sum_{i,j=1}^m \epsilon_i \epsilon_j \Delta \mathcal{W}_i \Delta \mathcal{W}_j K(x_i, x_j) \right)^{1/2} \\ &= \sigma \|\Delta h\|_K [\Delta(\mathcal{W} \epsilon)^\top \mathbf{K} \Delta(\mathcal{W} \epsilon)]^{\frac{1}{2}} \leq \sigma \|\Delta h\|_K \|\Delta \mathcal{W}\|_2 \lambda_{\max}^{\frac{1}{2}}(\mathbf{K}). \end{aligned} \quad (16)$$

In this derivation, the second inequality follows from the Cauchy-Schwarz inequality and the last inequality from the standard property of the Rayleigh quotient for PDS matrices. Since $\|\Delta \mathcal{W}\|_2 = l_2(\mathcal{W}, \mathcal{W}')$, this proves the lemma. $\square$

Theorem 1. *Let K be a kernel such that $K(x, x) \leq \kappa < \infty$ for all $x \in X$. Then, the regularization algorithm based on $N(\cdot) = \|\cdot\|_K^2$ is distributionally β -stable for the l_1 distance with $\beta \leq \frac{\sigma^2 \kappa^2}{2\lambda}$, and for the l_2 distance with $\beta \leq \frac{\sigma^2 \kappa \lambda_{\max}^{\frac{1}{2}}(\mathbf{K})}{2\lambda}$.*

Proof. For $N(\cdot) = \|\cdot\|_K^2$, we have $B_N(h'\|h) = \|h' - h\|_K^2$, thus $B_N(h'\|h) + B_N(h\|h') = 2\|\Delta h\|_K^2$ and by Lemma 1,

$$2\|\Delta h\|_K^2 \leq \frac{\sigma l_1(\mathcal{W}, \mathcal{W}')}{\lambda} \sup_{x \in S} |\Delta h(x)| \leq \frac{\sigma l_1(\mathcal{W}, \mathcal{W}')}{\lambda} \kappa \|\Delta h\|_K. \quad (17)$$

Thus $\|\Delta h\|_K \leq \frac{\sigma \kappa l_1(\mathcal{W}, \mathcal{W}')}{2\lambda}$. By σ -admissibility of c ,

$$\forall z \in X \times Y, |c(h', z) - c(h, z)| \leq \sigma |\Delta h(x)| \leq \kappa \sigma \|\Delta h\|_K. \quad (18)$$

Therefore,

$$\forall z \in X \times Y, |c(h', z) - c(h, z)| \leq \frac{\sigma^2 \kappa^2 l_1(\mathcal{W}, \mathcal{W}')}{2\lambda}, \quad (19)$$

which shows the distributional stability of a kernel-based regularization algorithm for the l_1 distance. Using Lemma 2, a similar derivation leads to

$$\forall z \in X \times Y, |c(h', z) - c(h, z)| \leq \frac{\sigma^2 \kappa \lambda_{\max}^{\frac{1}{2}}(\mathbf{K}) l_2(\mathcal{W}, \mathcal{W}')}{2\lambda}, \quad (20)$$

which shows the distributional stability of a kernel-based regularization algorithm for the l_2 distance. $\square$

Note that the standard setting of a sample with no weight is equivalent to a weighted sample with the uniform distribution $\mathcal{W}_{\mathcal{U}}$: each point is assigned the weight $1/m$. Removing a single point, say x_1 , is equivalent to assigning weight 0 to x_1 and $1/(m-1)$ to others. Let $\mathcal{W}_{\mathcal{U}'}$ be the corresponding distribution, then $l_1(\mathcal{W}_{\mathcal{U}}, \mathcal{W}_{\mathcal{U}'}) = \frac{1}{m} + \sum_{i=1}^{m-1} \left| \frac{1}{m} - \frac{1}{m-1} \right| = \frac{2}{m}$. Thus, in the case of kernel-based regularized algorithms and for the l_1 distance, standard uniform β -stability is a special case of distributional β -stability. It can be shown similarly that $l_2(\mathcal{W}_{\mathcal{U}}, \mathcal{W}_{\mathcal{U}'}) = \frac{1}{\sqrt{m(m-1)}}$.

4 Effect of Estimation Error for Kernel-Based Regularization Algorithms

This section analyzes the effect of an error in the estimation of the weight of a training example on the generalization error of the hypothesis h returned by a weight-sensitive learning algorithm. We will examine two estimation techniques: a straightforward histogram-based or cluster-based method, and kernel mean matching (KMM) (Huang et al., 2006b).

4.1 Cluster-Based Estimation

A straightforward estimate of the probability of sampling is based on the observed empirical frequencies. The ratio of the number of times a point x appears in S and the number of times it appears in U is an empirical estimate of $\Pr[s=1|x]$. Note that generalization to unseen points x is not needed since reweighting requires only assigning weights to the seen training points. However, in general, training instances are typically unique or very infrequent since features are real-valued numbers. Instead, features can be discretized based on a partitioning of the input space X . The partitioning may be based on a simple histogram buckets or the result of a clustering technique. The analysis of this section assumes such a prior partitioning of X .

We shall analyze how fast the resulting empirical frequencies converge to the true sampling probability. For $x \in U$, let U_x denote the subsample of U containing exactly all the instances of x and let $n = |U|$ and $n_x = |U_x|$. Furthermore,

let n' denote the number of unique points in the sample U . Similarly, we define S_x , m , m_x and m' for the set S . Additionally, denote by $p_0 = \min_{x \in U} \Pr[x] \neq 0$.

Lemma 3. *Let $\delta > 0$. Then, with probability at least $1 - \delta$, the following inequality holds for all x in S :*

$$\left| \Pr[s = 1|x] - \frac{m_x}{n_x} \right| \leq \sqrt{\frac{\log 2m' + \log \frac{1}{\delta}}{p_0 n}}. \quad (21)$$

Proof. For a fixed $x \in U$, by Hoeffding's inequality,

$$\begin{aligned} \Pr_U \left[\left| \Pr[s = 1|x] - \frac{m_x}{n_x} \right| \geq \epsilon \right] &= \sum_{i=1}^n \Pr_x \left[\left| \Pr[s = 1|x] - \frac{m_x}{i} \right| \geq \epsilon \mid n_x = i \right] \Pr[n_x = i] \\ &\leq \sum_{i=1}^n 2e^{-2i\epsilon^2} \Pr_U[n_x = i]. \end{aligned}$$

Since n_x is a binomial random variable with parameters $\Pr_U[x] = p_x$ and n , this last term can be expressed more explicitly and bounded as follows:

$$\begin{aligned} 2 \sum_{i=1}^n e^{-2i\epsilon^2} \Pr_U[n_x = i] &\leq 2 \sum_{i=0}^n e^{-2i\epsilon^2} \binom{n}{i} p_x^i (1-p_x)^{n-i} = 2(p_x e^{-2\epsilon^2} + (1-p_x))^n \\ &= 2(1 - p_x(1 - e^{-2\epsilon^2}))^n \leq 2 \exp(-p_x n(1 - e^{-2\epsilon^2})). \end{aligned}$$

Since for $x \in [0, 1]$, $1 - e^{-x} \geq x/2$, this shows that for $\epsilon \in [0, 1]$,

$$\Pr_U \left[\left| \Pr[s = 1|x] - \frac{m_x}{n_x} \right| \geq \epsilon \right] \leq 2e^{-p_x n \epsilon^2}.$$

By the union bound and the definition of p_0 ,

$$\Pr_U \left[\exists x \in S : \left| \Pr[s = 1|x] - \frac{m_x}{n_x} \right| \geq \epsilon \right] \leq 2m' e^{-p_0 n \epsilon^2}.$$

Setting δ to match the upper bound yields the statement of the lemma. $\square$

The following proposition bounds the distance between the distribution $\mathcal{W}$ corresponding to a perfectly reweighted sample $(S_{\mathcal{W}})$ and the one corresponding to a sample that is reweighted according to the observed bias $(S_{\widehat{\mathcal{W}}})$. For a sampled point $x_i = x$, these distributions are defined as follows:

$$\mathcal{W}(x_i) = \frac{1}{m} \frac{1}{p(x_i)} \quad \text{and} \quad \widehat{\mathcal{W}}(x_i) = \frac{1}{m} \frac{1}{\hat{p}(x_i)}, \quad (22)$$

where, for a *distinct* point x equal to the *sampled* point x_i , we define $p(x_i) = \Pr[s = 1|x]$ and $\hat{p}(x_i) = \frac{m_x}{n_x}$.

Proposition 3. *Let $B = \max_{i=1, \dots, m} \max(1/p(x_i), 1/\hat{p}(x_i))$. Then, the l_1 and l_2 distances of the distributions $\mathcal{W}$ and $\widehat{\mathcal{W}}$ can be bounded as follows,*

$$l_1(\mathcal{W}, \widehat{\mathcal{W}}) \leq B^2 \sqrt{\frac{\log 2m' + \log \frac{1}{\delta}}{p_0 n}} \quad \text{and} \quad l_2(\mathcal{W}, \widehat{\mathcal{W}}) \leq B^2 \sqrt{\frac{\log 2m' + \log \frac{1}{\delta}}{p_0 n m}}. \quad (23)$$

Proof. By definition of the l_2 distance,

$$\begin{aligned} l_2^2(\mathcal{W}, \widehat{\mathcal{W}}) &= \frac{1}{m^2} \sum_{i=1}^m \left(\frac{1}{p(x_i)} - \frac{1}{\hat{p}(x_i)} \right)^2 = \frac{1}{m^2} \sum_{i=1}^m \left(\frac{p(x_i) - \hat{p}(x_i)}{p(x_i)\hat{p}(x_i)} \right)^2 \\ &\leq \frac{B^4}{m} \max_i (p(x_i) - \hat{p}(x_i))^2. \end{aligned}$$

It can be shown similarly that $l_1(\mathcal{W}, \widehat{\mathcal{W}}) \leq B^2 \max_i |p(x_i) - \hat{p}(x_i)|$. The application of the uniform convergence bound of Lemma 3 directly yields the statement of the proposition. $\square$

The following theorem provides a bound on the difference between the generalization error of the hypothesis returned by a kernel-based regularization algorithm when trained on the perfectly unbiased distribution, and the one trained on the sample bias-corrected using frequency estimates.

Theorem 2. *Let K be a PDS kernel such that $K(x, x) \leq \kappa < \infty$ for all $x \in X$. Let $h_{\mathcal{W}}$ be the hypothesis returned by the regularization algorithm based on $N(\cdot) = \|\cdot\|_K^2$ using $S_{\mathcal{W}}$, and $h_{\widehat{\mathcal{W}}}$ the one returned after training the same algorithm on $S_{\widehat{\mathcal{W}}}$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the difference in generalization error of these hypotheses is bounded as follows*

$$\begin{aligned} |R(h_{\mathcal{W}}) - R(h_{\widehat{\mathcal{W}}})| &\leq \frac{\sigma^2 \kappa^2 B^2}{2\lambda} \sqrt{\frac{\log 2m' + \log \frac{1}{\delta}}{p_0 n}} \\ |R(h_{\mathcal{W}}) - R(h_{\widehat{\mathcal{W}}})| &\leq \frac{\sigma^2 \kappa \lambda_{\max}^{\frac{1}{2}}(\mathbf{K}) B^2}{2\lambda} \sqrt{\frac{\log 2m' + \log \frac{1}{\delta}}{p_0 n m}}. \end{aligned} \tag{24}$$

Proof. The result follows from Proposition 2, the distributional stability and the bounds on the stability coefficient β for kernel-based regularization algorithms (Theorem 1), and the bounds on the l_1 and l_2 distances between the correct distribution $\mathcal{W}$ and the estimate $\widehat{\mathcal{W}}$. $\square$

Let n_0 be the number of occurrences, in U , of the least frequent training example. For large enough n , $p_0 n \approx n_0$, thus the theorem suggests that the difference of error rate between the hypothesis returned after an optimal reweighting versus the one based on frequency estimates goes to zero as $\sqrt{\frac{\log m'}{n_0}}$. In practice, $m' \leq m$, the number of distinct points in S is small, a fortiori, $\log m'$ is very small, thus, the convergence rate depends essentially on the rate at which n_0 increases. Additionally, if $\lambda_{\max}(K) \leq m$ (such as with Gaussian kernels), the l_2 -based bound will provide convergence that is at least as fast.

4.2 Kernel Mean Matching

The following definitions introduced by Steinwart (2002) will be needed for the presentation and discussion of the kernel mean matching (KMM) technique. Let

X be a compact metric space and let $C(X)$ denote the space of all continuous functions over X equipped with the standard infinite norm $\|\cdot\|_\infty$. Let $K: X \times X \rightarrow \mathbb{R}$ be a PDS kernel. There exists a Hilbert space F and a map $\Phi: X \rightarrow F$ such that for all $x, y \in X$, $K(x, y) = \langle \Phi(x), \Phi(y) \rangle$. Note that for a given kernel K , F and Φ are not unique and that, for these definitions, F does not need to be a reproducing kernel Hilbert space (RKHS).

Let $\mathcal{P}$ denote the set of all probability distributions over X and let $\mu: \mathcal{P} \rightarrow F$ be the function defined by $\forall p \in \mathcal{P}$, $\mu(p) = \mathbb{E}_{x \sim p}[\Phi(x)]$. A function $g: X \rightarrow \mathbb{R}$ is said to be *induced* by K if there exists $w \in F$ such that for all $x \in X$, $g(x) = \langle w, \Phi(x) \rangle$. K is said to be *universal* if it is continuous and if the set of functions induced by K are dense in $C(X)$.

Theorem 3 (Huang et al. (2006a)). *Let F be a separable Hilbert space and let K be a universal kernel with feature space F and feature map $\Phi: X \rightarrow F$. Then, μ is injective.*

Proof. We give a full proof of the main theorem supporting this technique in a longer version of this paper. The proof given by Huang et al. (2006a) does not seem to be complete. $\square$

The KMM technique is applicable when the learning algorithm is based on a universal kernel. The theorem shows that for a universal kernel, the expected value of the feature vectors induced uniquely determines the probability distribution. KMM uses this property to reweight training points so that the average value of the feature vectors for the training data matches that of the feature vectors for a set of unlabeled points drawn from the unbiased distribution.

Let γ_i denote the perfect reweighting of the sample point x_i and $\hat{\gamma}_i$ the estimate derived by KMM. Let B' denote the largest possible reweighting coefficient γ and let ϵ be a positive real number. We will assume that ϵ is chosen so that $\epsilon \leq 1/2$. Then, the following is the KMM constraint optimization

$$\min_{\gamma} G(\gamma) = \left\| \frac{1}{m} \sum_{i=1}^m \gamma_i \Phi(x_i) - \frac{1}{n} \sum_{i=1}^n \Phi(x'_i) \right\| \quad \text{s.t. } \gamma_i \in [0, B'] \wedge \left| \frac{1}{m} \sum_{i=1}^m \gamma_i - 1 \right| \leq \epsilon.$$

Let $\hat{\gamma}$ be the solution of this optimization problem, then $\frac{1}{m} \sum_{i=1}^m \hat{\gamma}_i = 1 + \epsilon'$ with $-\epsilon \leq \epsilon' \leq \epsilon$. For $i \in [1, m]$, let $\hat{\gamma}'_i = \hat{\gamma}_i / (1 + \epsilon')$. The normalized weights used in KMM's reweighting of the sample are thus defined by $\hat{\gamma}'_i / m$ with $\frac{1}{m} \sum_{i=1}^m \hat{\gamma}'_i = 1$.

As in the previous section, given $x_1, \dots, x_m \in X$ and a strictly positive definite universal kernel K , we denote by $\mathbf{K} \in \mathbb{R}^{m \times m}$ the kernel matrix defined by $\mathbf{K}_{ij} = K(x_i, x_j)$ and by $\lambda_{\min}(\mathbf{K}) > 0$ the smallest eigenvalue of $\mathbf{K}$. We also denote by $\text{cond}(\mathbf{K})$ the condition number of the matrix $\mathbf{K}$: $\text{cond}(\mathbf{K}) = \lambda_{\max}(\mathbf{K}) / \lambda_{\min}(\mathbf{K})$. When K is universal, it is continuous over the compact $X \times X$ and thus bounded, and there exists $\kappa < \infty$ such that $K(x, x) \leq \kappa$ for all $x \in X$.

Proposition 4. *Let K be a strictly positive definite universal kernel. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the l_2 distance of the distributions $\hat{\gamma}'/m$ and γ/m is bounded as follows:*

$$\frac{1}{m} \|(\hat{\gamma}' - \gamma)\|_2 \leq \frac{2\epsilon B'}{\sqrt{m}} + \frac{2\kappa^{\frac{1}{2}}}{\lambda_{\min}^{\frac{1}{2}}(\mathbf{K})} \sqrt{\frac{B'^2}{m} + \frac{1}{n}} \left(1 + \sqrt{2 \log \frac{2}{\delta}}\right). \quad (25)$$

Proof. Since the optimal reweighting γ verifies the constraints of the optimization, by definition of $\hat{\gamma}$ as a minimizer, $G(\hat{\gamma}) \leq G(\gamma)$. Thus, by the triangle inequality,

$$\left\| \frac{1}{m} \sum_{i=1}^m \hat{\gamma}_i \Phi(x_i) - \frac{1}{m} \sum_{i=1}^m \gamma_i \Phi(x_i) \right\| \leq G(\hat{\gamma}) + G(\gamma) \leq 2G(\gamma). \quad (26)$$

Let L denote the left-hand side of this inequality: $L = \frac{1}{m} \left\| \sum_{i=1}^m (\hat{\gamma}_i - \gamma_i) \Phi(x_i) \right\|$. By definition of the norm in the Hilbert space, $L = \frac{1}{m} \sqrt{(\hat{\gamma} - \gamma)^\top \mathbf{K}(\hat{\gamma} - \gamma)}$. Then, by the standard bounds for the Rayleigh quotient of PDS matrices, $L \geq \frac{1}{m} \lambda_{\min}^{\frac{1}{2}}(\mathbf{K}) \|(\hat{\gamma} - \gamma)\|_2$. This combined with Inequality 26 yields

$$\frac{1}{m} \|(\hat{\gamma} - \gamma)\|_2 \leq \frac{2G(\gamma)}{\lambda_{\min}^{\frac{1}{2}}(\mathbf{K})}. \quad (27)$$

Thus, by the triangle inequality,

$$\begin{aligned} \frac{1}{m} \|(\hat{\gamma}' - \gamma)\|_2 &\leq \frac{1}{m} \|(\hat{\gamma}' - \hat{\gamma})\|_2 + \frac{1}{m} \|(\hat{\gamma} - \gamma)\|_2 \leq \frac{|\epsilon'|/m}{1 + \epsilon'} \|\gamma\|_2 + \frac{2G(\gamma)}{\lambda_{\min}^{\frac{1}{2}}(\mathbf{K})} \\ &\leq \frac{2|\epsilon'|B'\sqrt{m}}{m} + \frac{2G(\gamma)}{\lambda_{\min}^{\frac{1}{2}}(\mathbf{K})} \leq \frac{2\epsilon B'}{\sqrt{m}} + \frac{2G(\gamma)}{\lambda_{\min}^{\frac{1}{2}}(\mathbf{K})}. \end{aligned} \quad (28)$$

It is not difficult to show using McDiarmid's inequality that for any $\delta > 0$, with probability at least $1 - \delta$, the following holds (Lemma 4, (Huang et al., 2006a)):

$$G(\gamma) \leq \kappa^{\frac{1}{2}} \sqrt{\frac{B'^2}{m} + \frac{1}{n}} \left(1 + \sqrt{2 \log \frac{2}{\delta}}\right). \quad (29)$$

This combined with Inequality 28 yields the statement of the proposition. $\square$

The following theorem provides a bound on the difference between the generalization error of the hypothesis returned by a kernel-based regularization algorithm when trained on the true distribution, and the one trained on the sample bias-corrected KMM.

Theorem 4. *Let K be a strictly positive definite symmetric universal kernel. Let h_γ be the hypothesis returned by the regularization algorithm based on $N(\cdot) = \|\cdot\|_K^2$ using $S_{\gamma/m}$ and $h_{\hat{\gamma}'}$ the one returned after training the same algorithm on $S_{\hat{\gamma}'/m}$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the difference in generalization error of these hypotheses is bounded as follows*

$$|R(h_\gamma) - R(h_{\hat{\gamma}'})| \leq \frac{\sigma^2 \kappa \lambda_{\max}^{\frac{1}{2}}(\mathbf{K})}{\lambda} \left(\frac{\epsilon B'}{\sqrt{m}} + \frac{\kappa^{\frac{1}{2}}}{\lambda_{\min}^{\frac{1}{2}}(\mathbf{K})} \sqrt{\frac{B'^2}{m} + \frac{1}{n}} \left(1 + \sqrt{2 \log \frac{2}{\delta}}\right) \right).$$

For $\epsilon = 0$, the bound becomes

$$|R(h_\gamma) - R(h_{\hat{\gamma}'})| \leq \frac{\sigma^2 \kappa^{\frac{3}{2}} \text{cond}^{\frac{1}{2}}(\mathbf{K})}{\lambda} \sqrt{\frac{B'^2}{m} + \frac{1}{n}} \left(1 + \sqrt{2 \log \frac{2}{\delta}}\right). \quad (30)$$

Proof. The result follows from Proposition 2 and the bound of Proposition 4. $\square$

Comparing this bound for $\epsilon = 0$ with the l_2 bound of Theorem 4, we first note that B and B' are essentially related modulo the constant $\Pr[s = 1]$ which is not included in the cluster-based reweighting. Thus, the cluster-based convergence is of the order $O(\lambda_{\max}^{\frac{1}{2}}(\mathbf{K}) B^2 \sqrt{\frac{\log m'}{p_0 n m}})$ and the KMM convergence of the order $O(\text{cond}^{\frac{1}{2}}(\mathbf{K}) \frac{B}{\sqrt{m}})$. Taking the ratio of the former over the latter and noticing $p_0^{-1} \approx O(B)$, we obtain the expression $O\left(\sqrt{\frac{\lambda_{\min}(\mathbf{K}) B \log m'}{n}}\right)$. Thus, for $n > \lambda_{\min}(\mathbf{K}) B \log(m')$ the convergence of the cluster-based bound is more favorable, while for other values the KMM bound converges faster.

5 Experimental Results

In this section, we will compare the performance of the cluster-based reweighting technique and the KMM technique empirically. We will first discuss and analyze the properties of the clustering method and our particular implementation.

The analysis of Section 4.1 deals with discrete points possibly resulting from the use of a quantization or clustering technique. However, due to the relatively small size of the public training sets available, clustering could leave us with few cluster representatives to train with. Instead, in our experiments, we only used the clusters to estimate sampling probabilities and applied these weights to the full set of training points. As the following proposition shows, the l_1 and l_2 distance bounds of Proposition 5 do not change significantly so long as the cluster size is roughly uniform and the sampling probability is the same for all points within a cluster. We will refer to this as the *clustering assumption*. In what follows, let $\Pr[s = 1 | C_i]$ designate the sampling probability for all $x \in C_i$. Finally, define $q(C_i) = \Pr[s = 1 | C_i]$ and $\hat{q}(C_i) = |C_i \cap S| / |C_i \cap U|$.

Proposition 5. Let $B = \max_{i=1, \dots, m} \max(1/q(C_i), 1/\hat{q}(C_i))$. Then, the l_1 and l_2 distances of the distributions $\mathcal{W}$ and $\widehat{\mathcal{W}}$ can be bounded as follows,

$$l_1(\mathcal{W}, \widehat{\mathcal{W}}) \leq B^2 \sqrt{\frac{|C_M| k (\log 2k + \log \frac{1}{\delta})}{q_0 n m}} \quad l_2(\mathcal{W}, \widehat{\mathcal{W}}) \leq B^2 \sqrt{\frac{|C_M| k (\log 2k + \log \frac{1}{\delta})}{q_0 n m^2}},$$

where $q_0 = \min q(C_i)$ and $|C_M| = \max_i |C_i|$.

Proof. By definition of the l_2 distance,

$$\begin{aligned} l_2^2(\mathcal{W}, \widehat{\mathcal{W}}) &= \frac{1}{m^2} \sum_{i=1}^k \sum_{x \in C_i} \left(\frac{1}{p(x)} - \frac{1}{\hat{p}(x)} \right)^2 = \frac{1}{m^2} \sum_{i=1}^k \sum_{x \in C_i} \left(\frac{1}{q(C_i)} - \frac{1}{\hat{q}(C_i)} \right)^2 \\ &\leq \frac{B^4 |C_M|}{m^2} \sum_{i=1}^k \max_i (q(C_i) - \hat{q}(C_i))^2. \end{aligned}$$

Table 1. Normalized mean-squared error (NMSE) for various regression data sets using unweighted, ideal, clustered and kernel-mean-matched training sample reweightings

DATA SET	$ U $	$ S $	n_{test}	UNWEIGHTED	IDEAL	CLUSTERED	KMM
ABALONE	2000	724	2177	.654±.019	.551±.032	.623±.034	.709±.122
BANK32NH	4500	2384	3693	.903±.022	.610±.044	.635±.046	.691±.055
BANK8FM	4499	1998	3693	.085±.003	.058±.001	.068±.002	.079±.013
CAL-HOUSING	16512	9511	4128	.395±.010	.360±.009	.375±.010	.595±.054
CPU-ACT	4000	2400	4192	.673±.014	.523±.080	.568±.018	.518±.237
CPU-SMALL	4000	2368	4192	.682±.053	.477±.097	.408±.071	.531±.280
HOUSING	300	116	206	.509±.049	.390±.053	.482±.042	.469±.148
KIN8NM	5000	2510	3192	.594±.008	.523±.045	.574±.018	.704±.068
PUMAS8NH	4499	2246	3693	.685±.013	.674±.019	.641±.012	.903±.059

The right-hand side of the first line follows from the clustering assumption and the inequality then follows from exactly the same steps as in Proposition 5 and factoring away the sum over the elements of C_i . Finally, it is easy to see that the $\max_i(q(C_i) - \hat{q}(C_i))$ term can be bounded just as in Lemma 3 using a uniform convergence bound, however now the union bound is taken over the clusters rather than unique points. $\square$

Note that when the cluster size is uniform, then $|C_M|k = m$, and the bound above leads to an expression similar to that of Proposition 5.

We used the leaves of a decision tree to define the clusters. A decision tree selects binary cuts on the coordinates of $x \in X$ that greedily minimize a node impurity measure, e.g., MSE for regression (Breiman et al., 1984). Points with similar features and labels are clustered together in this way with the assumption that these will also have similar sampling probabilities.

Several methods for bias correction are compared in Table 1. Each method assigns corrective weights to the training samples. The *unweighted* method uses weight 1 for every training instance. The *ideal* method uses weight $\frac{1}{\Pr[s=1|x]}$, which is optimal but requires the sampling distribution to be known. The *clustered* method uses weight $|C_i \cap U|/|C_i \cap S|$, where the clusters C_i are regression tree leaves with a minimum count of 4 (larger cluster sizes showed similar, though declining, performance). The KMM method uses the approach of Huang et al. (2006b) with a Gaussian kernel and parameters $\sigma = \sqrt{d}/2$ for $x \in \mathbb{R}^d$, $B = 1000$, $\epsilon = 0$. Note that we know of no principled way to do cross-validation with KMM since it cannot produce weights for a held-out set (Sugiyama et al., 2008).

The regression datasets are from LIAAD² and are sampled with $P[s=1|x] = \frac{e^v}{1+e^v}$ where $v = \frac{4w \cdot (x - \bar{x})}{\sigma_{w \cdot (x - \bar{x})}}$, $x \in \mathbb{R}^d$ and $w \in \mathbb{R}^d$ chosen at random from $[-1, 1]^d$. In our experiments, we chose ten random projections w and reported results with the w , for each data set, that maximizes the difference between the unweighted and ideal methods over repeated sampling trials. In this way, we selected bias samplings that are good candidates for bias correction estimation.

² www.liaad.up.pt/~ltorgo/Regression/DataSets.html

For our experiments, we used a version of SVR available from LibSVM³ that can take as input weighted samples, with parameter values $C' = 1$, and $\epsilon = 0.1$ combined with a Gaussian kernel with parameter $\sigma = \sqrt{d/2}$. We report results using normalized mean-squared error (NMSE): $\frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \frac{(y_i - \hat{y}_i)^2}{\sigma_y^2}$, and provide mean and standard deviations for ten-fold cross-validation.

Our results show that reweighting with more reliable counts, due to clustering, can be effective in the problem of sample bias correction. These results also confirm the dependence that our theoretical bounds exhibit on the quantity n_0 . The results obtained using KMM seem to be consistent with those reported by the authors of this technique.⁴

6 Conclusion

We presented a general analysis of sample selection bias correction and gave bounds analyzing the effect of an estimation error on the accuracy of the hypotheses returned. The notion of distributional stability and the techniques presented are general and can be of independent interest for the analysis of learning algorithms in other settings. In particular, these techniques apply similarly to other importance weighting algorithms and can be used in other contexts such that of learning in the presence of uncertain labels. The analysis of the discriminative method of (Bickel et al., 2007) for the problem of covariate shift could perhaps also benefit from this study.

References

- Bickel, S., Brückner, M., Scheffer, T.: Discriminative learning for differing training and test distributions. In: ICML 2007, pp. 81–88 (2007)
- Bousquet, O., Elisseeff, A.: Stability and generalization. JMLR 2, 499–526 (2002)
- Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and regression trees. CRC Press, Boca Raton (1984)
- Cortes, C., Vapnik, V.N.: Support-Vector Networks. Machine Learning 20, 273–297 (1995)
- Devroye, L., Wagner, T.: Distribution-free performance bounds for potential function rules. IEEE Trans. on Information Theory, 601–604 (1979)
- Dudík, M., Schapire, R.E., Phillips, S.J.: Correcting sample selection bias in maximum entropy density estimation. In: NIPS 2005 (2006)
- Elkan, C.: The foundations of cost-sensitive learning. In: IJCAI, pp. 973–978 (2001)
- Fan, W., Davidson, I., Zadrozny, B., Yu, P.S.: An improved categorization of classifier’s sensitivity on sample selection bias. In: ICDM 2005, pp. 605–608. IEEE Computer Society, Los Alamitos (2005)
- Heckman, J.J.: Sample Selection Bias as a Specification Error. Econometrica 47, 151–161 (1979)

³ www.csie.ntu.edu.tw/~cjlin/libsvmtools

⁴ We thank Arthur Gretton for discussion and help in clarifying the choice of the parameters and design of the KMM experiments reported in (Huang et al., 2006b), and for providing the code used by the authors for comparison studies.

- Huang, J., Smola, A., Gretton, A., Borgwardt, K., Schölkopf, B.: Correcting Sample Selection Bias by Unlabeled Data. Technical Report CS-2006-44). University of Waterloo (2006a)
- Huang, J., Smola, A.J., Gretton, A., Borgwardt, K.M., Schölkopf, B.: Correcting sample selection bias by unlabeled data. In: NIPS 2006, pp. 601–608 (2006b)
- Kearns, M., Ron, D.: Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. In: COLT 1997, pp. 152–162 (1997)
- Little, R.J.A., Rubin, D.B.: Statistical analysis with missing data. John Wiley & Sons, Inc., New York (1986)
- Saunders, C., Gammerman, A., Vovk, V.: Ridge Regression Learning Algorithm in Dual Variables. In: ICML 1998, pp. 515–521 (1998)
- Steinwart, I.: On the influence of the kernel on the consistency of support vector machines. JMLR 2, 67–93 (2002)
- Sugiyama, M., Nakajima, S., Kashima, H., von Büna, P., Kawanabe, M.: Direct importance estimation with model selection and its application to covariate shift adaptation. In: NIPS 2008 (2008)
- Vapnik, V.N.: Statistical learning theory. Wiley-Interscience, New York (1998)
- Zadrozny, B.: Learning and evaluating classifiers under sample selection bias. In: ICML 2004 (2004)
- Zadrozny, B., Langford, J., Abe, N.: Cost-sensitive learning by cost-proportionate example weighting. In: ICDM 2003 (2003)

Exploiting Cluster-Structure to Predict the Labeling of a Graph

Mark Herbster

Department of Computer Science
University College London
Gower Street, London WC1E 6BT, England, UK
`m.herbster@cs.ucl.ac.uk`

Abstract. The *nearest neighbor* and the *perceptron* algorithms are intuitively motivated by the aims to exploit the “cluster” and “linear separation” structure of the data to be classified, respectively. We develop a new online perceptron-like algorithm, POUNCE, to exploit both types of structure. We refine the usual margin-based analysis of a perceptron-like algorithm to now additionally reflect the cluster-structure of the input space. We apply our methods to study the problem of predicting the labeling of a graph. We find that when both the quantity and extent of the clusters are small we may improve arbitrarily over a purely margin-based analysis.

1 Introduction

We study the problem of online learning over a graph. Consider the following game for predicting the labeling of a graph. *Nature* presents a vertex $\mathbf{v}_{i_1}$; the *learner* predicts the label of the vertex $\hat{y}_1 \in \{-1, 1\}$; *nature* presents a label y_1 ; *nature* presents a vertex $\mathbf{v}_{i_2}$; the *learner* predicts $\hat{y}_2$; and so forth. The learner’s goal is minimize the total number of mistakes ($|\{t : \hat{y}_t \neq y_t\}|$). If nature is adversarial, the learner will always mispredict; but if nature is regular or simple, there is hope that a learner may make only a few mispredictions. Thus, a methodological goal is to give learners whose total mispredictions can be bounded relative to the “complexity” of nature’s labeling. In [16,15], the *cut size* (the number of edges between disagreeing labels) and diameter were used as a measure of the complexity of a graph’s labeling. We will show that such bounds may be improved arbitrarily by also considering the *cluster-structure* of the graph.

The problem of labeling a graph online is not only of theoretical interest but may also be practically motivated. For example, consider a system which serves advertisements on web pages. The web pages may be identified with the vertices of a graph and the edges as links between pages. The online prediction problem is then that, at a given time t the system may receive a request to serve an advertisement on a particular web page. For simplicity, we assume that there are two alternatives to be served: either advertisement “A” or advertisement “B”. The system then interprets the feedback as the label and then may use this

information in responding to the next request to predict an advertisement for a requested web page.

Recently there has been extensive research into both transduction and semi-supervised learning in the batch setting. A motivating hypothesis is that the structure of the input space may be exploited to improve learning when either the *cluster* or *manifold* condition [5] is satisfied; informally that is

Cluster Condition: If points are in the same cluster, they are likely to be of the same class.

Manifold Condition: The (high-dimensional) data lie (roughly) on a low-dimensional manifold.

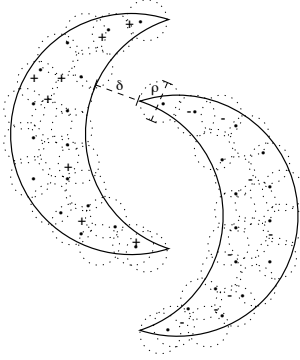
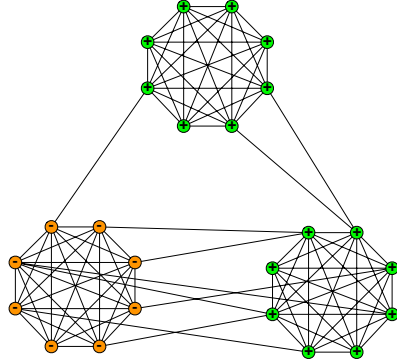
In this paper we will give bounds for an online perceptron-like algorithm which supports this hypothesis. In particular we will take advantage of “clumpiness” in the input space; thus when the inputs are distributed uniformly over a sphere we will find no advantage. However, if it is the case that the input space X is such that it is concentrated into a few dense clusters or if it lies on a smooth low-dimensional manifold, we can then *cover* the input space with a relatively moderate number of balls of modest diameter with respect to the extent of X .

In Section 4 we give a new online algorithm POUNCE with a mistake bound based on the size of the cover of an input space X . The minimum number of balls of (squared) diameter ρ that cover X is denoted as $\mathcal{N}(X, \rho)$. With the assumption that the input space is a subset of an inner-product space which is induced by a graph Laplacian, we refine the classical result of Novikoff [20] in Theorem 2 by incorporating the cover size to give

$$|\mathcal{M}| \leq \mathcal{N}(X, \rho) + \|\mathbf{u}\|^2 \rho + 1.$$

Here $|\mathcal{M}|$ is the cumulative mistakes of our algorithm, and $\|\mathbf{u}\|^2$ is the squared semi-norm of a *separating* classifier. This result significantly improves on the Novikoff bound when the input space consists of a few dense clusters but not uniformly as the squared diameter *may* be four times larger than the origin-based squared radius of Novikoff bound and the additive constant “1”.

A key issue in graph-based transductive learning is how we should use the basic inputs to build a graph for a given algorithm. In Section 5 we apply the previous result in order to understand implications of using the exponential embedding (“heat kernel” [2]) to build a graph. There we will additionally assume as with the “cluster condition” that points in the same ball will have the same label. In Figure 1 we illustrate our results with the classic two moons dataset [2]. Here each “moon” represents and contains points of only a single class. We can characterize this dataset via two types of now label-dependent covers of the input space. Their corresponding cover numbers $\mathcal{N}^\circ$ and $\mathcal{C}^\circ$ differ from $\mathcal{N}$ in that each set in these covers consist only of labeled points of the same class. The first kind of cover is illustrated by the $\mathcal{N}^\circ = 42$ balls which completely cover the input space. The diameter ρ of these balls must be less than the minimal separation between any two opposing labels δ . The second kind of cover $\mathcal{C}^\circ$ assumes now that we have particular knowledge of the unlabeled data. The second cover is

**Fig. 1.** Covering the two moons**Fig. 2.** Three clusters (one in isolation)

in terms of ρ -*path-connected* sets, i.e., every two points is connected by a path of points for which every point in the path is no more than ρ distant from its predecessor and successor in the path. Thus in Figure 1 the 30 black points cover the input space and define a ρ -path-connected cover of size $\mathcal{C}^\circ = 2$. Given such covers of the input space Theorem 8 then implies that there is an embedding of the input space with a graph Laplacian such that the mistakes of our algorithm may be then bounded by

$$|\mathcal{M}| \leq \mathcal{C}^\circ + 1 \leq \mathcal{N}^\circ + 1.$$

This result gives an insight into the value of unlabeled data in transductive (semi-supervised) learning within the context of our setting. From the bound above we see that with no assumptions on the unlabeled data beyond the geometry of the input space, a priori, our mistake bound is $\mathcal{N}^\circ + 1$ whereas increasing the quantity of unlabeled data may induce a cover via a few path-connected subsets such that $\mathcal{C}^\circ \ll \mathcal{N}^\circ$, significantly decreasing the upper bound.

2 Background

Our research builds on the extensive literature concerned with algorithmic variants of the perceptron and their mistake bound analysis; a few recent examples include [18, 8, 10, 13, 4, 6]. Our particular concern is to factor into the usual margin-based analyses a model which can capture simplifying elements of the cluster-structure of the input space. A mistake bound analysis which also incorporates the structure of the input space is that of the second-order perceptron [4]; those results are based on the spectrum of the correlation matrix of the inputs. In the algorithmic luckiness framework [12] an analysis of the max-margin classifier is also given in terms of the cover number of the input space.

Semi-supervised and transductive learning methods suppose that unlabeled data can aid the learner. Thus if the input space is benign as characterized by the cluster and/or manifold conditions, it is expected that the unlabeled data

may be exploited. A common approach is to use the labeled and unlabeled data to build a “graph” which is then used by the learning method. The seminal approach in [3] is to predict with a labeling which is consistent with a minimum label-separating cut. The approach which directly motivates our work is that based on the semi-norm which is induced by the graph Laplacian [22,2,17] which is either directly minimized subject to constraints, or used as a regularizer.

2.1 Prediction on a Graph with a Perceptron

In [16,15] the online graph labeling problem was studied. An aim of those papers was to provide a “natural” interpretation of the bound on the cumulative mistakes of the kernel perceptron. Results were given when the basic kernel was the pseudoinverse of the graph Laplacian. As the current work builds directly on the results in [16,15] we will selectively summarize and elaborate on those prior results.

The graph Laplacian is a positive semidefinite matrix which is defined from the adjacency (weight) matrix of the graph. Let $\mathbf{A}$ be the $n \times n$ symmetric weight matrix of the graph such that $A_{ij} \geq 0$, and define the edge set $E(\mathbf{G}) := \{(i, j) : 0 < A_{ij}\}$; note edges are unordered pairs thus $(i, j) \equiv (j, i)$. The graph Laplacian $\mathbf{G}$ is then the $n \times n$ matrix defined as

$$\mathbf{G} := \mathbf{D} - \mathbf{A}, \quad (1)$$

where $\mathbf{D}\text{diag}(d_1, \dots, d_n)$ and d_i is the *weighted* degree of vertex i , $d_i = \sum_{j=1}^n A_{ij}$. The induced semi-norm is then

$$\|\mathbf{u}\|^2 := \mathbf{u}^\top \mathbf{G} \mathbf{u} = \sum_{(i,j) \in E(\mathbf{G})} A_{ij} (u_i - u_j)^2. \quad (2)$$

The graph is naturally interpreted as an n -vertex resistive network where each edge $(i, j) \in E(\mathbf{G})$ is viewed as a resistor with resistance $\frac{1}{A_{ij}}$. Thus the *effective resistance* $r_{\mathbf{G}}(p, q)$ between vertex p and q is then the potential difference needed to induce a unit current flow between p and q . The effective resistance may be computed via [19],

$$r_{\mathbf{G}}(p, q) = (\mathbf{e}_p - \mathbf{e}_q)^\top \mathbf{G}^+ (\mathbf{e}_p - \mathbf{e}_q), \quad (3)$$

where “ $+$ ” denotes the pseudoinverse and $\mathbf{e}_p$ is the p -th coordinate vector of $\mathbb{R}^n$. The resistance diameter of a graph $R_{\mathbf{G}} := \max_{1 \leq p < q \leq n} r_{\mathbf{G}}(p, q)$ is the maximum of the effective resistances between each of the pairs of vertices on the graph. Surprisingly, both $r_{\mathbf{G}}(\cdot, \cdot)$ and $\sqrt{r_{\mathbf{G}}(\cdot, \cdot)}$ are metrics on the graph [19].

In [15, Theorem 4.2 (with $b = R_{\mathbf{G}}; c = 0$)] using the pseudoinverse of the graph Laplacian mixed with the “constant” function as a kernel the cumulative mistakes of the kernel perceptron was upper bounded by

$$|\mathcal{M}| \leq 2\|\mathbf{u}^*\|^2 R_{\mathbf{G}} + 2. \quad (4)$$

In the above $\mathbf{u}^*$ is the optimal classifier which is correct on the examples; thus

$$\mathbf{u}^* := \arg \min_{\mathbf{u} \in \mathbb{R}^n} \{\|\mathbf{u}\|^2 : u_1 = y_1, \dots, u_\ell = y_\ell\} \quad (5)$$

where $y_i \in \{-1, 1\}$ is the true label of vertex i . Thus if we view the graph as a resistive network and we fix the voltages at vertices $i = 1, \dots, \ell$ to $y_1, \dots, y_\ell$, respectively, then by Thomson's principle [7] $\mathbf{u}^*$ is then the vector of voltages that minimizes the energy dissipation (power). On an unweighted graph ($A_{ij} \in \{0, 1\}$) the energy dissipation and the resistance diameter may themselves be bounded by

$$\|\mathbf{u}^*\|^2 \leq 4\Phi(\mathbf{u}^*) \text{ and } R_{\mathbf{G}} \leq D_{\mathbf{G}},$$

four times the separating cut size and the geodesic diameter of the graph, respectively. The separating cut is the number of edges required to separate the positive and negative labels, and the geodesic diameter of a graph is the maximum of the geodesic distances between each of the pairs of vertices on the graph. In the central result of this paper, Theorem 2, we improve the leading term of the bound (equation (4)) by a factor of two. More significantly, we will see in the following subsection that we may improve over (4) without limit when the graph consists of three or more “dense” clusters.

2.2 Three Clusters Are Hard for the Perceptron

First we will recall the analysis of a graph with two clusters as given in [15, p. 7]; there it was found that the mistakes of the perceptron could be upper bounded by a constant independent of the size of the clusters. Then we will observe that the two-cluster result does not generalize to 3+ clusters. In this discussion, for simplicity, we represent a “cluster” in an unweighted graph by an m -vertex clique.

Following [15] consider two m -cliques (one labeled “+1”, one “-1”) with ℓ arbitrary edges ($\ell < m$) connecting the cliques. Note that between any two vertices there are at least ℓ edge-disjoint paths of length no more than five, and therefore the resistance diameter $R_{\mathbf{G}}$ is at most $5/\ell$ and the cut size is $\Phi(\mathbf{u}^*) = \ell$. Hence by (4) the bound on the cumulative mistakes is the constant 42.

Now consider the addition of a third cluster (m -clique) such that the first two cliques are connected by ℓ edges in proportion to m ($cm < \ell < m$), but the third cluster is connected to the initial two by a constant number of edges independent of m and thus $R_{\mathbf{G}} = \Theta(1)$ (see Figure 2). Thus as m increases, in relative terms the third cluster becomes increasingly remote, but counter to geometric intuition the perceptron upper bound (4) increases as $\Theta(\ell)$. Whereas with POUNCE applied to the three cluster problem an upper bound on mistakes is the constant 20 (Equation (10) with $\rho = \frac{2}{m-1}$, $\mathcal{N}(X, \rho) = 3$, and $\Phi(\mathbf{u}^*) < 2\ell$). The difficulty of the three cluster problem for the perceptron is not only a problem in upper bound but also in performance, as there exists a parameterized three-example separable set such that the perceptron must incur mistakes linear in the parameter. In contrast, the upper bound of POUNCE is a constant. We omit this example for reasons of space, however, see [14].

3 Preliminaries

We denote matrices (conventionally $n \times n$) by capital bold letters and vectors (conventionally $n \times 1$) by small bold case letters. So $\mathbf{M}$ denotes the $n \times n$ matrix $(M_{ij})_{i,j=1}^n$ and $\mathbf{w}$ the n -dimensional column vector $(w_1, \dots, w_n)^\top$ also denoted by $(\mathbf{w}(1), \dots, \mathbf{w}(n))^\top$ where “ $\top$ ” denotes transposition. The identity matrix is denoted by $\mathbf{I}$. We also let $\mathbf{0}$ and $\mathbf{1}$ be the n -dimensional vectors all of whose components equal to zero and one respectively, and $\mathbf{e}_i$ the i -th coordinate vector of $\mathbb{R}^n$. Let $\mathbb{N}$ be the set of natural numbers and $\mathbb{N}_\ell := \{1, \dots, \ell\}$. Let $\mathcal{H}$ denote a Hilbert space. If A and B are sets then the set difference is denoted $A \setminus B$ and the shorthand $A \setminus x := A \setminus \{x\}$.

A symmetric positive semidefinite matrix $\mathbf{M}$ induces a semi-inner product on $\mathbb{R}^n$ which is defined as

$$\langle \mathbf{u}, \mathbf{w} \rangle_{\mathbf{M}} := \mathbf{u}^\top \mathbf{M} \mathbf{w},$$

where $\|\mathbf{w}\|_{\mathbf{M}} := \langle \mathbf{w}, \mathbf{w} \rangle_{\mathbf{M}}$ denotes the associated semi-norm (note that the subscript “ $\mathbf{M}$ ” in both $\langle \cdot, \cdot \rangle_{\mathbf{M}}$ and $\|\cdot\|_{\mathbf{M}}$ may be omitted when clear from the context). The reproducing kernel [1] associated with the above semi-inner product is $\mathbf{K} = \mathbf{M}^+$, where “ $+$ ” denotes the pseudoinverse. We also define the *coordinate spanning set*

$$\mathcal{V}_{\mathbf{M}} := \{\mathbf{v}_i := \mathbf{M}^+ \mathbf{e}_i : i = 1, \dots, n\} \quad (6)$$

and let $\mathcal{H}(\mathbf{M}) := \text{span}(\mathcal{V}_{\mathbf{M}})$. The restriction of the semi-inner product $\langle \cdot, \cdot \rangle_{\mathbf{M}}$ to $\mathcal{H}(\mathbf{M})$ is an inner product on $\mathcal{H}(\mathbf{M})$. The set $\mathcal{V}_{\mathbf{M}}$ acts as “coordinates” for $\mathcal{H}(\mathbf{M})$, that is, if $\mathbf{w} \in \mathcal{H}(\mathbf{M})$ we have

$$\mathbf{w}(i) = \mathbf{e}_i^\top \mathbf{M}^+ \mathbf{M} \mathbf{w} = \mathbf{v}_i^\top \mathbf{M} \mathbf{w} = \langle \mathbf{v}_i, \mathbf{w} \rangle_{\mathbf{M}}, \quad (7)$$

although the vectors $\{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ are not necessarily normalized and are linearly independent only if $\mathbf{M}$ is positive definite. We note that equation (7) is simply the reproducing kernel property [1] for kernel $\mathbf{M}^+$.

A *discrepancy* function $d : \mathcal{X} \times \mathcal{X} \rightarrow [0, \infty]$ is symmetric $d(x, y) = d(y, x)$ and $(d(x, y) = 0) \iff (x = y)$. The diameter $D(X, d)$ of a set $X \subseteq \mathcal{X}$ is the maximum discrepancy between any two points in X , and thus $D(X, d) := \sup_{x, x' \in X} d(x, x')$. The covering number $\mathcal{N}(X, \rho, d)$ is the minimal number of sets of diameter ρ that contain set X . Thus

$$\mathcal{N}(X, \rho, d) := \min_{\{X'_i\}_{i=1}^k} \{k \in \mathbb{N}^+ : D(X'_i, d) \leq \rho, \forall i \in \mathbb{N}_k, \cup_{i=1}^k X'_i \supseteq X\}; \quad (8)$$

if the minimum does not exist then $\mathcal{N}(X, \rho, d) := \infty$. When $X \subseteq \mathcal{H}(\mathbf{M})$ we assume a discrepancy which is a squared norm $d_{\mathbf{M}}(x, y) := \|x - y\|_{\mathbf{M}}^2$ for which we define the abbreviated notation $\mathcal{N}(X, \rho) := \mathcal{N}(X, \rho, d_{\mathbf{M}})$.

3.1 The Signed Laplacian

The graph Laplacian (see (1)) is naturally generalized by the class of symmetric diagonally dominant matrices with nonnegative diagonals which we will refer to

as *signed Laplacians*. Recall that a matrix $\mathbf{M}$ is a *symmetric diagonally dominant* iff $|M_{ii}| \geq \sum_{j \neq i} |M_{ij}|$ for every $i \in \mathbb{N}_n$.

Goldberg et al. [11] introduced the use of the signed Laplacian into semi-supervised learning to explicitly encode dissimilarity relations. The following lemma decomposes the quadratic form $\mathbf{u}^\top \mathbf{M} \mathbf{u}$ into edge and vertex contributions enabling the interpretation of $\mathbf{u}^\top \mathbf{M} \mathbf{u}$ as a smoothness measure of a labeling $\mathbf{u} \in \{-1, 1\}^n$ of an associated graph.

Lemma 1. *If $\mathbf{M}$ is symmetric and $\mathbf{u} \in \mathbb{R}^n$ then*

$$\mathbf{u}^\top \mathbf{M} \mathbf{u} = \sum_{(i,j) \in E^+} |M_{ij}|(u_i - u_j)^2 + \sum_{(i,j) \in E^-} |M_{ij}|(u_i + u_j)^2 + \sum_{i \in \mathbb{N}_n} [M_{ii} - (D_i^+ + D_i^-)] u_i^2, \quad (9)$$

where $E^+ := \{(i, j) : M_{ij} < 0, i < j\}$ is the positive edge set and $E^- := \{(i, j) : M_{ij} > 0, i < j\}$ is the negative edge set and

$$D_i^+ := \sum_{j \in \{k : M_{ik} < 0, k \neq i\}} |M_{ij}|, \quad D_i^- := \sum_{j \in \{k : M_{ik} > 0, k \neq i\}} |M_{ij}|$$

are the positive and negative edge degrees of the i th vertex respectively. If for all $i \in \mathbb{N}_n$ the vertex weight $M_{ii} - (D_i^+ + D_i^-)$ is nonnegative then $\mathbf{M}$ is positive semidefinite.

Proof. Since $\mathbf{M}$ is symmetric then $\mathbf{u}^\top \mathbf{M} \mathbf{u} = 2 \sum_{i=1}^{n-1} \sum_{j=i+1}^n M_{ij} u_i u_j + \sum_{i=1}^n M_{ii} u_i^2$. Therefore as

$$2M_{ij} u_i u_j = \begin{cases} |M_{ij}|(u_i + u_j)^2 - |M_{ij}|(u_i^2 + u_j^2) & M_{ij} \geq 0 \\ |M_{ij}|(u_i - u_j)^2 - |M_{ij}|(u_i^2 + u_j^2) & M_{ij} \leq 0 \end{cases}$$

equation (9) follows immediately. Since the first two terms in (9) are nonnegative, and the third is nonnegative if the vertex weight $M_{ii} - (D_i^+ + D_i^-)$ is nonnegative $\forall i \in \mathbb{N}_n$, then $\mathbf{u}^\top \mathbf{M} \mathbf{u} \geq 0$ for $\mathbf{u} \in \mathbb{R}^n$, and thus $\mathbf{M}$ is positive semidefinite. $\square$

Thus if all the vertex weights are nonnegative $\mathbf{M}$ is a signed Laplacian and $\|\cdot\|_{\mathbf{M}}$ is a semi-norm. The “sign” of the edges reflect dissimilarity (similarity) penalties as when two vertices are connected by a positive (negative) edge its contribution to the semi-norm is zero if the labels are the same (different) and positive otherwise. A matrix with a zero vertex weighting and an empty negative edge set corresponds to the Laplacian matrix defined in (1). A matrix $\mathbf{M}$ is *irreducible* iff there does not exist partitioning sets $P, Q \subset \mathbb{N}_n$, $P \cup Q = \mathbb{N}_n$ such that $M_{pq} = 0$ for each $p \in P$ and $q \in Q$; this is equivalent to stating that the associated graph is *connected*.

4 The POUNCE Algorithm

In a mistake-driven algorithm the mistaken examples implicitly generate a cover of the input space via the hypothesis vector. The motivating idea of the POUNCE

Input: $\{(\mathbf{v}_{i_t}, y_t)\}_{t=1}^\ell \subseteq \mathcal{V}_\mathbf{M} \times \{-1, 1\}$.
Initialization: $\mathbf{w}_2 = \mathbf{0}$; $\mathcal{M} = \{1\}$.
for $t = 2, \dots, \ell$ **do**
 Receive: $i_t \in \{1, \dots, n\}$
 $\eta_t = \arg \min_{j \in \mathcal{M}} \|\mathbf{v}_{i_t} - \mathbf{v}_{i_j}\|$
 Predict: $\hat{y}_t = \text{sign}(y_{\eta_t} + \mathbf{w}_t(i_t) - \mathbf{w}_t(i_{\eta_t}))$
 Receive: y_t
 if $\hat{y}_t = y_t$ **then**
 $\mathbf{w}_{t+1} = \mathbf{w}_t$
 else
 $\mathbf{w}_{t+1} = \mathbf{w}_t + \frac{y_t - y_{\eta_t} - (\mathbf{w}_t(i_t) - \mathbf{w}_t(i_{\eta_t}))}{\|\mathbf{v}_{i_t} - \mathbf{v}_{i_{\eta_t}}\|^2} (\mathbf{v}_{i_t} - \mathbf{v}_{i_{\eta_t}})$
 $\mathcal{M} = \mathcal{M} \cup \{t\}$
end

Fig. 3. The POUNCE Algorithm

(*Projection-Orientated-Using-Nearby-Cover-Elements*) algorithm (see Figure 3) is to explicitly use that cover to design an algorithm whose analysis directly reflects both the margin of the separating hypothesis and the structure of the input space. Although it may be possible to achieve similar bounds for a “second order” algorithm our aim is to design an algorithm whose bound can improve on the kernel perceptron in natural scenarios with no increase in the order of computational complexity. Thus the time complexity of POUNCE is identical to the kernel perceptron that is $O(m_t)$ time required on trial t where m_t is the current total of the cumulative mistakes. Thus the total time required for ℓ trials is $O(m_\ell \ell)$. This is assuming the kernel is provided in advance. However, if we are interested in predicting the labeling of a graph with a (signed) Laplacian we must calculate the pseudoinverse of the (signed) Laplacian. The time required to initially compute this kernel in the general case of an n -vertex graph is pragmatically $O(n^3)$; however, see [9] for a significant improvement when the graph is a tree. As computing the kernel is a one-time cost, the existence of multiple problems or multiple validation experiments on the same graph may effectively offset this initial cost.

The input to POUNCE is an online sequence of vertices and labels from a graph with associated (signed) Laplacian $\mathbf{M}$. The first trial is automatically a mistake. On the t th trial vertex $i_t \in \mathbb{N}_n$ is input to the algorithm as its “coordinate” $\mathbf{v}_{i_t} \in \mathcal{V}_\mathbf{M}$ (see (6)). In order to predict, POUNCE finds the nearest neighboring vertex $\mathbf{v}_{\eta_t}$ to $\mathbf{v}_{i_t}$, among those in the mistake set $\mathcal{M}$, in norm $\|\cdot\|_\mathbf{M}$. The prediction and update rule are based on both the vertex to be predicted and its nearest neighbor. Intuitively, POUNCE is thus a combination of a nearest neighbors algorithm and a perceptron-like algorithm.

Theorem 2. *Let $\mathbf{M}$ be either an irreducible Laplacian or a positive definite signed Laplacian. If $X := \{\mathbf{v}_{i_t}\}_{t=1}^\ell \in \mathcal{V}_\mathbf{M}^\ell$ and $Y := \{y_t\}_{t=1}^\ell \in \{-1, 1\}^\ell$ are the sequences of inputs and associated labels and $\mathcal{M}$ is the set of trials in which the POUNCE algorithm predicted incorrectly, then the cumulative mistakes $|\mathcal{M}|$ are upper bounded by*

$$|\mathcal{M}| \leq \mathcal{N}(X, \rho) + \|\mathbf{u}\|^2 \rho + 1, \quad (10)$$

for all $0 < \rho$, and for all $\mathbf{u} \in \mathbb{R}^n$ such that $\mathbf{u}(i_t)y_t \geq 1$ for all $t \in \mathcal{M}$.

4.1 Proof of Theorem 2

The update step of POUNCE is a projection. We recall the definition of projection.

Definition 3. *The projection of a point $\mathbf{w} \in \mathcal{H}$ onto a closed convex nonempty set $\mathcal{U} \subseteq \mathcal{H}$ is defined by*

$$P(\mathcal{U}; \mathbf{w}) := \arg \min_{\mathbf{u} \in \mathcal{U}} \|\mathbf{u} - \mathbf{w}\|. \quad (11)$$

We recall the following two facts about projection: the first is the pythagorean theorem, and the second the explicit equation for the projection to a hyperplane.

Lemma 4. *If $\mathcal{U} \subseteq \mathcal{H}$ is an affine set and $\mathbf{w} \in \mathcal{H}$, $\mathbf{u} \in \mathcal{U}$ then*

$$\|\mathbf{u} - \mathbf{w}\|^2 = \|\mathbf{u} - P(\mathcal{U}; \mathbf{w})\|^2 + \|P(\mathcal{U}; \mathbf{w}) - \mathbf{w}\|^2, \quad (12)$$

and if $(\mathbf{x}, y) \in \mathcal{H} \setminus \mathbf{0} \times \mathbb{R}$ and $\mathbf{w} \in \mathcal{H}$ then

$$P(\{\mathbf{u} : \langle \mathbf{u}, \mathbf{x} \rangle = y\}; \mathbf{w}) = \mathbf{w} + \frac{y - \langle \mathbf{w}, \mathbf{x} \rangle}{\|\mathbf{x}\|^2} \mathbf{x}. \quad (13)$$

Inspired by the maximum principle [7, p. 7] for the graph Laplacian, we prove the following theorem which holds for the more general signed Laplacian.

Theorem 5. *If $\mathbf{M}$ is a signed Laplacian and $\mathbf{y} \in \{-1, 1\}^\ell$ then*

$$\min_{\mathbf{u} \in U_{\mathbf{y}}} \mathbf{u}^\top \mathbf{M} \mathbf{u} = \min_{\mathbf{u} \in \tilde{U}_{\mathbf{y}}} \mathbf{u}^\top \mathbf{M} \mathbf{u} \quad (14)$$

with

$$U_{\mathbf{y}} := \{\mathbf{u} \in \mathbb{R}^n : u_1 = y_1, \dots, u_\ell = y_\ell\} \text{ and } \tilde{U}_{\mathbf{y}} := \{\mathbf{u} \in \mathbb{R}^n : u_1 y_1 \geq 1, \dots, u_\ell y_\ell \geq 1\}.$$

Proof. Equation (14) is trivially true if $\mathbf{M} = 0$. Consider the case that $\mathbf{M}$ is irreducible. Suppose (14) is false then there exists (see [21, Corollary 27.3.1]) a possibly nonunique vector $\mathbf{v} \in \tilde{U}_{\mathbf{y}}$, such that

$$\mathbf{v}^\top \mathbf{M} \mathbf{v} = \min_{\mathbf{u} \in \tilde{U}_{\mathbf{y}}} \mathbf{u}^\top \mathbf{M} \mathbf{u} < \min_{\mathbf{u} \in U_{\mathbf{y}}} \mathbf{u}^\top \mathbf{M} \mathbf{u}.$$

Let $\iota = \arg \max_{j \in \mathbb{N}_n} |v_j|$ be the index of a component of $\mathbf{v}$ of maximal magnitude. By our supposition $1 < |v_\iota|$. Since the constraints are orthogonal we have that

$$\frac{\partial}{\partial u_\iota} (\mathbf{u}^\top \mathbf{M} \mathbf{u})|_{\mathbf{v}} = 0$$

otherwise $\mathbf{v}$ is not a minimum. Thus computing the partial and upper bounding gives

$$|v_\iota| = \left| \sum_{j \neq \iota} \frac{M_{\iota j} v_j}{M_{\iota \iota}} \right| \leq \sum_{j \neq \iota} \frac{|M_{\iota j}| |v_j|}{|M_{\iota \iota}|}. \quad (15)$$

Since $\mathbf{M}$ is a signed Laplacian it is diagonally dominant, and thus

$$\sum_{j \neq \iota} \frac{|M_{\iota j}|}{|M_{\iota \iota}|} \leq 1. \quad (16)$$

Therefore as $|v_\iota| \geq |v_j|$ for each $j \in \mathbb{N}_n$ inequalities (15) and (16) imply that $1 < |v_\iota| = |v_j|$ for each $j \in J := \{j \in \mathbb{N}_n : M_{\iota j} \neq 0\}$.

For each $j \in J$ the above argument may be iterated and then as we have assumed $\mathbf{M}$ is irreducible we may continue iterating to conclude that

$$1 < |v_1| = \dots = |v_n|.$$

Thus $\frac{\mathbf{v}}{|v_1|} \in \tilde{U}_{\mathbf{y}}$ and as $\frac{\mathbf{v}^\top}{|v_1|} \mathbf{M} \frac{\mathbf{v}}{|v_1|} < \mathbf{v}^\top \mathbf{M} \mathbf{v}$ this contradicts the assumption that $\mathbf{v}$ is minimal hence (14) follows for $\mathbf{M}$ irreducible.

Now, alternatively, suppose $\mathbf{M}$ is reducible then there exists k irreducible matrices $\mathbf{M}^{(1)}, \dots, \mathbf{M}^{(k)}$ such that

$$\mathbf{u}^\top \mathbf{M} \mathbf{u} = \mathbf{u}^{(1)\top} \mathbf{M}^{(1)} \mathbf{u}^{(1)} + \dots + \mathbf{u}^{(k)\top} \mathbf{M}^{(k)} \mathbf{u}^{(k)}$$

for all $\mathbf{u} \in \mathbb{R}^n$ and $\mathbf{u} = (u_1^{(1)}, u_2^{(1)}, \dots, u_1^{(k)}, \dots, u_{i_k}^{(k)})$ with $i_1 + \dots + i_k = n$. We conclude by applying (14) to the k independent problems. $\square$

Proof of Theorem 2: From the algorithm we have that for each $t \in \mathcal{M} \setminus 1$,

$$\mathbf{w}_{t+1} := P(\mathcal{U}_t; \mathbf{w}_t) = \mathbf{w}_t + \frac{y_t - y_{\eta_t} - (\mathbf{w}_t(i_t) - \mathbf{w}_t(i_{\eta_t}))}{\|\mathbf{v}_{i_t} - \mathbf{v}_{i_{\eta_t}}\|^2} (\mathbf{v}_{i_t} - \mathbf{v}_{i_{\eta_t}}),$$

which is the projection of $\mathbf{w}_t$ to the hyperplane

$$\mathcal{U}_t := \{\mathbf{u} \in \mathcal{H}(\mathbf{M}) : \langle \mathbf{u}, \mathbf{v}_{i_t} - \mathbf{v}_{i_{\eta_t}} \rangle = y_t - y_{\eta_t}\} = \{\mathbf{u} \in \mathcal{H}(\mathbf{M}) : \mathbf{u}(i_t) - \mathbf{u}(i_{\eta_t}) = y_t - y_{\eta_t}\},$$

as follows from (13). Thus for every $t \in \{2, \dots, \ell\}$

$$\|\mathbf{w}_t - \mathbf{w}_{t+1}\|^2 = \|\mathbf{u} - \mathbf{w}_t\|^2 - \|\mathbf{u} - \mathbf{w}_{t+1}\|^2, \quad (\mathbf{u} \in \mathcal{U}_t)$$

by Lemma 4 for $t \in \mathcal{M} \setminus 1$ and trivially otherwise. Summing these telescoping equalities for $t = 2, \dots, \ell$ then removing from the sum on the left hand side the terms when $t \notin \mathcal{M}$ as they are zero and on the right bounding the term $-\|\mathbf{u} - \mathbf{w}_{\ell+1}\|^2$ by zero gives

$$\sum_{t \in \mathcal{M} \setminus 1} \|\mathbf{w}_t - \mathbf{w}_{t+1}\|^2 \leq \|\mathbf{u}\|^2 \quad \text{for all } \mathbf{u} \in \mathcal{U}^* := \bigcap_{t \in \mathcal{M} \setminus 1} \mathcal{U}_t. \quad (17)$$

Now if a mistake occurs on trial $t \neq 1$ we have that $1 \leq |y_t - y_{\eta_t} - \mathbf{w}_t(i_t) + \mathbf{w}_t(\eta_t)|$. Since $\mathbf{w}_{t+1} \in \mathcal{U}_t$, then $1 \leq |\mathbf{w}_{t+1}(i_t) - \mathbf{w}_{t+1}(\eta_t) - \mathbf{w}_t(i_t) + \mathbf{w}_t(\eta_t)| = |\langle \mathbf{w}_{t+1} - \mathbf{w}_t, \mathbf{v}_{i_t} - \mathbf{v}_{i_{\eta_t}} \rangle|$. We then apply the Cauchy-Schwarz inequality to obtain that

$$1 \leq \|\mathbf{w}_{t+1} - \mathbf{w}_t\| \|\mathbf{v}_{i_t} - \mathbf{v}_{i_{\eta_t}}\|. \quad (18)$$

Now substituting (18) into (17) gives

$$\sum_{t \in \mathcal{M} \setminus 1} \frac{1}{\|\mathbf{v}_{i_t} - \mathbf{v}_{i_{\eta_t}}\|^2} \leq \|\mathbf{u}\|^2 \quad (\mathbf{u} \in \mathcal{U}^*). \quad (19)$$

Given $X \subset \mathcal{H}$ and $\rho > 0$ such that there are $\mathcal{N}(X, \rho) < \infty$ balls denoted $Z_1, \dots, Z_{\mathcal{N}(X, \rho)}$ of diameter ρ which cover X then define $\mathcal{F}(X, \rho)$ to be the set of the initial trial indices in which a mistake first occurred in a ball (excepting $t = 1$) thus $t \in \mathcal{F}(X, \rho)$ if $t \in \mathcal{M} \setminus 1$ and there exists a j such that $\mathbf{v}_{i_t} \in Z_j$ and there does not exist an $s < t$ such that $\mathbf{v}_{i_s} \in Z_j$ and $s \in \mathcal{M} \setminus 1$. We lower bound the left hand side of (19) by removing from sum over trials those trials which are in $\mathcal{F}(X, \rho)$, hence

$$\sum_{t \in (\mathcal{M} \setminus \mathcal{F}(X, \rho)) \setminus 1} \frac{1}{\|\mathbf{v}_{i_t} - \mathbf{v}_{i_{\eta_t}}\|^2} \leq \|\mathbf{u}\|^2 \quad (\mathbf{u} \in \mathcal{U}^*). \quad (20)$$

Since for every two points $\mathbf{v}, \mathbf{v}'$ in the same ball we have that $\|\mathbf{v} - \mathbf{v}'\|^2 \leq \rho$ then

$$\|\mathbf{v}_{i_t} - \mathbf{v}_{i_{\eta_t}}\|^2 \leq \rho$$

for $t \in (\mathcal{M} \setminus \mathcal{F}(X, \rho)) \setminus 1$. Thus substituting the equation above into (20) we have

$$|\mathcal{M}| - |\mathcal{F}(X, \rho)| - 1 \leq \|\mathbf{u}\|^2 \rho \quad (\mathbf{u} \in \mathcal{U}^*). \quad (21)$$

Substituting the upper bound $|\mathcal{F}(X, \rho)| \leq \mathcal{N}(X, \rho)$ into (21) it follows that the mistake bound (10) holds for $\mathbf{u} \in \mathcal{U}^*$. We proceed to show that if $\mathbf{u} \notin \mathcal{U}^*$ and $\mathbf{u}(i_t)y_t \geq 1$ for all $t \in \mathcal{M}$ then there exists a proxy $\mathbf{u}' \in \mathcal{U}^*$ with $\|\mathbf{u}'\|^2 \leq \|\mathbf{u}\|^2$ which hence proves the theorem.

If $\mathbf{u} \in \mathbb{R}^n$ and $\mathbf{u}(i_t)y_t \geq 1$ for $t \in \mathcal{M}$ then by Theorem 5 there exists a $\mathbf{u}' \in \mathbb{R}^n$ such that

$$(\mathbf{u}')^\top \mathbf{M}(\mathbf{u}') \leq \mathbf{u}^\top \mathbf{M} \mathbf{u} \text{ and } \mathbf{u}'(i_t) = y_t \text{ for } t \in \mathcal{M}.$$

If $\mathbf{M}$ is positive definite then $\mathbf{u}' \in \mathcal{U}^* \subset \mathbb{R}^n = \mathcal{H}(\mathbf{M})$ and we are done; otherwise $\mathbf{M}$ is an irreducible Laplacian. If $\mathbf{M}$ is an irreducible Laplacian then from (2) the vector $\mathbf{1}$ spans the null space of $\mathcal{H}(\mathbf{M})$ and thus

$$\mathbf{z} \in \mathbb{R}^n \text{ and } \mathbf{1}^\top \mathbf{z} = 0 \implies \mathbf{z} \in \mathcal{H}(\mathbf{M}). \quad (22)$$

Set $\mathbf{u}'' := \mathbf{u}' - \mathbf{1}(\frac{\mathbf{1}^\top \mathbf{u}'}{n})$ then (22) implies $\mathbf{u}'' \in \mathcal{H}(\mathbf{M})$ and

$$(\mathbf{u}'')^\top \mathbf{M}(\mathbf{u}'') = (\mathbf{u}')^\top \mathbf{M}(\mathbf{u}') \leq \mathbf{u}^\top \mathbf{M} \mathbf{u}.$$

Finally $\mathbf{u}'' \in \mathcal{U}^*$ since $\mathbf{u}''(i_t) - \mathbf{u}''(\eta_t) = \mathbf{u}'(i_t) - (\frac{\mathbf{1}^\top \mathbf{u}'}{n}) - (\mathbf{u}'(\eta_t) - (\frac{\mathbf{1}^\top \mathbf{u}'}{n})) = \mathbf{u}'(i_t) - \mathbf{u}'(\eta_t) = y_t - y_{\eta_t}$, holds for $t \in \mathcal{M} \setminus 1$. $\square$

5 The Exponential Embedding

In transductive and semi-supervised learning if a graph is not inherent in the problem it is necessary to build the graph from the data. The usual procedure is to use a discrepancy function over the data and build a graph using edge weights derived by the k-NN, ϵ -ball, or the exponential (also known as the heat kernel [22,2]) embedding.

Our model here is that a learning problem is determined by a possibly infinite set $\mathcal{X}$ and a *label function* $\mathcal{Y} : \mathcal{X} \rightarrow \{-1, 1\}$ which are both unknown to the learner. Structure is imposed on $\mathcal{X}$ through a discrepancy $d : \mathcal{X} \times \mathcal{X} \rightarrow [0, \infty]$. The learner is initially given a finite input set $X \subseteq \mathcal{X}$ and $d(X, X)$. Subsequently, the learner will predict a subset of the labels of X in an online fashion. In this section we study the performance of POUNCE if we predict by building a graph Laplacian using the exponential embedding of X .

Definition 6. *If $X = \{x_1, \dots, x_n\} \subseteq \mathcal{X}$ is an indexed finite set, $d : \mathcal{X} \times \mathcal{X} \rightarrow [0, \infty]$ is a discrepancy function, and $a > 0$ is a scale parameter then the exponential embedding of (X, d, a) is the map $F_{(X, d, a)} : X \rightarrow \mathcal{H}(\mathbf{G}^a)$ constructed as follows. First define the Laplacian matrix*

$$G_{ij}^a := \begin{cases} -e^{-ad(x_i, x_j)} & i \neq j \\ \sum_{k \neq i}^n e^{-ad(x_i, x_k)} & i = j \end{cases} \quad (23)$$

then define the map from X to the coordinate spanning set $\mathcal{V}_{\mathbf{G}^a} \subset \mathcal{H}(\mathbf{G}^a)$ (recall (6)),

$$F_{(X, d, a)}(x_i) := \mathbf{v}_i = (\mathbf{G}^a)^+ \mathbf{e}_i. \quad (24)$$

The bound in the following theorem is based on label-dependent cover numbers. Thus we will require the following preliminary definitions. The points $x, x' \in X$ are ρ -path-connected if $d(x, x') \leq \rho$ or if there exists a point $x'' \in X$ such that $d(x, x'') \leq \rho$ and x'', x' are ρ -path-connected. The set X is ρ -path-connected if all pairs of points in X are thus connected. A set X is *connected* if there exists a $\rho > 0$ such that it is ρ -path-connected. The component covering number, $\mathcal{C}(X, \rho, d)$ is the minimal number of ρ -path-connected subsets (components) of X that cover X , thus

$$\mathcal{C}(X, \rho, d) := \min_{\{X'_i\}_{i=1}^k \subseteq X} \left\{ k \in \mathbb{N}^+ : X'_i \text{ is } \rho\text{-path-connected for } i \in \mathbb{N}_k, \bigcup_{i=1}^k X'_i = X \right\}. \quad (25)$$

Observe that $\mathcal{C}(X, \rho, d) \leq \mathcal{N}(X, \rho, d)$ and if $X \subseteq X'$ then $\mathcal{N}(X, \rho, d) \leq \mathcal{N}(X', \rho, d)$ but a superset of X may decrease or increase the component covering number. The label function $\mathcal{Y} : X \rightarrow \{-1, 1\}$ maps the input set to label set. The *separating-cover* (*component separating-cover*) number denoted $\mathcal{N}^\circ(X, \mathcal{Y}, d)$ ($\mathcal{C}^\circ(X, \mathcal{Y}, d)$) is the minimal number of sets of maximum diameter ρ (ρ -path-connected) that contain X such that every set contains points only with a single label and every two points with differing labels are more than ρ distant.

Definition 7. If $X \subseteq \mathcal{X}$ is a set, $\mathcal{Y} : \mathcal{X} \rightarrow \{-1, 1\}$ is a label function and $d : \mathcal{X} \times \mathcal{X} \rightarrow [0, \infty]$ is a discrepancy function then the separating-cover number is

$$\mathcal{N}^\circ(X, \mathcal{Y}, d) := \min_{0 < \rho < \delta} \mathcal{N}(X, \rho, d) \quad (26)$$

and the component separating-cover number is

$$\mathcal{C}^\circ(X, \mathcal{Y}, d) := \min_{0 < \rho < \delta} \mathcal{C}(X, \rho, d) \quad (27)$$

with $\delta = \inf\{d(x^+, x^-) : x^+ \in \mathcal{Y}^{-1}(1) \cap X, x^- \in \mathcal{Y}^{-1}(-1) \cap X\}$ and if the infimum is zero then the cover numbers are ∞ .

Given an exponential embedding $F_{(X, d, a)}$ we consider the performance of the POUNCE algorithm in the following theorem as the parameter $a \rightarrow \infty$. This “tuning” of a minimizes the margin term in (10) thus increasing the cover term. Thus this tuning is optimized for the case in which the data is “aligned” with a small component separating-cover. If the data is not well-aligned a less severe tuning may be more useful in practice.

Theorem 8. If $X \subseteq \mathcal{X}$ is a finite set, $d : \mathcal{X} \times \mathcal{X} \rightarrow [0, \infty]$ is a discrepancy function and X is connected then for any label function $\mathcal{Y} : \mathcal{X} \rightarrow \{-1, 1\}$ and any sequence of labeled examples $\{(x_{i_t}, \mathcal{Y}(x_{i_t}))\}_{t=1}^\ell \in (X \times \{-1, 1\})^\ell$ there exists an $a' > 0$ such that for all $a > a'$ the cumulative mistakes $|\mathcal{M}|$ of the POUNCE algorithm on the embedded sequence $\{(F_{(X, d, a)}(x_{i_t}), \mathcal{Y}(x_{i_t}))\}_{t=1}^\ell$ are bounded by

$$|\mathcal{M}| \leq \mathcal{C}^\circ(X, \mathcal{Y}, d) + 1 \leq \mathcal{N}^\circ(\mathcal{X}, \mathcal{Y}, d) + 1. \quad (28)$$

Proof. As X is finite there exists $\rho, \epsilon > 0$ and a component separating cover $\mathcal{C}^\circ(X, \mathcal{Y}, d)$ of X such that $\mathcal{Y}(x) \neq \mathcal{Y}(x') \rightarrow d(x, x') > \rho + \epsilon$. Define $\mathcal{H}(\mathbf{G}^a)$ from X via (23). Since X is connected, $\mathbf{G}^a$ is irreducible. Given $x_p, x_q \in X$ in the same ρ -path-connected set, there exists a path P from x_p to x_q along fewer than $|X|$ edges such that the discrepancy on each edge is no more than ρ . Therefore the resistance $1/G_{ij}^a$ on each edge (i, j) of the embedding of path P into $\mathcal{H}(\mathbf{G}^a)$ is smaller than $e^{a\rho}$; thus the effective resistance (recalling (3)) between vertex p and q is upper bounded by $\|\mathbf{v}_p - \mathbf{v}_q\|_{\mathbf{G}^a}^2 \leq |X|e^{a\rho}$, as follows from Rayleigh’s monotonicity law (see [15, Corollary 3.1] also [7]). Thus we can cover $\mathcal{V}_{\mathbf{G}^a}$ such that

$$\mathcal{N}(\mathcal{V}_{\mathbf{G}^a}, |X|e^{a\rho}) \leq \mathcal{C}^\circ(X, \mathcal{Y}, d). \quad (29)$$

Therefore from Theorem 2 we may bound the mistakes of the algorithm by

$$|\mathcal{M}| \leq \mathcal{N}(\mathcal{V}_{\mathbf{G}^a}, |X|e^{a\rho}) + \|\mathbf{u}^*\|^2 |X|e^{a\rho} + 1, \quad (30)$$

with $\mathbf{u}^*(i) = \mathcal{Y}(x_i)$ for $i = 1, \dots, |X|$. We upper bound $\|\mathbf{u}^*\|^2$ using the fact that x and x' in the same ρ -path-connected set have the same label,

$$\|\mathbf{u}^*\|^2 = \sum_{1 \leq i \leq j \leq n} (u_i^* - u_j^*)^2 e^{-ad(x_i, x_j)} = \sum_{x_i, x_j \in X : \mathcal{Y}(x_i) \neq \mathcal{Y}(x_j)} 4e^{-ad(x_i, x_j)} \leq 4|X|^2 e^{-a(\rho + \epsilon)}. \quad (31)$$

We proceed by substituting the upper bounds (29) and (31) into (30) to give $|\mathcal{M}| \leq \mathcal{C}^\circ(X, \mathcal{Y}, d) + 1$, for all $a > \frac{3 \ln 4 |X|}{\epsilon}$ as there cannot be a fractional mistake. Finally the assumption $X \subseteq \mathcal{X}$ implies that $\mathcal{C}^\circ(X, \mathcal{Y}, d) \leq \mathcal{N}^\circ(\mathcal{X}, \mathcal{Y}, d)$. $\square$

An interpretation of the above result is that the separating-cover number $\mathcal{N}^\circ(\mathcal{X}, \mathcal{Y}, d)$ is an upper bound which is independent of prior knowledge of a particular set X' to be predicted. However, a supersample $X \supseteq X'$ may potentially induce a component separating-cover (e.g., see Figure 1) such that $\mathcal{C}^\circ(X, \mathcal{Y}, d) \ll \mathcal{N}^\circ(\mathcal{X}, \mathcal{Y}, d)$. Therefore the prior knowledge of X reduces the mistake bound by $\mathcal{N}^\circ(\mathcal{X}, \mathcal{Y}, d) - \mathcal{C}^\circ(X, \mathcal{Y}, d)$ a possibly significant gain from prior knowledge of the input space in this idealized learning scenario.

We observe that the separating-cover number $\mathcal{N}^\circ$ is also an upper bound on the number of mistakes incurred by the online 1-“nearest neighbors” algorithm, because once a mistake is made in a given ball, that mistaken point is always nearer to any other point in that ball than to a point of an opposite label. Thus there can be no more mistakes than balls in the cover. Given a discrepancy function d , there then exists an $a' > 0$ such that the component separating-cover number $\mathcal{C}^\circ$ is the mistake bound of graph geodesic 1-“nearest neighbors” for every discrepancy $d'_a = e^{ad}$ with $a > a'$. These upper bounds are tight in so far as an adversary may select a discrepancy such that a mistake is forced for every ball (component). Thus the bounds in Theorem 8 are tight up to a single additional mistake.

The preceding analysis connects the mistake bound analysis of the exponential embedding to graph geodesic nearest neighbors as $a \rightarrow \infty$. For small a the comparison may mislead as there exists a family of unweighted graphs such that the mistakes of POUNCE is upper bounded by a constant, while the mistakes of geodesic nearest neighbors is linear in the size of the graph as follows from [15, Section 5.1].

The Value of Unlabeled Data

Does unlabeled data help the learner in our framework? In the framework of this section the *initially* unlabeled data is just the input set X and the discrepancy d given to the learner before prediction. Can we obtain similar mistake bounds for predicting in X if instead it is revealed to the learner sequentially as we predict? In the following example we construct a problem for which any algorithm which does not preview the unlabeled input set will incur mistakes linear in the data set size in expectation whereas POUNCE will make no more than three mistakes.

Consider the following learning task illustrated in Figure 4. The task is generated at random as follows. The n points from $\mathbb{R}^2$ to be predicted are at $\{(1, 1), (2, 1), \dots, (n, 1)\}$. An additional, $4n$ points are then situated at the loci $\{(1, 0), (3/2, 0), \dots, (n, 0)\}$ and at $\{(1, 2), (3/2, 2), \dots, (n, 2)\}$. Each of the initial n points is labeled independently $+1$ or -1 with equal probability. We denote the first-coordinates of the positively (negatively) labeled points $\{a_1^+, \dots, a_k^+\}$ ($\{a_1^-, \dots, a_{n-k}^-\}$) and then generate points at $\{(a_1^+, 1/2), \dots, (a_k^+, 1/2)\}$ and also generate the points $\{(a_1^-, 3/2), \dots, (a_{n-k}^-, 3/2)\}$ which path-connect to the their labels. Thus our task has $6n$ points in total and we only consider prediction of the initial n . Any algorithm which does not preview the initially unlabeled data

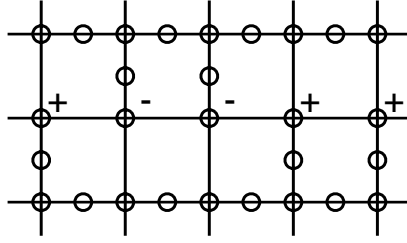


Fig. 4. Two $(1/2)$ -Path-Connected Sets

must incur $n/2$ expected mistakes whereas using Euclidean distance as a discrepancy the positively labeled and negatively labeled points are separated into two $(1/2)$ -path-connected components with no points of differing labels less than a unit distant. Thus from Theorem 8, POUNCE incurs no more than 3 mistakes. Consequently, we observe that although there exists an algorithm that obtains the bound $|\mathcal{M}| \leq \mathcal{N}^\circ(X, \mathcal{Y}, d)$ without prior knowledge of its input set X , no algorithm may exist that obtains either $|\mathcal{M}| \leq \mathcal{C}^\circ(X, \mathcal{Y}, d) + 1$ or (10) without a preview of X or equivalently $\mathbf{M}$, respectively.

6 Discussion

We have presented a novel perceptron-like algorithm POUNCE. We've given a mistake bound analysis of POUNCE which builds on the classic Novikoff analysis via a cover number to provide a finer measure of the structure of the input space. When the input space corresponds to an embedding via a signed Laplacian and its cover is relatively “small,” we may significantly improve over the traditional analysis. This work is a continuation of the researches begun in [16,15], and as such it improves the previous bound [15, Theorem 4.2] at a minimum by a factor two¹ except for an additive constant of “1” even when we cover the space with a single “ball.” The improvement in bound may be arbitrarily large as shown by the three-cluster example in Section 2.2. Furthermore this improvement cannot be obtained by the perceptron [14].

Although the bounds for predicting the online labeling of “small” diameter graphs improve on those given by a straightforward application of the classical halving algorithm, the bounds presented here are weaker than the halving algorithm for “large” diameter graphs as exemplified by an n -vertex line graph (a simple path) [15, p. 8]. Here we can see that a straightforward application of the halving algorithm to the concept class of labelings of a line graph with a cut-size of one leads to a mistake bound of $O(\log n)$. In contrast, the application of Theorem 2 using a cover of $O(\sqrt{n})$ line segments each of diameter $O(\sqrt{n})$ gives the suboptimal mistake bound for POUNCE of $O(\sqrt{n})$; this however improves on the bound for the perceptron of $O(n)$ in [15]. Thus this leaves as an open

¹ There are subtleties in an exact comparison, one of which is the issue of resistance “radius” versus resistance diameter. Surprisingly these may be asymptotically equivalent even in unweighted graphs (see the “flower” graph example of [15, p. 5]).

question whether there is an efficient algorithm which incorporates the strengths of a halving algorithm based analysis along with the analysis presented here.

Acknowledgments. I would like to thank Massimiliano Pontil for useful discussions and anonymous referees for valuable comments.

References

1. Aronszajn, N.: Theory of reproducing kernels. *Trans. Amer. Math. Soc.* 68, 337–404 (1950)
2. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.* 7, 2399–2434 (2006)
3. Blum, A., Chawla, S.: Learning from labeled and unlabeled data using graph min-cuts. In: *Proc. 18th International Conf. on Machine Learning* (2001)
4. Cesa-Bianchi, N., Conconi, A., Gentile, C.: A second-order perceptron algorithm. *SIAM J. Comput.* 34(3), 640–668 (2005)
5. Chapelle, O., Schölkopf, B., Zien, A. (eds.): *Semi-Supervised Learning*. MIT Press, Cambridge (2006)
6. Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., Singer, Y.: Online passive-aggressive algorithms. *J. Mach. Learn. Res.* 7, 551–585 (2006)
7. Doyle, P., Snell, J.: *Random walks and electric networks*. MAA (1984)
8. Freund, Y., Schapire, R.E.: Large margin classification using the perceptron algorithm. *Machine Learning* 37(3), 277–296 (1999)
9. Galeano, S.R., Herbster, M.: A fast method to predict the labeling of a tree. In: *ECML 2007 Workshop on Graph Labeling* (2007)
10. Gentile, C.: The robustness of the p-norm algorithms. *Machine Learning* 53(3), 265–299 (2003)
11. Goldberg, A., Zhu, X., Wright, S.: Dissimilarity in graph-based semi-supervised classification. In: *11th Intl. Conf. on Artificial Intelligence and Statistics* (2007)
12. Herbrich, R., Williamson, R.C.: Algorithmic luckiness. *J. Mach. Learn. Res.* 3, 175–212 (2003)
13. Herbster, M.: Learning additive models online with fast evaluating kernels. In: *Proc. of the 14th Annual Conf. on Computational Learning Theory* (2001)
14. Herbster, M.: A linear lower bound for the perceptron for input sets of constant cardinality. *Research Note RN/08/03*, University College London, London (2008)
15. Herbster, M., Pontil, M.: Prediction on a graph with a perceptron. In: *Advances in Neural Information Processing Systems* 19, pp. 577–584. MIT Press, Cambridge (2007)
16. Herbster, M., Pontil, M., Wainer, L.: Online learning over graphs. In: *Proc. 22nd Intl. Conf. on Machine Learning*, pp. 305–312. ACM Press, New York (2005)
17. Johnson, R., Zhang, T.: On the effectiveness of laplacian normalization for graph semi-supervised learning. *J. Mach. Learn. Res.* 8, 1489–1517 (2007)
18. Kivinen, J., Warmuth, M.: Additive versus exponentiated gradient updates for linear prediction. In: *Proc. 27th Annu. ACM Symp. on Theory of Computing* (1995)
19. Klein, D., Randić, M.: Resistance distance. *J. of Math. Chem.* 12(1), 81–95 (1993)
20. Novikoff, A.: On convergence proofs for perceptrons. In: *Proc. Sympos. Math. Theory of Automata*, Polytechnic Press of Polytechnic Inst. of Brooklyn, NY (1963)
21. Rockafellar, R.: *Convex Analysis*. Princeton University Press, Princeton (1970)
22. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using gaussian fields and harmonic functions. In: *20th Intl. Conf. on Machine Learning* (2003)

A Uniform Lower Error Bound for Half-Space Learning

Andreas Maurer¹ and Massimiliano Pontil²

¹ Adalbertstrasse 55
D-80799 München, Germany
andreasmaurer@compuserve.com

² Dept. of Computer Science
University College London
Malet Pl., WC1E, London, UK
m.pontil@cs.ucl.ac.uk

Abstract. We give a lower bound for the error of any unitarily invariant algorithm learning half-spaces against the uniform or related distributions on the unit sphere. The bound is uniform in the choice of the target half-space and has an exponentially decaying deviation probability in the sample. The technique of proof is related to a proof of the Johnson Lindenstrauss Lemma. We argue that, unlike previous lower bounds, our result is well suited to evaluate the benefits of multi-task or transfer learning, or other cases where an expense in the acquisition of domain knowledge has to be justified.

1 Introduction

We will prove the following lower bound for half-space learning from the uniform distribution σ on the unit sphere $\mathcal{S}$ in $\mathbb{R}^N$.

Theorem 1. *Let $m < N$ and suppose that $f : \mathcal{S}^m \times \{-1, 1\}^m \rightarrow \mathcal{S}$ is any learning algorithm such that*

$$f(V\mathbf{x}, \mathbf{y}) = Vf(\mathbf{x}, \mathbf{y}), \quad \forall \text{unitary } V \text{ on } \mathbb{R}^N. \quad (1)$$

Then for every $u \in \mathcal{S}$

$$\Pr_{\mathbf{x} \sim \sigma^m} \left\{ \text{err}_{\sigma, u}(f(\mathbf{x}, u(\mathbf{x}))) < \frac{1}{\pi} \sqrt{\frac{N-m}{N}} - t \right\} \leq e^{-N(t\pi)^2}.$$

Here $u \in \mathcal{S}$ defines the target function $u(x) = \text{sign}\langle u, x \rangle$, $x \in \mathcal{S}$ and $f(\mathbf{x}, u(\mathbf{x}))$ is the hypothesis returned from the algorithm trained on the sample $\mathbf{x} = (x_1, \dots, x_m)$ labeled by u , where $u(\mathbf{x}) = (u(x_1), \dots, u(x_m))$. The classification error $\text{err}_{\sigma, u}(f(\mathbf{x}, u(\mathbf{x})))$ is the σ -measure of the set of points $x \in \mathcal{S}$ where the sign of $\langle u, x \rangle$ and that of $\langle f(\mathbf{x}, u(\mathbf{x})), x \rangle$ disagree.

The symmetry condition (1) plays an important role in the interpretation of our result. For the proof we only use the fact that symmetry of f implies that

$f(\mathbf{x}, \mathbf{y})$ lies in the subspace spanned by the sample $\mathbf{x}$. Clearly equation (1) is satisfied by all kernel-based algorithms which only depend on the Gramian of $\mathbf{x}$.

Our bound appears weaker than existing lower bounds in [4] and [6] in the sense that it only applies to symmetric algorithms. In another sense it is stronger, because it applies uniformly to *all* target functions, not just to a target function mischievously designed to make the algorithm f fail. It is precisely because of these differences that our bound is suitable for the evaluation of transfer learning or other methods to obtain domain knowledge, where the previous bounds in [4] and [6] are not applicable. A lower bound which holds for *all* algorithms cannot be used to justify the choice of any algorithm. A lower bound which holds for all *symmetric* algorithms can justify the use of an asymmetric algorithm if symmetry-breaking side information is available at a tolerable cost. This will be explained in detail in Section 5.

Theorem 1 is weaker and stronger than the results in [4] and [6] in two other respects. It is weaker, because it is restricted to small sample sizes $m < N$. When we expect small sample sizes and high-dimensional phenomena, this is not a problem. On the other hand Theorem 1 exhibits an exponential concentration property. If the ambient dimension N is large, the quantity $\sqrt{(N-m)/N}/\pi$ becomes an effective performance barrier, as smaller errors have negligible probability.

A simple technique adapts our result to the case when the input marginal μ is not equal to, but absolutely continuous with respect to σ . If the corresponding density function η satisfies $0 < a \leq \eta(x) \leq b$ for almost all $x \sim \sigma$, then the above bound reads

$$\Pr_{\mathbf{x} \sim \mu^m} \left\{ \text{err}_{\mu, u}(f(\mathbf{x}, u(\mathbf{x}))) < \frac{a}{\pi} \sqrt{\frac{N-m}{N}} - t \right\} \leq \exp \left(-N \left(\frac{t\pi}{a} \right)^2 + m \ln b \right),$$

which reduces to the bound in Theorem 1 for $\eta = 1$, i.e. $\mu = \sigma$. We will prove this more general version below (Theorem 2).

We introduce some notation in the next section and give a proof of Theorem 1 in Section 3. Sections 4 and Section 5 briefly discuss previous work and the application to the evaluation of domain knowledge.

2 Notation

We will work in the space $\mathbb{R}^N$ with euclidean inner product $\langle \cdot, \cdot \rangle$, norm $\|x\| = \sqrt{\langle x, x \rangle}$ and euclidean metric $d(x, y) = \|x - y\|$. For $x \in \mathbb{R}^N$ and $F \subseteq \mathbb{R}^N$ we write $d(x, F) = \inf \{d(x, y) : y \in F\}$ and we denote with $F^\perp$ the subspace $F^\perp = \{x : \langle x, y \rangle = 0, \forall y \in F\}$. We write $G_{N, m}$ for the set of all linear m -dimensional subspaces of $\mathbb{R}^N$. If M is a subspace of $\mathbb{R}^N$ then P_M is the orthogonal projection operator onto M .

With $\mathcal{S}$ we denote the unit sphere in $\mathbb{R}^N$, that is $\mathcal{S} = \{x \in \mathbb{R}^N : \|x\| = 1\}$. If $u, v \in \mathcal{S}$ we denote with $\rho(u, v)$ the (shortest) angle between u and v , so that $\rho(\cdot, \cdot)$ is the geodesic metric on $\mathcal{S}$. There is a unique unitarily invariant

probability measure σ (called the Haar measure) on $\mathcal{S}$. If γ is a standard Gaussian random variable (zero mean and unit variance), then σ is defined by

$$\mathbb{E}_{x \sim \sigma} [\psi(x)] = \mathbb{E}_{x \sim \gamma^N} \psi\left(\frac{x}{\|x\|}\right)$$

for every Borel function ψ on $\mathcal{S}$.

An m -tuple $\mathbf{x} = (x_1, \dots, x_m) \in \mathcal{S}^m$ is called a sample. A labeled sample is a member $(\mathbf{x}, \mathbf{y}) \in \mathcal{S}^m \times \{-1, 1\}^m$. If $\mathbf{x} = (x_1, \dots, x_m)$ is a sample then we use $[\mathbf{x}]$ to denote the linear span $\{x_1, \dots, x_m\}$, and if V is a unitary transformation on $\mathbb{R}^N$, we denote with $V\mathbf{x}$ the sample $V\mathbf{x} = (Vx_1, \dots, Vx_m)$.

For labeling we use the sign function

$$\text{sgn}(t) = \begin{cases} 1 & \text{if } t > 0, \\ -1 & \text{otherwise} \end{cases}$$

which differs from the usual definition, but this will simplify notation and otherwise be immaterial in the following. If $u, x \in \mathcal{S}$ we let $u(x) = \text{sgn}(\langle u, x \rangle)$. The target function $u(\cdot)$ defines the open half-space

$$\{x : u(x) = 1\} = \{x : \langle u, x \rangle > 0\}.$$

If $\mathbf{x} \in \mathcal{S}^m$ is a sample then we denote

$$u(\mathbf{x}) = (u(x_1), \dots, u(x_m)) \in \{-1, 1\}^m.$$

Every $u \in \mathcal{S}$ induces a labeled sample $(\mathbf{x}, u(\mathbf{x}))$.

A learning algorithm is a function $f : \mathcal{S}^m \times \{-1, 1\}^m \rightarrow \mathcal{S}$, which assigns to every labeled sample $(\mathbf{x}, \mathbf{y})$ the hypothesis $f(\mathbf{x}, \mathbf{y}) \in \mathcal{S}$. A learning algorithm is called symmetric if

$$f(V\mathbf{x}, \mathbf{y}) = Vf(\mathbf{x}, \mathbf{y})$$

for all unitary V . A symmetric algorithm has no preferred coordinate system. Note that all kernel-based algorithms are symmetric.

For $u, v \in \mathcal{S}$ we denote

$$\Delta(u, v) = \{x : u(x) \neq v(x)\} \subseteq \mathcal{S}.$$

If μ is a probability measure on $\mathcal{S}$ and $u, v \in \mathcal{S}$ then

$$\text{err}_{\mu, u}(v) = \mu(\Delta(u, v))$$

is the error probability for the hypothesis v when the true half-space is u and the underlying input probability is μ .

3 Proofs

In this section we prove the results announced in the introduction.

The idea is the following: The expected error of any hypothesis v is equal to the geodesic distance from v to the target u , divided by π . The hypothesis

generated by a symmetric algorithm lies in $[\mathbf{x}]$, the span of the data, so the error of this hypothesis is lower bounded by the euclidean distance from u to $[\mathbf{x}]$, divided by π . This distance is sharply concentrated at $\sqrt{(N-m)/N}$, as follows from a result of Dasgupta and Gupta [3], given in their proof of the Johnson-Lindenstrauss Lemma.

Proposition 1. *Let μ be a probability measure on $\mathcal{S}$ such that $d\mu(x) = \eta(x)d\sigma(x)$ with $0 < a \leq \eta(x) \leq b$ for almost all $x \sim \sigma$. Then for every symmetric learning algorithm f , every $u \in \mathcal{S}$ and every $t > 0$ we have*

$$\Pr_{\mathbf{x} \sim \mu^m} \{ \text{err}_{\mu,u}(f(\mathbf{x}, u(\mathbf{x}))) < t \} \leq b^m \sup_{M \in G_{N,m}} \Pr_{w \sim \sigma} \left\{ d(w, M) < \frac{t\pi}{a} \right\}.$$

Proof. For any $v \in \mathcal{S}$ we have

$$\text{err}_{\mu,u}(v) = \int_{\Delta(u,v)} \eta d\sigma \geq a \sigma(\Delta(u, v)).$$

Since σ is invariant under rotations in the u - v -plane, it is easily seen that $\sigma(\Delta(u, v))$ is just the angle between u and v in radians, divided by π , that is $\sigma(\Delta(u, v)) = \rho(u, v)/\pi$. Since $\rho(u, v) \geq d(u, v)$, it follows that

$$\text{err}_{\mu,u}(v) \geq d(u, v) \frac{a}{\pi}. \quad (2)$$

Let $(\mathbf{x}, \mathbf{y})$ be an arbitrary labeled sample and let V be the unitary map $V = I$ on $[\mathbf{x}]$ and $V = -I$ on $[\mathbf{x}]^\perp$. By symmetry of f we must have $f(\mathbf{x}, \mathbf{y}) = f(V\mathbf{x}, \mathbf{y}) = V f(\mathbf{x}, \mathbf{y})$, which clearly implies that $f(\mathbf{x}, \mathbf{y}) \in [\mathbf{x}]$. Combining this observation with (2), we have that

$$\text{err}_{\mu,u}(f(\mathbf{x}, u(\mathbf{x}))) \geq d(u, [\mathbf{x}]) \frac{a}{\pi}.$$

We then have

$$\begin{aligned} \Pr_{\mathbf{x} \sim \mu^m} \{ \text{err}_{\mu,u}(f(\mathbf{x}, u(\mathbf{x}))) < t \} &\leq \Pr_{\mathbf{x} \sim \mu^m} \left\{ d(u, [\mathbf{x}]) < \frac{t\pi}{a} \right\} \\ &\leq b^m \Pr_{\mathbf{x} \sim \sigma^m} \left\{ d(u, [\mathbf{x}]) < \frac{t\pi}{a} \right\}, \end{aligned} \quad (3)$$

where we used the upper bound on the density function η in the second inequality.

We now use the unitary symmetry of the Haar measure σ . For any $w \in \mathcal{S}$ we denote with $V_{w \rightarrow u}$ the rotation which takes w to u and $V_{u \rightarrow w}$ its inverse. Then

$$\begin{aligned} \Pr_{\mathbf{x} \sim \sigma^m} \left\{ d(u, [\mathbf{x}]) < \frac{t\pi}{a} \right\} &= \mathbb{E}_{w \sim \sigma} \Pr_{\mathbf{x} \sim \sigma^m} \left\{ d(u, V_{w \rightarrow u}[\mathbf{x}]) < \frac{t\pi}{a} \right\} \\ &= \mathbb{E}_{w \sim \sigma} \Pr_{\mathbf{x} \sim \sigma^m} \left\{ d(V_{u \rightarrow w}u, [\mathbf{x}]) < \frac{t\pi}{a} \right\} \\ &= \mathbb{E}_{w \sim \sigma} \mathbb{E}_{\mathbf{x} \sim \sigma^m} \mathbf{1} \left\{ d(w, [\mathbf{x}]) < \frac{t\pi}{a} \right\}. \end{aligned}$$

Exchanging the two expectations and bounding the expectation in $\mathbf{x}$ by a supremum gives

$$\Pr_{\mathbf{x} \sim \sigma^m} \left\{ d(u, [\mathbf{x}]) < \frac{t\pi}{a} \right\} \leq \sup_{M \in G_{N,m}} \Pr_{w \sim \sigma} \left\{ d(w, M) < \frac{t\pi}{a} \right\},$$

which, together with (3), gives the conclusion. $\square$

In their proof of the Johnson-Lindenstrauss Theorem Dasgupta and Gupta [3, Lemma 2.2] give the following lemma.

Lemma 1. *Let $k < N$ and M be a k -dimensional subspace of $\mathbb{R}^N$ and $\beta \in (0, 1)$. Then*

$$\Pr_{x \sim \sigma} \left\{ \|P_M x\|^2 \leq \frac{\beta k}{N} \right\} \leq \exp \left(\frac{k}{2} (1 - \beta + \ln \beta) \right).$$

We bring this result into a weaker but simpler form which is better suited for our purposes.

Lemma 2. *Let $k < N$ and M be a k -dimensional subspace of $\mathbb{R}^N$ and $t \in (0, \sqrt{(N-k)/N})$. Then*

$$\Pr_{x \sim \sigma} \left\{ d(x, M) \leq \sqrt{\frac{N-k}{N}} - t \right\} \leq e^{-Nt^2}.$$

Proof. For $s \in (0, 1)$ let

$$g(s) = (1-s)^2 - 1 - 2 \ln(1-s).$$

Now, if $h(s) = g(s) - 2s^2$, then $h(0) = 0$ and $h'(s) = 2s^2/(1-s) \geq 0$. This shows that $g(s) \geq 2s^2$. Denote $k' = \dim M^\perp = N - k$ and let $s = t\sqrt{N/k'}$. Then $(1-s)^2 \in (0, 1)$, so by Lemma 1 applied to $M^\perp$ we get

$$\begin{aligned} & \Pr_{x \sim \sigma} \left\{ d(x, M) \leq \sqrt{\frac{k'}{N}} - t \right\} \\ &= \Pr_{x \sim \sigma} \left\{ \|P_{M^\perp} x\|^2 \leq (1-s)^2 \frac{k'}{N} \right\} \\ &\leq \exp \left(\frac{k'}{2} \left(1 - (1-s)^2 + 2 \ln(1-s) \right) \right) \\ &= \exp \left(\frac{-k'g(s)}{2} \right) \leq e^{-k's^2} = e^{-Nt^2}. \end{aligned} \quad \square$$

Now we can state and prove our main result.

Theorem 2. *Let μ be a probability measure on $\mathcal{S}$, such that $d\mu(x) = \eta(x) d\sigma(x)$ with $0 < a \leq \eta(x) \leq b$ for almost all $x \sim \sigma$. Then for every symmetric learning algorithm f and for every target $u \in \mathcal{S}$*

$$\Pr_{\mathbf{x} \sim \mu^m} \left\{ \text{err}_{\mu,u}(f(\mathbf{x}, u(\mathbf{x}))) < \frac{a}{\pi} \sqrt{\frac{N-m}{N}} - t \right\} \leq \exp \left(-N \left(\frac{t\pi}{a} \right)^2 + m \ln b \right).$$

Proof. We have

$$\begin{aligned}
& \Pr_{\mathbf{x} \sim \mu^m} \left\{ \text{err}_{\mu, u} (f(\mathbf{x}, u(\mathbf{x}))) < \frac{a}{\pi} \sqrt{\frac{N-m}{N}} - \frac{a}{\pi} t \right\} \\
& \leq b^m \sup_{M \in G_{N, m}} \Pr_{w \sim \sigma} \left\{ d(w, M) < \sqrt{\frac{N-m}{N}} - t \right\} \\
& \leq b^m e^{-Nt^2},
\end{aligned}$$

where we used Proposition 1 in the first and Lemma 2 in the second inequality. The result follows. $\square$

4 Previous Lower Bounds

In learning theory a lot of attention has been devoted to upper error bounds for learning algorithms, and comparatively little work has been done on lower error bounds. For a deeper understanding of the foundations of the subject, however, lower bounds are interesting and they can serve to establish the tightness of upper bounds.

Ehrenfeucht, Haussler, Kearns and Valiant [4] gave a lower bound on the sample complexity of distribution free PAC learning of a function class $\mathcal{F}$ of VC-dimension d on a domain $\mathcal{X}$. They showed that there is a probability distribution μ on $\mathcal{X}$ such that any learning algorithm requires

$$\Omega \left(\frac{d}{\epsilon} + \frac{1}{\epsilon} \ln \frac{1}{\delta} \right)$$

examples in order to learn every function in $\mathcal{F}$, with an error at most ϵ (as measured by μ) and probability at least $1 - \delta$ in the examples drawn i.i.d. from μ . While this result is a milestone in statistical learning theory, the method of proof, as in [4] or [1], constructs a special distribution μ concentrated in a particularly mischievous way on a set shattered by $\mathcal{F}$, and it can be expected that the distributions underlying realistic learning problems are less pathological.

This deficiency motivated Phil Long [6] to prove the following result pertaining to the learning of the class $\mathcal{F}$ of half-spaces in $\mathbb{R}^N$ from the uniform distribution σ on the unit sphere $\mathcal{S} \subseteq \mathbb{R}^N$: For *any* learning algorithm it takes

$$\Omega \left(\frac{N}{\epsilon} + \frac{1}{\epsilon} \ln \frac{1}{\delta} \right)$$

examples in order to learn *every* half space on $\mathbb{R}^N$ with an error at most ϵ (as measured by μ) and probability at least $1 - \delta$ in the examples drawn i.i.d. from σ . This result replaces the pathological distribution above by a particularly well behaved one and is of considerable importance, because the notion of halfspace learning is central to many learning techniques (support vector machines, perceptron, etc.).

If the sample size m is smaller than the asserted complexity, then these results can be reformulated as follows: For every algorithm f there *exists* a target vector $u \in \mathcal{S}$ such that the probability, that the error of f with respect to u is less than ϵ , is upper bounded by δ . This is substantially different from our result, which restricts f to be symmetric but holds uniformly for *all* target vectors.

5 Evaluation of Domain Knowledge

We now return to the case, where the marginal distribution of the data is given by the Haar measure σ and describe circumstances under which our bound is preferable to the results above.

Complete ignorance of the nature of potential target functions can be expressed as a maximal entropy assumption, which in our case corresponds to the uniform prior σ and assigns the same a-priori probability to all halfspaces. Under this assumption it is reasonable to use an algorithm f^* which is optimal in the sense that it minimizes the expected error of the hypotheses it generates, on average over all training samples and target functions drawn from the uniform distribution. This algorithm, which would correspond to the Bayes-point algorithm as in [5], should therefore minimize the functional

$$\mathcal{E}(f) = \mathbb{E}_{u \sim \sigma} \mathbb{E}_{\mathbf{x} \sim \sigma^m} \text{err}_{\sigma, u}(f(\mathbf{x}, u(\mathbf{x}))).$$

For a labeled sample $(\mathbf{x}, \mathbf{y}) \in \mathcal{S}^m \times \{-1, 1\}^m$ we denote

$$C(\mathbf{x}, \mathbf{y}) = \{u \in \mathcal{S} : u(\mathbf{x}) = \mathbf{y}\}.$$

$C(\mathbf{x}, \mathbf{y})$ is thus the set of all hypotheses consistent with $(\mathbf{x}, \mathbf{y})$, sometimes also called the version-space. Observe that, given $\mathbf{x}$ and u , there is exactly one $\mathbf{y}$ such that $\mathbf{y} = u(\mathbf{x})$, that is $u \in C(\mathbf{x}, \mathbf{y})$. We therefore obtain

$$\begin{aligned} \mathcal{E}(f) &= \pi^{-1} \mathbb{E}_{u \sim \sigma} \mathbb{E}_{\mathbf{x} \sim \sigma^m} \rho(f(\mathbf{x}, u(\mathbf{x})), u) \\ &= \pi^{-1} \mathbb{E}_{\mathbf{x} \sim \sigma^m} \sum_{\mathbf{y} \in \{-1, 1\}^m} \mathbb{E}_{u \sim \sigma} \rho(f(\mathbf{x}, u(\mathbf{x})), u) 1_{C(\mathbf{x}, \mathbf{y})}(u) \\ &= \pi^{-1} \mathbb{E}_{\mathbf{x} \sim \sigma^m} \sum_{\mathbf{y} \in \{-1, 1\}^m} \mathbb{E}_{u \sim \sigma} \rho(f(\mathbf{x}, \mathbf{y}), u) 1_{C(\mathbf{x}, \mathbf{y})}(u), \end{aligned}$$

and, so, the optimal algorithm is given by

$$f^*(\mathbf{x}, \mathbf{y}) = \arg \min_{w \in \mathcal{S}} \mathbb{E}_{u \sim \sigma} \rho(w, u) 1_{C(\mathbf{x}, \mathbf{y})}(u).$$

The minimizer exists and is unique [7], so that this algorithm is indeed well defined. We also have, for any unitary matrix V , that

$$\begin{aligned} \mathbb{E}_{u \sim \sigma} \rho(w, u) 1_{C(V\mathbf{x}, \mathbf{y})}(u) &= \mathbb{E}_{u \sim \sigma} \rho(w, u) 1_{C(\mathbf{x}, \mathbf{y})}(V^{-1}u) \\ &= \mathbb{E}_{u \sim \sigma} \rho(V^{-1}w, u) 1_{C(\mathbf{x}, \mathbf{y})}(u), \end{aligned}$$

so that $f^*(V\mathbf{x}, \mathbf{y}) = Vf^*(\mathbf{x}, \mathbf{y})$. The optimal algorithm f^* is therefore symmetric and the lower bound in Theorem 1 applies.

In summary these considerations show that in the absence of domain knowledge one is led to the use of a symmetric algorithm, with the limitations implied by Theorem 1. These limitations then also imply lower bounds on the functional $\mathcal{E}$, valid for *every* algorithm f , for example

$$\mathcal{E}(f) \geq \mathcal{E}(f^*) \geq \frac{1}{2\pi} \left(1 - e^{-\frac{N-m}{4}}\right) \sqrt{\frac{N-m}{N}},$$

as can be obtained by setting $t = (1/2\pi) \sqrt{(N-m)/N}$ in Theorem 1. Similar bounds cannot be derived from the results in [4] and [6], because they only hold for single target functions constructed in response to the algorithm f .

We were led to the use of the symmetric algorithm f^* by our ignorance of the true distribution of target functions. Suppose now that this distribution, rather than being uniform, is concentrated on a small subset M of the sphere. An example would be the intersection of a low-dimensional subspace with the sphere. As long as we do not know M the uniform prior still represents our knowledge on the potential target functions, and therefore the optimal algorithm is still given by f^* as above. On the other hand, if we have knowledge of M , we can adopt an algorithm f' which only searches M . Since M is small there will be a considerable improvement incurred by replacing f^* with f' . A quantitative guarantee on this improvement can be calculated by applying an upper error bound (using standard techniques) to f' , and by an application of Theorem 1 to f^* .

As a simple example, suppose that M is finite. Theorem 1, a standard result and a union bound show the following: For every target function $u \in M$, with probability at least $1 - \delta$ in $\mathbf{x} \sim \sigma^m$, the difference of the errors of the two algorithms satisfies

$$\text{err}_{\sigma, u}(f^*(\mathbf{x}, u(\mathbf{x}))) - \text{err}_{\sigma, u}(f'(\mathbf{x}, u(\mathbf{x}))) \geq \frac{1}{\pi} \sqrt{\frac{N-m-2\ln \frac{2}{\delta}}{2N}} - \frac{\ln |M| + \ln \frac{2}{\delta}}{m},$$

which can be rather large if $\ln |M| \ll m \ll N$.

If M is infinite, similar high probability bounds for the difference between the errors of the two algorithms can be derived using the VC-dimension of the hypothesis class corresponding to M . Such results cannot be obtained from the bounds in [4] and [6], because they do not hold for every u and may only be valid for a target function outside M . Observe also that the classical lower bounds cannot distinguish between f^* and f' , while Theorem 1 holds only for f^* but not for f' , which is not symmetric.

In practice the symmetry breaking knowledge of M will not come for free, but at a sometimes considerable cost. A case in point is multi-task or transfer-learning (as in [2]), where knowledge of M is obtained from a large number of tasks, with corresponding target functions drawn from M , and the cost of the acquired knowledge takes the form of the sampling burden for these tasks and the increased computational complexity of the transfer learning algorithm.

To justify such an expense it is necessary to compute the savings made in moving from f^* to f' . A rigorous computation of the *guaranteed* savings is possible only by comparing an upper error bound for f' to a lower error bound for f^* , as described above.

Acknowledgments

This work was supported by EPSRC Grants GR/T18707/01 and EP/D071542/1 and by the IST Programme of the European Community, under the PASCAL Network of Excellence IST-2002-506778.

References

- [1] Anthony, M., Bartlett, P.: Learning in Neural Networks: Theoretical Foundations. Cambridge University Press, Cambridge (1999)
- [2] Baxter, J.: A model of inductive bias learning. *Journal of Artificial Intelligence Research* 12, 149–198 (2000)
- [3] Dasgupta, S., Gupta, A.: An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures and Algorithms* 22, 60–65 (2003)
- [4] Ehrenfeucht, A., Haussler, D., Kearns, M., Valiant, L.G.: A general lower bound on the number of examples needed for learning. *Information and Computation* 82(3), 247–251 (1989)
- [5] Herbrich, R., Graepel, T., Campbell, C.: Bayes Point Machines. *Journal of Machine Learning Research* 1, 245–279 (2001)
- [6] Long, P.M.: On the sample complexity of PAC learning halfspaces against the uniform distribution. *IEEE Transactions on Neural Networks* 6(6), 1556–1559 (1995)
- [7] Maurer, A.: An optimization problem on the sphere. Technical Report arXiv:0805.2362

Generalization Bounds for K-Dimensional Coding Schemes in Hilbert Spaces

Andreas Maurer¹ and Massimiliano Pontil²

¹ Adalbertstrasse 55
D-80799 München, Germany
andreasmaurer@compuserve.com

² Dept. of Computer Science
University College London
Malet Pl., WC1E, London, UK
m.pontil@cs.ucl.ac.uk

Abstract. We give a bound on the expected reconstruction error for a general coding method where data in a Hilbert space are represented by finite dimensional coding vectors. The result can be specialized to K-means clustering, nonnegative matrix factorization and the sparse coding techniques introduced by Olshausen and Field.

1 Introduction

We consider the generalization performance of a general class of K -dimensional coding schemes for data drawn from a distribution μ on the unit ball of a Hilbert space H . These schemes encode a data point $x \sim \mu$ as a vector $\hat{y} \in \mathbb{R}^K$, according to the formula

$$\hat{y} = \arg \min_{y \in A} \left(\|x - Ty\|^2 + g(y) \right),$$

where $A \subseteq \mathbb{R}^K$ is some set of *codes* (which we can always assume to span $\mathbb{R}^K$) and $g : \mathbb{R}^K \rightarrow \mathbb{R}_+$ is some *regularizing function* used to encourage or discourage the use of certain codes, but g may also be chosen zero. The pair (A, g) defines the particular *coding scheme*.

$T : \mathbb{R}^K \rightarrow H$ is a linear map, which defines a particular *implementation* of the coding scheme. It embeds the set A of codes in H and yields the set $T(A)$ of exactly codable patterns. If $\hat{y}$ is the code found for x then $\hat{x} = T\hat{y}$ is the reconstructed data point. The quantity

$$f_T(x) = \min_{y \in A} \left(\|x - Ty\|^2 + g(y) \right)$$

is the (regularized) reconstruction error.

Given a coding scheme (A, g) and a finite number of independent observations $x_1, \dots, x_m \in H$, a common sense approach searches for an implementation T_{opt} which is optimal on average over the observed points, that is

$$T_{\text{opt}} = \arg \min_{T \in \mathcal{C}} \frac{1}{m} \sum_{i=1}^m f_T(x_i), \quad (1)$$

where $\mathcal{C}$ denotes some class of linear embeddings $T : \mathbb{R}^K \rightarrow H$. As we shall see, this framework is general enough to include principal component analysis, K -means clustering, non-negative matrix factorization [9] and the sparse coding schemes as proposed in [12].

To give a justification of this approach (which can be regarded as empirical risk minimization) we require that the class of sets $\{T(A) : T \in \mathcal{C}\}$ is uniformly bounded, or, equivalently, that the quantity

$$\|\mathcal{C}\|_A = \sup_{T \in \mathcal{C}} \|T\|_A = \sup_{T \in \mathcal{C}} \sup_{y \in A} \|Ty\|$$

is finite. We then have the following high probability bound on the expected reconstruction error, uniformly valid for all $T \in \mathcal{C}$.

Theorem 1. *Assume that $K > 1$, $\|\mathcal{C}\|_A \geq 1$, that the functions f_T for $T \in \mathcal{C}$, when restricted to the unit ball of H , have range contained in $[0, b]$, and that the measure μ is supported on the unit ball of H . Fix $\delta > 0$.*

Then with probability at least $1 - \delta$ in the observed data $\mathbf{x} \sim \mu^m$ we have for every $T \in \mathcal{C}$ that

$$\mathbb{E}_{x \sim \mu} f_T(x) - \frac{1}{m} \sum_{i=1}^m f_T(x_i) \leq \frac{K}{\sqrt{m}} \left(20 \|\mathcal{C}\|_A + \frac{b}{2} \sqrt{\ln(16m \|\mathcal{C}\|_A^2)} \right) + b \sqrt{\frac{\ln 1/\delta}{2m}}.$$

If $\|\mathcal{C}\|_A < \infty$ and $b < \infty$ our result immediately implies convergence in probability, uniform in all possible implementations of the respective coding scheme. We are not aware of other comparable results for nonnegative matrix factorization [9] or the sparse coding techniques as in [12].

Before providing a proof of Theorem 1 we illustrate its implications in some specific cases of interest.

2 Examples of Coding Schemes

Several coding schemes can be expressed in our framework. We briefly describe these methods and how our result applies.

2.1 Principal Component Analysis

This classical method (PCA) seeks the K -dimensional orthogonal projection which maximizes the projected variance and then uses this projection to encode future data. Let T_P be an isometry which maps $\mathbb{R}^K$ to the range of a projection P . Since

$$\|Px\|^2 = \|x\|^2 - \min_{y \in \mathbb{R}^K} \|x - T_P y\|^2,$$

finding P to maximize the true or empirical expectation of $\|Px\|^2$ is equivalent to finding T to minimize the corresponding expectation of $\min_{y \in \mathbb{R}^K} \|x - Ty\|^2$. If we use the projection P to encode a given $x \in H$ then $Px = T_P \hat{y}$ where $\hat{y} \in \mathbb{R}^K$

is the minimizer $\|x - T_P y\|^2$. We see that PCA is described by our framework upon the identifications $A = \mathbb{R}^K$, $g \equiv 0$ where $\mathcal{C}$ is restricted to the class of isometries $T : \mathbb{R}^K \rightarrow H$. Given $T \in \mathcal{C}$ and $x \in H$ the reconstruction error is

$$f_T(x) = \min_{y \in \mathbb{R}^K} \|x - Ty\|^2.$$

If the data are constrained to be in the unit ball of H , as we generally assume, then it is easily seen that we can take A to be the unit ball of $\mathbb{R}^K$ without changing any of the encodings. We can therefore apply our result with $\|\mathcal{C}\|_A = 1$ and $b = 1$. This is besides the point however, because in the simple case of PCA much better bounds are available ([13], [17]). In fact we will prove a bound of order $\sqrt{K/m}$ in the course of the proof of Theorem 1 (see Lemma 4 below). In [17] local Rademacher averages are used to give faster rates under certain circumstances.

An objection to PCA is, that generic codes have K nonzero components, while for practical and theoretical reasons sparse codes with much less than K nonzero components are preferable.

2.2 K-Means Clustering or Vector Quantization

Here $A = \{e_1, \dots, e_K\}$, where the e_k form an orthonormal basis of $\mathbb{R}^K$ and $g \equiv 0$. An implementation T now defines a set of centers $\{Te_1, \dots, Te_K\}$, the reconstruction error is $\min_{k=1}^K \|x - Te_k\|^2$ and a data point x is coded by the e_k such that Te_k is nearest to x . The algorithm (1) becomes

$$T_{\text{opt}} = \arg \min_{T \in \mathcal{C}} \frac{1}{m} \sum_{i=1}^m \min_{k=1}^K \|x_i - Te_k\|^2.$$

It is clear that every center Te_k has at most unit norm, so that $\|\mathcal{C}\|_A = 1$. Since all data points are in the unit ball we have $\|x - Te_k\|^2 \leq 4$ so we can set $b = 4$ and the bound on the estimation error becomes

$$\left(20 + 2\sqrt{\ln(16m)}\right) \frac{K}{\sqrt{m}} + \sqrt{\frac{8 \ln(1/\delta)}{m}}.$$

The order of this bound matches up to $\sqrt{\ln m}$ the order given in [3] or [14]. To illustrate our method we will also prove the bound

$$\sqrt{18\pi} \frac{K}{\sqrt{m}} + \sqrt{\frac{8 \ln(1/\delta)}{m}}$$

(Theorem 5), which is slightly better than those in [3] or [14]. There is a lower bound of order $\sqrt{K/m}$ in [2], and it is unknown which of the two bounds (upper or lower) is tight.

In K -means clustering every code has only one nonzero component, so that sparsity is enforced in a maximal way. On the other hand this results in a weaker approximation capability of the coding scheme.

2.3 Nonnegative Matrix Factorization

Here A is the cone $A = \left\{ \sum_{k=1}^K \lambda_k e_k : \lambda_i \geq 0 \right\}$ and $g \equiv 0$. A chosen embedding T generates a cone $T(A) \subset H$ onto which incoming data is projected. In the original formulation by Lee and Seung [9] it is postulated that both the data and the vectors Te_k be contained in the positive orthant of some finite dimensional space, but we can drop most of these restrictions, keeping only the requirement that $\langle Te_k, Te_l \rangle \geq 0$ for $1 \leq k, l \leq K$.

No coding will change if we require that $\|Te_k\| = 1$ for all $1 \leq k \leq K$ by a suitable normalization. The set $\mathcal{C}$ is then given by

$$\mathcal{C} = \{T : \mathbb{R}^K \rightarrow H : \|Te_k\| = 1, \langle Te_k, Te_l \rangle \geq 0, 1 \leq k, l \leq K\}.$$

We can restrict A to its intersection with the unit ball in $\mathbb{R}^K$ (see Lemma 2 below) and set $\|\mathcal{C}\|_A = \sqrt{K}$. From Theorem 1 we obtain the bound

$$\frac{K}{\sqrt{m}} \left(20\sqrt{K} + \frac{1}{2}\sqrt{\ln(16mK)} \right) + \sqrt{\frac{\ln(1/\delta)}{2m}}$$

on the estimation error. We do not know of any other generalization bounds for this coding scheme.

Nonnegative matrix factorization appears to encourage sparsity, but cases have been reported where sparsity was not observed [10]. In fact this undesirable behaviour should be generic for exactly codable data. Various authors have therefore proposed additional constraints ([10], [6]). It is clear that additional constraints on $\mathcal{C}$ can only improve generalization and that the passage from A to a subset can only improve our bounds.

2.4 Sparse Coding of Olshausen and Field

In the original formulation [12] $A = \mathbb{R}^K$ but g is one of the functions $g(y) = -\lambda \sum_i e^{-y_i^2}$, $g(y) = \lambda \sum_i \ln(1 + y_i^2)$ or $g(y) = \lambda \sum_i |y_i|$ and $\lambda > 0$ is a regularization parameter which controls how strongly sparsity is to be encouraged. To see how our result applies, we focus on the last and most conventional regularizer $g(y) = \lambda \|y\|_1$. If $\hat{y}$ is a minimizer for $\|x - Ty\|^2 + \lambda \|y\|_1$ with $\|x\| \leq 1$ then

$$\begin{aligned} \lambda \|\hat{y}\| &\leq \lambda \|\hat{y}\|_1 \leq \|x - T\hat{y}\|^2 + \lambda \|\hat{y}\|_1 \\ &\leq \|x - T0\|^2 + \lambda \|0\|_1 = \|x\|^2 \leq 1, \end{aligned}$$

so $\|\hat{y}\| \leq \lambda^{-1}$, which shows that we can equivalently set A to be the ball of radius λ^{-1} in the definition of this coding scheme. We let $\mathcal{C} = \{T : \mathbb{R}^K \rightarrow H : \|T\|_\infty \leq c\}$. Then we have $\|\mathcal{C}\|_A \leq \lambda^{-1}c$. By the same argument as above all f_T have range contained in $[0, 1]$, so the Theorem can be applied with $b = 1$ to yield the bound

$$\frac{K}{\sqrt{m}} \left(\frac{20c}{\lambda} + \frac{1}{2}\sqrt{\ln(16m\lambda^{-2}c)} \right) + \sqrt{\frac{\ln(1/\delta)}{2m}}$$

on the estimation error. It is interesting to observe that increasing the regularization parameter λ , both encourages sparsity and improves estimation. With similar but more complicated methods the Theorem can also be applied to the other regularizers.

The method of Olshausen and Field [12] approximates with a compromise of geometric proximity and sparsity and our result asserts that the observed value of this compromise generalizes to unseen data if enough data have been observed.

3 Proofs

We first introduce some notation, conventions and auxiliary results. Then we set about to prove our main result.

3.1 Notation, Definitions and Auxiliary Results

Throughout H denotes a Hilbert space. The term *norm* and the notation $\|\cdot\|$ and $\langle \cdot, \cdot \rangle$ always refer to the euclidean norm and inner product on $\mathbb{R}^K$ or on H . Other norms are characterized by subscripts. If H_1 and H_2 are any Hilbert spaces $\mathcal{L}(H_1, H_2)$ denotes the vector space of bounded linear transformations from H_1 to H_2 . If $H_1 = H_2$ we just write $\mathcal{L}(H_1) = \mathcal{L}(H_1, H_1)$. With $\mathcal{U}(H_1, H_2)$ we denote the set of isometries in $\mathcal{L}(H_1, H_2)$, that is maps U satisfying $\|Ux\| = \|x\|$ for all $x \in H_1$.

We use $\mathcal{L}_2(H)$ for the set of Hilbert-Schmidt operators on H , which becomes itself a Hilbert space with the inner product $\langle T, S \rangle_2 = \text{tr}(T^*S)$ and the corresponding (Frobenius-) norm $\|\cdot\|_2$.

For $x \in H$ the operator Q_x is defined by $Q_x z = \langle z, x \rangle$. For any $T \in \mathcal{L}_2(H)$ the identity

$$\langle T^*T, Q_x \rangle_2 = \|Tx\|^2 \quad (2)$$

is easily verified.

Suppose that $A \subseteq \mathbb{R}^K$ spans $\mathbb{R}^K$, that H' is any Hilbert space (which could also be $\mathbb{R}^K$). It is easily verified that the quantity

$$\|T\|_A = \sup_{y \in A} \|Ty\|$$

defines a norm on $\mathcal{L}(\mathbb{R}^K, H')$.

We use the following well known result on covering numbers (e.g. Proposition 5 in [4]).

Proposition 1. *Let B be a ball of radius r in an N -dimensional Banach space and $\epsilon > 0$. There exists a subset $B_\epsilon \subset B$ such that $|B_\epsilon| \leq (4r/\epsilon)^N$ and $\forall z \in B, \exists z' \in B_\epsilon$ with $d(z, z') \leq \epsilon$, where d is the metric of the Banach space.*

The following concentration inequality, known as the bounded difference inequality [11], goes back to the work of Hoeffding [5].

Theorem 2. Let μ_i be a probability measure on a space Ω_i , for $i = 1, \dots, m$. Let $\Omega = \prod_{i=1}^m \Omega_i$ and $\mu = \otimes_{i=1}^m \mu_i$ be the product space and product measure respectively. Suppose the function $\Psi : \Omega \rightarrow \mathbb{R}$ satisfies

$$|\Psi(\mathbf{x}) - \Psi(\mathbf{x}')| \leq c_i$$

whenever $\mathbf{x}$ and $\mathbf{x}'$ differ only in the i -th coordinate. Then

$$\Pr_{\mathbf{x} \sim \mu} \{\Psi(\mathbf{x}) - \mathbb{E}_{\mathbf{x}' \sim \mu} \Psi(\mathbf{x}') \geq t\} \leq \exp \left(\frac{-2t^2}{\sum_{i=1}^m c_i^2} \right).$$

Throughout σ_i will denote a sequence of mutually independent random variables, uniformly distributed on $\{-1, 1\}$ and γ_i, γ_{ij} will be (multiplied indexed) sequences of mutually independent Gaussian random variables, with zero mean and unit standard deviation.

If $\mathcal{F}$ is a class of real functions on a space $\mathcal{X}$ and μ a probability measure on $\mathcal{X}$ then for $m \in \mathbb{N}$ the Rademacher and Gaussian complexities of $\mathcal{F}$ w.r.t. μ are defined ([8],[1]) as

$$\begin{aligned} \mathcal{R}_m(\mathcal{F}, \mu) &= \frac{2}{m} \mathbb{E}_{\mathbf{x} \sim \mu^m} \mathbb{E}_{\sigma} \sup_{f \in \mathcal{F}} \sum_{i=1}^m \sigma_i f(x_i), \\ \Gamma_m(\mathcal{F}, \mu) &= \frac{2}{m} \mathbb{E}_{\mathbf{x} \sim \mu^m} \mathbb{E}_{\gamma} \sup_{f \in \mathcal{F}} \sum_{i=1}^m \gamma_i f(x_i) \end{aligned}$$

respectively.

Appropriately scaled Gaussian complexities can be substituted for Rademacher complexities, by virtue of the next Lemma. For a proof see, for example, [8, p. 97].

Lemma 1. For $A \subseteq \mathbb{R}^k$ we have $\mathcal{R}(A) \leq \sqrt{\pi/2} \Gamma(A)$.

The next result is known as Slepian's lemma ([15], [8]).

Theorem 3. Let Ω and Ξ be mean zero, separable Gaussian processes indexed by a common set $\mathcal{S}$, such that

$$\mathbb{E}(\Omega_{s_1} - \Omega_{s_2})^2 \leq \mathbb{E}(\Xi_{s_1} - \Xi_{s_2})^2 \text{ for all } s_1, s_2 \in \mathcal{S}.$$

Then

$$\mathbb{E} \sup_{s \in \mathcal{S}} \Omega_s \leq \mathbb{E} \sup_{s \in \mathcal{S}} \Xi_s.$$

The following result, which generalizes Theorem 8 in [1], plays a central role in our proof.

Theorem 4. Let $\{\mathcal{F}_n : 1 \leq n \leq N\}$ be a finite collection of $[0, b]$ -valued function classes on a space $\mathcal{X}$, and μ a probability measure on $\mathcal{X}$. Then $\forall \delta \in (0, 1)$ we have with probability at least $1 - \delta$ that

$$\max_{n \leq N} \sup_{f \in \mathcal{F}_n} \left[\mathbb{E}_{x \sim \mu} f(x) - \frac{1}{m} \sum_{i=1}^m f(x_i) \right] \leq \max_{n \leq N} \mathcal{R}_m(\mathcal{F}_n, \mu) + b \sqrt{\frac{\ln N + \ln(1/\delta)}{2m}}.$$

Proof. Denote with Ψ_n the function on $\mathcal{X}^m$ defined by

$$\Psi_n(\mathbf{x}) = \sup_{f \in \mathcal{F}_n} \left[\mathbb{E}_{x \sim \mu} f(x) - \frac{1}{m} \sum_{i=1}^m f(x_i) \right], \quad \mathbf{x} \in \mathcal{X}^m.$$

By standard symmetrization (see [16]) we have $\mathbb{E}_{\mathbf{x} \sim \mu^m} \Psi_n(\mathbf{x}) \leq \mathcal{R}_m(\mathcal{F}_n, \mu) \leq \max_{n \leq N} \mathcal{R}_m(\mathcal{F}_n, \mu)$. Modifying one of the x_i can change the value of any $\Psi_n(\mathbf{x})$ by at most b/m , so that by a union bound and the bounded difference inequality (Theorem 2)

$$\Pr \left\{ \max_{n \leq N} \Psi_n > \max_{n \leq N} \mathcal{R}_m(\mathcal{F}_n, \mu) + t \right\} \leq \sum_n \Pr \{ \Psi_n > \mathbb{E} \Psi_n + t \} \leq N e^{-2m(t/b)^2}.$$

Solving $\delta = N e^{-2m(t/b)^2}$ for t gives the result. $\square$

The following lemma was used in Section 2.3.

Lemma 2. Suppose $\|x\| \leq 1$, $\|c_k\| = 1$, $\langle c_k, c_l \rangle \geq 0$, $y \in \mathbb{R}^K$, $y_i \geq 0$. If y minimizes

$$h(y) = \left\| x - \sum_{k=1}^K y_k c_k \right\|^2,$$

then $\|y\| \leq 1$.

Proof. Assume that y is a minimizer of h and $\|y\| > 1$. Then

$$\left\| \sum_{k=1}^K y_k c_k \right\|^2 = \|y\|^2 + \sum_{k \neq l} y_k y_l \langle c_k, c_l \rangle > 1.$$

Let the real function f be defined by $f(t) = h(ty)$. Then

$$\begin{aligned} f'(1) &= 2 \left(\left\| \sum_{k=1}^K y_k c_k \right\|^2 - \left\langle x, \sum_{k=1}^K y_k c_k \right\rangle \right) \\ &\geq 2 \left(\left\| \sum_{k=1}^K y_k c_k \right\|^2 - \left\| \sum_{k=1}^K y_k c_k \right\| \right) \\ &= 2 \left(\left\| \sum_{k=1}^K y_k c_k \right\| - 1 \right) \left\| \sum_{k=1}^K y_k c_k \right\| \\ &> 0. \end{aligned}$$

So f cannot have a minimum at 1, whence y cannot be a minimizer of h . $\square$

3.2 Proof of the Main Results

We now fix a spanning set $A \subseteq \mathbb{R}^K$ and a "regularizer" $g : A \rightarrow \mathbb{R}_+$. Recall that, for $T \in \mathcal{L}(\mathbb{R}^K, H)$, we had introduced the notation

$$f_T(x) = \inf_{y \in A} \left(\|x - Ty\|^2 + g(y) \right), x \in H.$$

Our principal object of study is the function class

$$\mathcal{F} = \left\{ x \mapsto \inf_{y \in A} \left(\|x - Ty\|^2 + g(y) \right) : T \in \mathcal{C} \right\} = \{f_T : T \in \mathcal{C}\},$$

restricted to the unit ball in H , when $\mathcal{C} \subset \mathcal{L}(\mathbb{R}^K, H)$ is some fixed set of candidate implementations of our coding scheme.

To illustrate our method we first consider the somewhat simpler special case of K -means clustering, corresponding to the choices $A = \{e_1, \dots, e_K\}$, $g \equiv 0$ and $\mathcal{C} = \{T : \|T\|_A \leq 1\}$, equivalent to the requirement that $\|Te_k\| \leq 1$ for all $T \in \mathcal{C}$ and all $k \in \{1, \dots, K\}$. As already noted in Section 2.2 the vectors Te_k define the cluster centers.

Theorem 5. *For every $\delta > 0$ with probability greater $1 - \delta$ in the sample $\mathbf{x} \sim \mu^m$ we have for all $T \in \mathcal{C}$*

$$\mathbb{E}_{\mathbf{x} \sim \mu} \min_{k=1}^K \|x - Te_k\|^2 \leq \frac{1}{m} \sum_{i=1}^m \min_{k=1}^K \|x_i - Te_k\|^2 + K \sqrt{\frac{18\pi}{m}} + \sqrt{\frac{8 \ln(1/\delta)}{m}}.$$

Proof. According to [1] we need to bound the Rademacher complexity of the function class $\mathcal{F}$. By Lemma 1 it suffices to bound the corresponding Gaussian complexity, which we shall do using Slepian's Lemma (Theorem 3). We have

$$\mathcal{R}(\mathcal{F}, \mu) \leq \sqrt{\frac{\pi}{2}} \Gamma(\mathcal{F}, \mu) = \sqrt{\frac{\pi}{2}} \frac{2}{m} \mathbb{E}_{\mathbf{x} \sim \mu^m} \mathbb{E}_\gamma \sup_{T \in \mathcal{C}} \sum_{i=1}^m \gamma_i \min_{k=1}^K \|x_i - Te_k\|^2. \quad (3)$$

Now we fix a sample $\mathbf{x}$ and define Gaussian processes Ω and Ξ indexed by $\mathcal{C}$

$$\Omega_T = \sum_{i=1}^m \gamma_i \min_{k=1}^K \|x_i - Te_k\|^2 \quad \text{and} \quad \Xi_T = \sum_{i=1}^m \sum_{k=1}^K \gamma_{ik} \|x_i - Te_k\|^2.$$

Using orthonormality of the γ_i and γ_{ik} we obtain for $T_1, T_2 \in \mathcal{C}$

$$\begin{aligned} \mathbb{E}(\Omega_{T_1} - \Omega_{T_2})^2 &= \sum_{i=1}^m \left(\min_k \|x_i - T_1 e_k\|^2 - \min_k \|x_i - T_2 e_k\|^2 \right)^2 \\ &\leq \sum_{i=1}^m \max_k \left(\|x_i - T_1 e_k\|^2 - \|x_i - T_2 e_k\|^2 \right)^2 \\ &\leq \sum_{i=1}^m \sum_{k=1}^K \left(\|x_i - T_1 e_k\|^2 - \|x_i - T_2 e_k\|^2 \right)^2 \quad (*) \\ &= \mathbb{E}(\Xi_{T_1} - \Xi_{T_2})^2. \end{aligned}$$

By Slepian's Lemma, the triangle inequality, Schwarz' and Jensen's inequalities

$$\begin{aligned}
& \mathbb{E}_\gamma \sup_{T \in \mathcal{C}} \sum_{i=1}^m \gamma_i \min_{k=1}^K \|x_i - T e_k\|^2 \\
&= \mathbb{E}_\gamma \sup_{T \in \mathcal{C}} \Omega_T \\
&\leq \mathbb{E}_\gamma \sup_{T \in \mathcal{C}} \Xi_T \text{ (Slepian)} \\
&= \mathbb{E}_\gamma \sup_{T \in \mathcal{C}} \sum_{i=1}^m \sum_{k=1}^K \gamma_{ik} \|x_i - T e_k\|^2 \\
&\leq 2K \mathbb{E}_\gamma \left\| \sum_{i=1}^m \gamma_i x_i \right\|^2 + K \mathbb{E}_\gamma \left| \sum_{i=1}^m \gamma_i \right| \text{ (triangle and Schwarz)} \\
&\leq 3K \sqrt{m} \text{ (Jensen)}.
\end{aligned}$$

Substitution in (3) yields $\mathcal{R}(\mathcal{F}, \mu) \leq K \sqrt{18\pi/m}$, which, using Theorem 4 with $N = 1$ and $b = 4$ implies the result. $\square$

It is tempting to use the same technique in the general case. Unfortunately an essential step in the application of Slepian's Lemma, marked (*) above, is impossible if A is infinite, so that a more devious path has to be chosen.

The idea is the following: Every implementing map $T \in \mathcal{C}$ can be factored as $T = U \circ S$, where S is a $K \times K$ matrix, $S \in \mathcal{L}(\mathbb{R}^K)$, and U is an isometry, $U \in \mathcal{U}(\mathbb{R}^K, H)$. Suitably bounded $K \times K$ matrices form a compact, finite dimensional set, the complexity of which can be controlled using covering numbers, while the complexity arising from the set of isometries can be controlled with Rademacher and Gaussian averages. Theorem 4 then combines these complexity estimates.

For fixed $S \in \mathcal{L}(\mathbb{R}^K)$ we denote

$$\mathcal{G}_S = \{f_{US} : U \in \mathcal{U}(\mathbb{R}^K, H)\}.$$

Recall the notation $\|\mathcal{C}\|_A = \sup_{T \in \mathcal{C}} \|T\|_A = \sup_{T \in \mathcal{C}} \sup_{y \in A} \|Ty\|$. With $\mathcal{S}$ we denote the set of $K \times K$ -matrices

$$\mathcal{S} = \{S \in \mathcal{L}(\mathbb{R}^K) : \|S\|_A \leq \|\mathcal{C}\|_A\}.$$

Lemma 3. *Assume $\|\mathcal{C}\|_A \geq 1$, that the functions in $\mathcal{F}$, when restricted to the unit ball of H , have range contained in $[0, b]$, and that the measure μ is supported on the unit ball of H . Then with probability at least $1 - \delta$ for all $T \in \mathcal{C}$*

$$\begin{aligned}
& \mathbb{E}_{x \sim \mu} f_T(x) - \frac{1}{m} \sum_{i=1}^m f_T(x_i) \\
&\leq \sup_{S \in \mathcal{S}} \mathcal{R}_m(\mathcal{G}_S, \mu) + \frac{bK}{2} \sqrt{\frac{\ln(16m \|\mathcal{C}\|_A^2)}{m}} + \frac{8 \|\mathcal{C}\|_A}{\sqrt{m}} + b \sqrt{\frac{\ln(1/\delta)}{2m}}.
\end{aligned}$$

Proof. Fix $\epsilon > 0$. The set $\mathcal{S}$ is the ball of radius $\|\mathcal{C}\|_A$ in the K^2 -dimensional Banach space $(\mathcal{L}(\mathbb{R}^K), \|\cdot\|_A)$ so by Proposition 1 we can find a subset $\mathcal{S}_\epsilon \subset \mathcal{S}$, of cardinality $|\mathcal{S}_\epsilon| \leq (4\|\mathcal{C}\|_A/\epsilon)^{K^2}$ such that every member of $\mathcal{S}$ can be approximated by a member of $\mathcal{S}_\epsilon$ up to distance ϵ in the norm $\|\cdot\|_A$.

We claim that for all $T \in \mathcal{C}$ there exist $U \in \mathcal{U}(\mathbb{R}^K, H)$ and $S_\epsilon \in \mathcal{S}_\epsilon$ such that

$$|f_T(x) - f_{US_\epsilon}(x)| < 4\|\mathcal{C}\|_A \epsilon,$$

for all x in the unit ball of H . To see this write $T = US$ with $U \in \mathcal{U}(\mathbb{R}^K, H)$ and $S \in \mathcal{L}(\mathbb{R}^K)$. Then, since U is an isometry, we have

$$\|S\|_A = \sup_{y \in A} \|Sy\| = \sup_{y \in A} \|Ty\| = \|T\|_A \leq \|\mathcal{C}\|_A$$

so that $S \in \mathcal{S}$. We can therefore choose $S_\epsilon \in \mathcal{S}_\epsilon$ such that $\|S_\epsilon - S\|_A < \epsilon$. Then for $x \in H$, with $\|x\| \leq 1$, we have

$$\begin{aligned} |f_T(x) - f_{US_\epsilon}(x)| &= \inf_{y \in A} \left(\|x - USy\|^2 + g(y) \right) - \inf_{y \in A} \left(\|x - US_\epsilon y\|^2 + g(y) \right) \\ &\leq \sup_{y \in A} \left(\|x - USy\|^2 - \|x - US_\epsilon y\|^2 \right) \\ &= \sup_{y \in A} \langle US_\epsilon y - USy, 2x - (USy + US_\epsilon y) \rangle \\ &\leq (2 + 2\|\mathcal{C}\|_A) \sup_{y \in A} \|(S_\epsilon - S)y\| \leq 4\|\mathcal{C}\|_A \epsilon. \end{aligned}$$

Apply Theorem 4 to the finite collection of function classes $\{\mathcal{G}_S : S \in \mathcal{S}_\epsilon\}$ to see that with probability at least $1 - \delta$

$$\begin{aligned} &\sup_{T \in \mathcal{C}} \mathbb{E}_{x \sim \mu} f_T(x) - \frac{1}{m} \sum_{i=1}^m f_T(x_i) \\ &\leq \max_{S \in \mathcal{S}_\epsilon} \sup_{U \in \mathcal{U}(\mathbb{R}^K, H)} \mathbb{E}_{x \sim \mu} f_{US}(x) - \frac{1}{m} \sum_{i=1}^m f_{US}(x_i) + 8\|\mathcal{C}\|_A \epsilon \\ &\leq \max_{S \in \mathcal{S}_\epsilon} \mathcal{R}_m(\mathcal{G}_S, \mu) + b \sqrt{\frac{\ln |\mathcal{S}_\epsilon| + \ln(1/\delta)}{2m}} + 8\|\mathcal{C}\|_A \epsilon \\ &\leq \sup_{S \in \mathcal{S}} \mathcal{R}_m(\mathcal{G}_S, \mu) + \frac{bK}{2} \sqrt{\frac{\ln(16m\|\mathcal{C}\|_A^2)}{m}} + \frac{8\|\mathcal{C}\|_A}{\sqrt{m}} + b \sqrt{\frac{\ln(1/\delta)}{2m}}, \end{aligned}$$

where the last line follows from the known bound on $|\mathcal{S}_\epsilon|$, subadditivity of the square root and the choice $\epsilon = 1/\sqrt{m}$. $\square$

To complete the proof of Theorem 1 we now fix some $S \in \mathcal{S}$ and focus on the corresponding function class $\mathcal{G}_S$. Observe that for an isometry $U \in \mathcal{U}(\mathbb{R}^K, H)$ the

operator U^*U is the identity on $\mathbb{R}^K$ and that UU^* is the orthogonal projection onto the range of U . We therefore have, for $x \in H$,

$$\begin{aligned} \inf_{y \in A} \|x - USy\|^2 &= \|x - UU^*x\|^2 + \inf_{y \in A} \|UU^*x - USy\|^2 \\ &= \|x\|^2 - \|UU^*x\|^2 + \inf_{y \in A} \|U^*x - Sy\|^2 \end{aligned}$$

so that $\mathcal{G}_S = \mathcal{D} + \mathcal{E}_S$, where

$$\begin{aligned} \mathcal{D} &= \left\{ x \mapsto \|x\|^2 - \|UU^*x\|^2 : U \in \mathcal{U}(\mathbb{R}^K, H) \right\} \\ \mathcal{E}_S &= \left\{ x \mapsto \inf_{y \in A} \|U^*x - Sy\|^2 + g(y) : U \in \mathcal{U}(\mathbb{R}^K, H) \right\}. \end{aligned}$$

We will bound the Rademacher complexities of these two function classes in turn.

Observe that the function class $\mathcal{D}$ is the class of reconstruction errors of PCA, so the next lemma and an application of Theorem 4 with $N = 1$ and $b = 1$ also give a generalization bound for PCA of order $\sqrt{K/m}$.

Lemma 4. $\mathcal{R}(\mathcal{D}, \mu) \leq 2\sqrt{K/m}$.

Proof. For $z \in H$ define the outer product operator Q_z by $Q_z x = \langle x, z \rangle z$. With $\langle \cdot, \cdot \rangle_2$ and $\|\cdot\|_2$ denoting the Hilbert-Schmidt inner product and norm respectively we have for $\|x_i\| \leq 1$

$$\begin{aligned} \mathbb{E}_\sigma \sup_{f \in \mathcal{D}} \sum_{i=1}^m \sigma_i f(x_i) &= \mathbb{E}_\sigma \sup_{U \in \mathcal{U}} \sum_{i=1}^m \sigma_i \left(\|x_i\|^2 - \|UU^*x_i\|^2 \right) \\ &= \mathbb{E}_\sigma \sup_{U \in \mathcal{U}} \left\langle \sum_{i=1}^m \sigma_i Q_{x_i}, UU^* \right\rangle_2 \\ &\leq \mathbb{E}_\sigma \left\| \sum_{i=1}^m \sigma_i Q_{x_i} \right\|_2 \sup_{U \in \mathcal{U}} \|UU^*\|_2 \\ &\leq \sqrt{mK}, \end{aligned}$$

since the Hilbert-Schmidt norm of a K -dimensional projection is $\sqrt{K}$. The result follows upon multiplication with $2/m$ and taking the expectation in μ^m . $\square$

Lemma 5. For any $S \in \mathcal{L}(\mathbb{R}^K)$ we have

$$\mathcal{R}(\mathcal{E}_S, \mu) \leq \frac{4(1 + \|S\|_A)K}{\sqrt{m}} \sqrt{\frac{\pi}{2}}.$$

Proof. Let $\|x_i\| \leq 1$ and define Gaussian processes Ω_U and Ξ_U indexed by $\mathcal{U}(\mathbb{R}^K, H)$

$$\begin{aligned} \Omega_U &= \sum_{i=1}^m \gamma_i \inf_{y \in A} \left(\|U^*x_i - Sy\|^2 + g(y) \right) \\ \Xi_U &= 2(1 + \|S\|_A) \sum_{k=1}^K \sum_{i=1}^m \gamma_{ik} \langle x_i, Ue_k \rangle, \end{aligned}$$

where the e_k are the canonical basis of $\mathbb{R}^K$. For $U_1, U_2 \in \mathcal{U}(\mathbb{R}^K, H)$ we have

$$\begin{aligned} \mathbb{E}(\Omega_{U_1} - \Omega_{U_2})^2 &\leq \sum_{i=1}^m \sup_{y \in A} \langle U_1^* x_i - U_2^* x_i, U_1^* x_i + U_2^* x_i - 2Sy \rangle^2 \\ &\leq \sum_{i=1}^m \|U_1^* x_i - U_2^* x_i\|^2 \sup_{y \in A} \|U_1^* x_i + U_2^* x_i - 2Sy\|^2 \\ &\leq 4(1 + \|S\|_A)^2 \sum_{i=1}^m \sum_{k=1}^K (\langle x_i, U_1 e_k \rangle - \langle x_i, U_2 e_k \rangle)^2 \\ &= \mathbb{E}(\Xi_{U_1} - \Xi_{U_2})^2. \end{aligned}$$

It follows from Lemma 1 and Slepian's lemma (Theorem 3) that

$$\mathcal{R}_m(\mathcal{E}_S, \mu) \leq \mathbb{E}_{\mathbf{x} \sim \mu^m} \frac{2}{m} \sqrt{\frac{\pi}{2}} \mathbb{E}_\gamma \sup_U \Xi_U,$$

so the result follows from the following inequalities, using Schwarz' and Jensens inequality, the orthonormality of the γ_{ik} and the fact that $\|x_i\| \leq 1$ on the support of μ .

$$\begin{aligned} \mathbb{E}_\gamma \sup_U \Xi_U &= 2(1 + \|S\|_A) \mathbb{E} \sup_U \sum_{k=1}^K \left\langle \sum_{i=1}^m \gamma_{ik} x_i, U e_k \right\rangle \\ &\leq 2(1 + \|S\|_A) \sum_{k=1}^K \mathbb{E} \left\| \sum_{i=1}^m \gamma_{ik} x_i \right\| \\ &\leq 2(1 + \|S\|_A) K \sqrt{m}. \end{aligned} \quad \square$$

Using the subadditivity of the Rademacher complexity, the last two results give for $K > 1$ and $\|\mathcal{C}\|_A \geq 1$

$$\begin{aligned} \sup_{S \in \mathcal{S}} \mathcal{R}_m(\mathcal{G}_S, \mu) &\leq \mathcal{R}_m(\mathcal{D}, \mu) + \sup_{S \in \mathcal{S}} \mathcal{R}_m(\mathcal{E}_S, \mu) \\ &\leq \frac{1}{\sqrt{m}} \left(2\sqrt{K} + 8K \|\mathcal{C}\|_A \sqrt{\frac{\pi}{2}} \right) \leq \frac{12K \|\mathcal{C}\|_A}{\sqrt{m}}, \end{aligned}$$

and substitution in Lemma 3 gives Theorem 1.

Acknowledgments

This work was supported by EPSRC Grants GR/T18707/01 and EP/D071542/1 and by the IST Programme of the European Community, under the PASCAL Network of Excellence IST-2002-506778.

References

- [1] Bartlett, P.L., Mendelson, S.: Rademacher and Gaussian Complexities: Risk Bounds and Structural Results. *Journal of Machine Learning Research* 3, 463–482 (2002)
- [2] Bartlett, P., Linder, T., Lugosi, G.: The minimax distortion redundancy in empirical quantizer design. *IEEE Transactions on Information Theory* 44, 1802–1813 (1998)

- [3] Biau, G., Devroye, L., Lugosi, G.: On the performance of clustering in Hilbert spaces. *IEEE Transactions on Information Theory* 54, 781–790 (2008)
- [4] Cucker, F., Smale, S.: On the mathematical foundations of learning. *Bulletin of the American Mathematical Society* 39(1), 1–49 (2001)
- [5] Hoeffding, W.: Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association* 58, 13–30 (1963)
- [6] Hoyer, P.O.: Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research* 5, 1457–1469 (2004)
- [7] Koltchinskii, V., Panchenko, D.: Empirical margin distributions and bounding the generalization error of combined classifiers. *The Annals of Statistics* 30(1), 1–50 (2002)
- [8] Ledoux, M., Talagrand, M.: *Probability in Banach Spaces*. Springer, Heidelberg (1991)
- [9] Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. *Nature* 401, 788–791 (1999)
- [10] Li, S.Z., Hou, X., Zhang, H., Cheng, Q.: Learning spatially localized parts-based representations. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Hawaii, USA, vol. I, pp. 207–212 (2001)
- [11] McDiarmid, C.: Concentration. In: *Probabilistic Methods of Algorithmic Discrete Mathematics*, pp. 195–248. Springer, Berlin (1998)
- [12] Olshausen, B.A., Field, D.J.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature* 381, 607–609 (1996)
- [13] Shawe-Taylor, J., Williams, C.K.I., Cristianini, N., Kandola, J.S.: On the eigen-spectrum of the Gram matrix and the generalization error of kernel-PCA. *IEEE Transactions on Information Theory* 51(7), 2510–2522 (2005)
- [14] Wigelius, O., Ambroladze, A., Shawe-Taylor, J.: Statistical analysis of clustering with applications (preprint, 2007)
- [15] Slepian, D.: The one-sided barrier problem for Gaussian noise. *Bell System Tech. J.* 41, 463–501 (1962)
- [16] van der Vaart, A.W., Wallner, J.A.: *Weak Convergence and Empirical Processes*. Springer, Heidelberg (1996)
- [17] Zwald, L., Bousquet, O., Blanchart, G.: Statistical properties of kernel principal component analysis. *Machine Learning* 66(2-3), 259–294 (2006)

Learning and Generalization with the Information Bottleneck

Ohad Shamir¹, Sivan Sabato^{1,3}, and Naftali Tishby^{1,2}

¹ School of Computer Science and Engineering

² Interdisciplinary Center for Neural Computation,
The Hebrew University, Jerusalem 91904, Israel

³ IBM Research Laboratory in Haifa, Haifa 31905, Israel
{ohadsh,sivan_sabato,tishby}@cs.huji.ac.il

Abstract. The Information Bottleneck is an information theoretic framework that finds concise representations for an ‘input’ random variable that are as relevant as possible for an ‘output’ random variable. This framework has been used successfully in various supervised and unsupervised applications. However, its learning theoretic properties and justification remained unclear as it differs from standard learning models in several crucial aspects, primarily its explicit reliance on the joint input-output distribution. In practice, an empirical plug-in estimate of the underlying distribution has been used, so far without any finite sample performance guarantees. In this paper we present several formal results that address these difficulties. We prove several finite sample bounds, which show that the information bottleneck can provide concise representations with good generalization, based on smaller sample sizes than needed to estimate the underlying distribution. The bounds are non-uniform and adaptive to the complexity of the specific model chosen. Based on these results, we also present a preliminary analysis on the possibility of analyzing the information bottleneck method as a learning algorithm in the familiar performance-complexity tradeoff framework. In addition, we formally describe the connection between the information bottleneck and minimal sufficient statistics.

1 Introduction

The Information Bottleneck (IB) method, introduced in [23], is an information-theoretic framework for extracting relevant components of an ‘input’ random variable X , with respect to an ‘output’ random variable Y . This is performed by finding a *compressed*, non-parametric and model-independent representation T of X , that is most *informative* about Y . Formally speaking, the notion of compression is quantified by the mutual information between T and X , while the informativeness is quantified by the mutual information between T and Y . A scalar Lagrange multiplier β smoothly controls the tradeoff between these two quantities.

The method has proven to be useful for a number of important applications (see [24, 8, 21] and references therein), but its learning theoretic justification has

remained unclear, for two main reasons: (i) The method assumes that the joint distribution of X and Y is known, and uses it explicitly. This stands in contrast to most finite-sample based machine learning algorithms. In practice, the empirical co-occurrence distribution is used to calculate a plug-in estimate of the IB functional, but without explicit regularization, finite-sample generalization bounds or error guarantees of any kind. Moreover, it was not clear what is left to be learned if it is assumed that this distribution is known. (ii) IB is formally related to classical information theoretic problems, such as Rate-Distortion theory and Coding with Side-Information. It is, however, unclear why maximizing mutual information about Y is useful for any “natural” learning theoretic model, and in particular how it is related to classification error.

In this paper we provide rigorous answers to some of the above issues concerning the IB framework. We focus on a learning theoretic analysis of this framework, where X and Y are assumed to be discrete, and the empirical distribution of $p(x, y)$ is used as a plug-in for the true distribution. We develop several finite sample bounds, and show that despite this use of plug-in estimation, the IB framework can actually generalize quite well, with realistic sample sizes that can be much smaller than the dimensionality of this joint distribution, provided that we are looking for a reasonably *simple* representation T of our data. In fact, it is exactly the reliance of the framework on explicit manipulation of the joint distribution that allows us to derive non-uniform bounds that are adaptive to the complexity of the specific model chosen. In addition, we present a preliminary analysis regarding the question in which settings the information bottleneck can be seen as a standard learning algorithm, trading off a risk-like term and a regularization term controlling the generalization. Finally, we discuss its utility as a natural extension of the concept of minimal sufficient statistics for discrimination.

The paper is organized as follows. In Sec. 2, we formally present the information bottleneck framework and the notation used in the paper. We then turn to analyze its finite sample behavior in Sec. 3. Sec. 4 discusses the characteristics of the information bottleneck as a learning algorithm, while its relation to minimal sufficient statistics is considered in Sec. 5. Selected proofs are presented in Sec. 6; Full proofs can be found in [19]. We finish with a discussion in Sec. 7.

2 The Information Bottleneck Framework

In this section we explain and formally describe the basic information bottleneck (IB) framework. This framework has several variants and extensions, both to multivariate variables and to continuous representations (see [20, 4] for more details), but these are not the focus of this paper.

The IB framework attempts to find a simple representation of one random variable X through an auxiliary variable T , which is relevant to another random variable Y . Let us first exemplify how the IB method can be used for both supervised and unsupervised learning. Consider the area of text analysis. A typical unsupervised problem can be clustering documents based on their

word-statistics in order to discover similarities and relationships between them. In this case the X variable is taken as the document identity (typically considered as “bags of words”) and the Y as the words in the documents. In this case, the T variable will be clusters of documents with similar word-statistics, based, for instance, on the “the two sample problem” [13] similarity measure.

In a typical supervised application in this domain, X can denote the words while Y are topic-labels of the documents. Here T are clusters of words that are (approximately) sufficient for document categorization [24]. In all the applications a variable β allows us to smoothly move between a low resolution - highly compressed - solution, to a solution with higher resolution and more information about Y . This form of dimensionality reduction, a special case of the information bottleneck, was introduced under the name of distributional clustering in [16], and has proven to be quite effective in analyzing high dimensional data [2, 9].

In this work, we assume that X and Y take values in the finite sets $\mathcal{X}$ and $\mathcal{Y}$ respectively, and use x and y respectively to denote elements of these sets. The basic quantity that is utilized in the IB framework is Shannon’s mutual information between random variables, which for discrete variables is formally defined as:

$$I(X; Y) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) \log \left(\frac{p(x, y)}{p(x)p(y)} \right).$$

Mutual information is well known to be the unique measure of informativeness, up to a multiplicative constant, under very mild assumptions [5]. The IB functional is built upon the relationship between minimal sufficiency and information. It captures a tradeoff between minimality of the representation of X , achieved by minimizing $I(X; T)$, and sufficiency of information on Y , achieved by constraining the value of $I(Y; T)$. The auxiliary variable T is thus determined by the minimization of the IB-Lagrangian

$$\mathcal{L}_{IB}[p(t|x)] = I(X; T) - \beta I(Y; T) \quad (1)$$

with respect to the mapping $p(t|x)$. T is subject to the Markovian relation $T - X - Y$, and $p(t|x)$ is subject to the obvious normalization constraints. The tradeoff parameter β is a positive Lagrange multiplier associated with the constraint on $I(Y; T)$. Formally, T is defined over some space $\mathcal{T}$, but the elements of this space are arbitrary - only the probabilistic relationships between T and X, Y are relevant.

The solutions of this constrained optimization problem are characterized by the *bottleneck equations*,

$$\begin{cases} p(t|x) = \frac{p(t)}{Z(\beta, x)} \exp(-\beta D_{KL}[p(y|x) || p(y|t)]) \\ p(t) = \sum_{x \in \mathcal{X}} p(t|x)p(x) \\ p(y|t) = \sum_{x \in \mathcal{X}} p(y|x)p(x|t) \end{cases} \quad (2)$$

where D_{KL} is the Kullback-Leibler divergence and $Z(\beta, x)$ is a normalization function. These equations need to be satisfied simultaneously, given $p(x, y)$ and β . In [23] it is shown that alternating iterations of these equations converge - at

least locally - to a solution for any initial $p(t|x)$, similar to the Arimoto-Blahut algorithm in information theory [5]. In [3] it is shown that the set of achievable $p(x, y, t)$ distributions form a strictly convex set in the $(I(X; T), I(Y; T))$ plane, bounded by a smooth optimal function - *the information curve* - similar to the rate-distortion function in source coding. By increasing the value of β one can move smoothly along this curve from the trivial, $I(X; T) = I(Y; T) = 0$ solution at the origin, all the way to the most complex solution where T captures all the relevant information from X and $I(X; T) = H(X)$, $H(X)$ denoting the entropy of X . In addition, as β is increased, $I(Y; T)$ increases and T captures more information on Y . Due to the data-processing inequality, $I(Y; T) \leq I(X; Y)$, with equality only when T becomes an exact sufficient statistic for Y . The tradeoff inherent in Eq. (1) forces us to find a simple representation T of X , which preserves only those aspects of X which are informative, i.e. relevant, about Y .

It should be emphasized that despite superficial similarities, IB is *not* a hidden variable model. In such models, we assume that the joint distribution $p(x, y)$ can be factorized using an auxiliary random variable T , forming a Markovian relation $X - T - Y$. In IB, we make no generative assumption on the distribution, and the Markovian relation is $T - X - Y$. Namely, T is a generic compression of X , and the information-curve is characterized by the joint distribution $p(x, y)$ independently of any modeling assumptions.

An important observation is that the effective cardinality of an optimal T is not fixed and depends on β . When $\beta \leq 1$, even a trivial T of cardinality 1 will optimize Eq. (1), since we always have $I(Y; T) \leq I(X; T)$. On the other hand, as β increases, more emphasis is put on informativeness with respect to Y , and the cardinality of T will increase, although the cardinality of an optimal T need not exceed the cardinality of X , as proven in [10].

In order to optimize Eq. (1) we need to calculate the quantities $I(X; T)$ and $I(Y; T)$ for any chosen T and β . Since T is defined only via X , we need to know $p(x, y)$ in order to calculate these two quantities. In most applications, however, $p(x, y)$ is unknown. Instead, we assume that we have an i.i.d sample of m instances drawn according to $p(x, y)$, and we use this sample to create a maximum-likelihood estimate of the distribution using $\hat{p}(x, y)$, the empirical distribution of the sample. Following current practice, this empirical estimate is then plugged into the calculation of $I(X; T)$ and $I(Y; T)$ instead of the true joint distribution, and Eq. (1) is optimized using this plug-in estimate. In general, we use the $\hat{\cdot}$ symbol to denote quantities calculated using $\hat{p}(x, y)$ instead of $p(x, y)$. Thus, instead of calculating $I(X; T)$ and $I(Y; T)$ precisely, we rely on the empirical estimates $\hat{I}(X; T)$ and $\hat{I}(Y; T)$ respectively. In this work we investigate how much these empirical estimates can deviate from the true values when we optimize for T - in other words, whether this plug-in practice is justified. Note that the sample size m is often smaller than the number of bins $|\mathcal{X}||\mathcal{Y}|$, and thus $\hat{p}(x, y)$ can be a very poor approximation to $p(x, y)$. Nevertheless, this is precisely the regime we are interested in for many applications, text categorization to name one.

3 Finite Sample Analysis

We begin our analysis by focusing on the finite-sample behavior of the IB framework, and in particular on the relationship between $I(X; T)$ and $I(Y; T)$ that appear in Eq. (1) and their empirical estimates $\hat{I}(X; T)$ and $\hat{I}(Y; T)$.

Our first result shows that for any *fixed* T defined as a random mapping of X via $p(t|x)$, it is possible to determine the value of the objective function Eq. (1) within reasonable accuracy based on a random sample. The proof outline is provided in Sec. 6.1. The full proof can be found in [19].

Theorem 1. *Let T be a given probabilistic function of X into an arbitrary finite target space, determined by $p(t|x)$, and let $\mathcal{S}$ be a sample of size m drawn from the joint probability distribution $p(X, Y)$. For any confidence parameter $\delta \in (0, 1)$, it holds with a probability of at least $1 - \delta$ over the sample $\mathcal{S}$ that*

$$|I(X; T) - \hat{I}(X; T)| \leq \frac{(|T| \log(m) + \log(|T|)) \sqrt{\log(4/\delta)}}{\sqrt{2m}} + \frac{|T| - 1}{m},$$

and that

$$|I(Y; T) - \hat{I}(Y; T)| \leq \frac{(3|T| + 2) \log(m) \sqrt{\log(4/\delta)}}{\sqrt{2m}} + \frac{(|\mathcal{Y}| + 1)(|T| + 1) - 4}{m}.$$

Note that the theorem holds for any fixed T , not just ones which optimize Eq. (1). In particular, the theorem holds for any T found by an IB algorithm, even if T is not a globally optimal solution.

The theorem shows that estimating the objective function for a certain solution T is much easier than estimating $p(x, y)$. Indeed, the bound does not depend on $|\mathcal{X}|$, which might even be countably infinite. In addition, it depends on $|\mathcal{Y}|$ only as a second-order factor, since $|\mathcal{Y}|$ is multiplied by $1/m$ rather than by $1/\sqrt{m}$. The complexity of the bound is thus mainly controlled by $|T|$. By constraining $|T|$ to be small, or by setting β in Eq. (1) to be small enough so that the optimal T has low cardinality, a tight bound can be achieved.

Thm. 1 provides us with a bound on a certain pre-specified T , where the sample $\mathcal{S}$ is not part of the process of selecting T . The next theorem is a full generalization bound, determined by the sample when it is used as a training set by which T is selected.

For presenting the theorem compactly, we will use some extra notation. Let $x_1, \dots, x_{|\mathcal{X}|}$ be some fixed ordering of the elements of $\mathcal{X}$, and $y_1, \dots, y_{|\mathcal{Y}|}$ be an ordering of the elements of $\mathcal{Y}$. We use the shorthand $\mathbf{p}(T = t|x)$ to denote the vector $(p(t|x_1), \dots, p(t|x_{|\mathcal{X}|}))$. Similarly, we denote the vector $(\hat{H}(T|y_1), \dots, \hat{H}(T|y_{|\mathcal{Y}|}))$ by $\hat{\mathbf{H}}(T|y)$ where $\hat{H}(T|y_i)$ is the entropy of $\hat{p}(T|y_i)$. Furthermore, the vector $(H(T|x_1), \dots, H(T|x_{|\mathcal{X}|}))$ is denoted by $\mathbf{H}(T|x)$, where $H(T|x_i)$ is the entropy of $p(T|x_i)$. Note that $p(T|x_i)$ is known as it defines T , and thus does not need to be estimated empirically.

For any real-valued vector $\mathbf{a} = (a_1, \dots, a_n)$, we define the function $V(\mathbf{a})$ as follows:

$$V(\mathbf{a}) = \left\| \mathbf{a} - \frac{1}{n} \sum_{j=1}^n a_j \right\|^2 \triangleq \sum_{i=1}^n \left(a_i - \frac{1}{n} \sum_{j=1}^n a_j \right)^2. \quad (3)$$

Note that $\frac{1}{n}V(\mathbf{a})$ is simply the variance of the elements of $\mathbf{a}$. In addition, we define the real-valued function ϕ as follows:

$$\phi(x) = \begin{cases} 0 & x = 0 \\ x \log(1/x) & 0 < x \leq 1/e \\ 1/e & x > 1/e. \end{cases} \quad (4)$$

Note that ϕ is a continuous, monotonically increasing and concave function.

Theorem 2. *Let $\mathcal{S}$ be a sample of size m drawn from the joint probability distribution $p(X, Y)$. For any confidence parameter $\delta \in (0, 1)$, it holds with a probability of at least $1 - \delta$ over the sample $\mathcal{S}$ that for all T , $|I(X; T) - \hat{I}(X; T)|$ is upper bounded by*

$$\sqrt{\frac{C \log(|\mathcal{Y}|/\delta) \cdot V(\mathbf{H}(T|x))}{m}} + \sum_t \phi \left(\sqrt{\frac{C \log(|\mathcal{Y}|/\delta) \cdot V(\mathbf{p}(T = t|x))}{m}} \right), \quad (5)$$

and $|I(Y; T) - \hat{I}(Y; T)|$ is upper bounded by

$$\sqrt{\frac{C \log(|\mathcal{Y}|/\delta) \cdot V(\hat{\mathbf{H}}(T|y))}{m}} + 2 \sum_t \phi \left(\sqrt{\frac{C \log(|\mathcal{Y}|/\delta) \cdot V(\mathbf{p}(T = t|x))}{m}} \right), \quad (6)$$

where V and ϕ are defined in Eq. (3) and Eq. (4), and C is a small constant.

As in Thm. 1, this theorem holds for all T , not just those optimizing Eq. (1). Also, the bound enjoys the advantage of not being uniform over a hypothesis class of possible T 's, but rather depending directly on the T of interest. This is achieved by avoiding standard uniform complexity tools (see the proof for further details).

Intuitively, these bounds tell us that the ‘smoother’ T is with respect to X , the tighter the bound. To see this, assume that for any fixed $t \in \mathcal{T}$, $p(t|x)$ is more or less the same for any choice of x . By definition, this means that $V(\mathbf{p}(T = t|x))$ is close to zero. In a similar manner, if $H(T|x)$ is more or less the same for any x , then $V(\mathbf{H}(T|x))$ is close to zero, and so is $V(\hat{\mathbf{H}}(T|y))$ if $\hat{H}(T|y)$ is more or less the same for any y . In the extreme case, if T is independent of X , then $p(t|x) = p(t)$, $H(T|x) = H(T)$ and $\hat{H}(T|y) = \hat{H}(T)$ for any choice of x, y , and the generalization bound becomes zero. This is not too surprising, since in this case $I(X; T) = I(\hat{X}; T) = 0$ and $I(Y; T) = \hat{I}(Y; T) = 0$ regardless of $p(x, y)$ or its empirical estimate $\hat{p}(x, y)$.

This theorem thus suggests that generalization becomes better as T becomes less statistically dependent on X , and so provides a more compressed probabilistic representation of X . This is exactly in line with empirical findings [20], and with the intuition that ‘simpler’ models should lead to better generalization.

A looser but simpler bound on Thm. 2 can be achieved by fixing the cardinality of T , and analyzing the bound with worst-case assumptions on the statistical dependency between X and T . The proof, which is rather technical, is omitted in this version and may be found in [19].

Theorem 3. *Under the conditions and notation of Thm. 2, we have that with a probability of at least $1 - \delta$, for all T ,*

$$|I(X; T) - \hat{I}(X; T)| \leq \frac{\frac{1}{2}\sqrt{C \log(|\mathcal{Y}|/\delta)}(\sqrt{|\mathcal{T}||\mathcal{X}|} \log(m) + |\mathcal{X}|^{\frac{1}{2}} \log(|\mathcal{T}|)) + \frac{1}{e}|\mathcal{T}|}{\sqrt{m}}$$

and

$$|I(Y; T) - \hat{I}(Y; T)| \leq \frac{\sqrt{C \log(|\mathcal{Y}|/\delta)}\left(\sqrt{|\mathcal{T}||\mathcal{X}|} \log(m) + \frac{1}{2}|\mathcal{Y}|^{\frac{1}{2}} \log(|\mathcal{T}|)\right) + \frac{2}{e}|\mathcal{T}|}{\sqrt{m}},$$

where C is the same constant as in Thm. 1.

Even with this much looser bound, if $|\mathcal{Y}|$ is large and $|\mathcal{T}| \ll |\mathcal{Y}|$ the bound can be quite tight, even with sample sizes which are in general insufficient to reasonably estimate the joint distribution $p(x, y)$. One relevant setting is in unsupervised learning, when Y models the feature space.

In this section, we have shown that the quantities that make up the IB objective function can be estimated reliably from a sample of a reasonable size, depending on the characteristics of T . In the next section we investigate the motivation for using these quantities in the objective function in the first place.

4 A Learning Theoretic Perspective

The IB framework optimizes a trade-off between $I(X; T)$ and $I(Y; T)$. In this section we provide a preliminary discussion of the learning theoretic properties of this tradeoff, investigating when mutual information provides reasonable measures for both learning complexity and accuracy.

In an unsupervised setting, such as clustering, it is rather easy to see how $I(X; T)$ and $I(Y; T)$ control the complexity and granularity of the clustering by trading between homogeneity and resolution of the clusters; this has been discussed previously in the literature (such as [24], [3]). Therefore, we will focus here mainly on the use of this framework in supervised learning, where the objectives are more well defined.

Most supervised learning algorithms are based on a tradeoff between two quantities: a risk term, measuring the performance of a hypothesis on the sample data, and a regularization term, which penalizes complex hypotheses and so

ensures reasonable generalization to unseen data. In the following we argue that under relevant settings it is reasonable to consider $I(Y;T)$ as a measure of risk and $I(X;T)$ as a regularization term that controls generalization.

4.1 $I(Y;T)$ as a Measure of Performance

In this section we investigate the plausibility of $I(Y;T)$ as a measure of performance or risk in a supervised learning setting. We show that in those supervised learning settings where IB was demonstrated to be highly effective, such as document categorization [22], there is a strong connection between the classification error and the mutual information $I(Y;T)$, especially when the categories are uniformly spread. The discussion here is a first step towards a full analysis of the IB classification performance in a more general setting, which we leave for future work.

In a typical document classification task we model X as a random variable over the set of possible words, and Y as a random variable over the set of document categories or classes. Each document is treated as an i.i.d. sample of words drawn from $p(x|y)$, in accordance with the bag of words representation, where y is the class of the document. Unlike the simple supervised learning settings, where each example is described as a single data point, in this case each example (document) to be labeled is described by a sample of points (words) of variable size (usually large) and we seek the most probable class of the whole sample (document) *collectively*.

IB is used in this setting to find T , a compressed representation of the words in a document, which is as informative as possible on the categories Y . The bottleneck equations Eq. (2) provide for each class y its conditional distribution on T , via $\hat{p}(t|y) = \sum_x p(t|x)\hat{p}(x|y)$. When a new document $D = \{x_1, \dots, x_n\}$ of size n is to be classified, the empirical distribution of T given D is $\tilde{p}(t) = \sum_{i=1}^n p(t|x_i)\hat{p}(x_i)$. Assuming that the document is sampled according to $p(t|y)$ for some class y , the most probable class y^* can be selected using the maximum likelihood principle, namely $y^* = \operatorname{argmin}_y D_{\text{KL}}[\tilde{p}(t) \parallel \hat{p}(t|y)]$.

We now show that $\hat{I}(Y;T)$ is indeed a reasonable objective function whenever we wish to collectively label an entire set of sampled instances.

Assume that the true class for document D is y_1 , with its word distribution sampled via $p(t|y_1)$. The probability α_n of misclassifying this sample as y_2 for some $y_2 \neq y_1$ via the likelihood test decreases exponentially with the sample size n . The rate of exponential decrease is larger if the two distributions $p(t|y_1), p(t|y_2)$ are more distinct. Formally, by Stein's lemma [5], if $\hat{p}(t|y_1) = p(t|y_1)$ and $\hat{p}(t|y_2) = p(t|y_2)$, then

$$\lim_{n \rightarrow \infty} \frac{1}{n} \log(\alpha_n) = D_{\text{KL}}[p(t|y_2) \parallel p(t|y_1)]. \quad (7)$$

When $\hat{p}(t|y_1)$ and $\hat{p}(t|y_2)$ deviate from the true conditional distributions, Stein's Lemma still holds up to an additive constant which depends on the amount of deviation, and the exponent is still controlled mainly by $D_{\text{KL}}[p(t|y_2) \parallel p(t|y_1)]$. In the following we will assume for simplicity that Eq. (7) holds exactly.

The overall probability of misclassifying a document when there are more than two possible classes is thus upper bounded by

$$\sum_{y \neq y_1} \exp(-nD_{\text{KL}}[p(t|y)||p(t|y_1)]). \quad (8)$$

On the other hand, by the definition of mutual information and the convexity of the Kullback-Leibler divergence we have that

$$\begin{aligned} I(Y; T) &= \mathbb{E}_y D_{\text{KL}}[p(t|y)||p(t)] = \mathbb{E}_y D_{\text{KL}}[p(t|y)||\mathbb{E}_{y'} p(t|y')] \\ &\leq \mathbb{E}_{y, y'} D_{\text{KL}}[p(t|y)||p(t|y')], \end{aligned} \quad (9)$$

Hence $-nI(Y; T)$ is an upper bound on the expected value of the exponent in Eq. (7), assuming that y_1 and y_2 are picked according to $p(y)$. The relationship between Eq. (9) on the one hand, and Eq. (7), Eq. (8) on the other hand, is not direct. Nonetheless, these equations indicate that if the examples to classify are represented by a large sample, as in the document classification setting, higher values of $I(Y; T)$ should correspond to a reduced probability of misclassification. For example, if $D_{\text{KL}}[p(t|y)||p(t|y_1)]$ is equal for every $y \neq y_1$, we have that Eq. (8) is upper bounded by

$$(n - 1) \exp(-nI(Y; T)/(|\mathcal{Y}| - 1)),$$

in which case the probability of misclassification is exponentially dominated by $I(Y; T)$. This is the case when categories are uniformly spread, which happens for many applications incidently or by design. In this case, when the bottleneck variable T captures just a fraction $\alpha = I(Y; T)/I(X; Y)$ of the relevant information, the test (document) size should increase only by a factor $1/\alpha$ in order to achieve a similar bound on the classification error.

4.2 $I(X; T)$ as a Regularization Term

In this subsection we discuss the role of $I(X; T)$, the compression term in IB, as a regularizer when maximizing $I(Y; T)$. Note that without regularization, $I(Y; T)$ can be maximized by setting $T = X$. However, $p(x|y)$ cannot be estimated efficiently from a sample of a reasonable size; therefore the formal solution $T = X$ cannot be used to perform reliable classification. Moreover, in the context of unsupervised learning, setting $T = X$ is generally a meaningless operation, corresponding to singleton clusters.

The bottleneck variable T must therefore be restricted to allow reasonable generalization in a supervised setting and to generate a reasonable model in an unsupervised setting. In the IB framework $I(X; T)$ can be viewed as a penalty term that restricts the complexity of T . A more formal justification for this is given in the following theorem, which is derived from Thm. 2. Since the proof is quite technical, it is omitted in this version and may be found in [19].

Theorem 4. *For any probability distribution $p(x, y)$, with a probability of at least $1 - \delta$ over the draw of the sample of size m from $p(x, y)$, we have that for all T ,*

$$|I(Y; T) - \hat{I}(Y; T)| \leq \sqrt{\frac{C \log(|\mathcal{Y}|/\delta)}{m}} \left(C_1 \log(m) \sqrt{|T| I(X; T)} \right. \\ \left. + C_2 |T|^{3/4} (I(X; T))^{1/4} + C_3 \hat{I}(X; T) \right),$$

where C is the same constant as in Thm. 1, and C_1, C_2, C_3 depend only on $p(x)$ and $p(y)$.

This bound is controlled by $I(X; T)$ and $\hat{I}(X; T)$, which are closely related as Thm. 3 shows. This is not a fully empirical bound, as it depends on the unknown quantity $I(X; T)$ and the marginal distributions of X, Y . The bound does however illustrate the relationship between the generalization error, as embodied in the difference between $I(Y; T)$ and $\hat{I}(Y; T)$, and the mutual information $I(X; T)$. This provides motivation for the use of $I(X; T)$ as a regularization term, beyond its obvious description length interpretation or coding interpretation.

5 Relationship with Sufficient Statistics

A fundamental issue in statistics, pattern recognition, and machine learning is the notion of relevance. Finding the relevant components of data is implicitly behind the problems of efficient data representation, feature selection and dimension reduction for supervised learning, and is the essence of most unsupervised learning problems. One of the earliest and more principled approaches to relevance was the concept of *sufficient statistics* for parametric distributions, introduced by Fisher [7] as function(s) of a sample that capture all the information about the parameter(s). A *sufficient statistic* is defined as follows:

Definition 1 (Sufficient Statistic). *Let Y be a parameter indexing a family of probability distributions. Let X be random variable drawn from a probability distribution determined by Y . Let T be a deterministic function of X . T is sufficient for Y if*

$$\forall x \in \mathcal{X}, t \in \mathcal{T}, y \in \mathcal{Y} \quad p(x|t, y) = p(x|t).$$

Throughout this section we assume that it suffices that the equality holds almost everywhere with respect to the probability of y and x .

In words, the sufficiency of T means that given the value of T , the distribution of X does not depend on the value of Y .

In the parametric statistics setting, Y is a random variable that parameterizes a family of probability distributions, and X is a data point drawn from $p(x|y)$ where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. For example, the family of probability distributions may be the set of Bernoulli distributions with success probability p determined

by y , with $\mathcal{Y} \subseteq [0, 1]$ and some prior distribution $p(y)$. In this case, for a given y , $p(X = 1|y) = y$, and $p(X = 0|y) = 1 - y$.

Y and X may be high dimensional. For instance, Y may determine the mean and the variance of a normal distribution, or fully parameterize a multinomial distribution. X may be a high dimensional data point. For any family of probability distributions, we can consider a sample of m i.i.d data points, all drawn from the same distribution determined by a single draw of Y . In the context of sufficient statistics, this is just a special case of a high dimensional X which is drawn from the cross-product of m identical probability distributions determined by the value of Y .

Just as X and Y may be high dimensional, so can T map X to a multidimensional space. If X denotes an i.i.d sample, the number of dimensions in T may depend on the size of the sample m . Specifically, $T = X$ is always sufficient for Y . To avoid trivial sufficient statistics such as this, Lehmann and Scheffé [12] introduced the concept of a minimal sufficient statistic, which denotes the coarsest sufficient partition of X , as follows:

Definition 2 (Minimal Sufficient Statistic). *A sufficient statistic S is minimal if and only if for any sufficient statistic T , there exists a deterministic function f such that $S = f(T)$ almost everywhere w.r.t X .*

For instance, for an i.i.d sample of size m of the Bernoulli distribution in the example above, $T = X$ is trivially a sufficient statistic, but the one-dimensional $T = \frac{1}{m} \sum_i x_i$ where $x = (x_1, \dots, x_m)$ is also sufficient. It can be shown that the latter T (and any one-to-one function of it) is a minimal sufficient statistic.

By the Pitman-Koopman-Darmois theorem [17], sufficient statistics whose dimension does not depend on the sample size exist only for families of exponential form. This makes the original concept of sufficiency rather restricted.

Kullback and Leibler [11] related sufficiency to Shannon's information theory, showing that sufficiency is equivalent to preserving mutual information on the parameter, while minimal sufficient statistics minimize the mutual information with the sample due to the data-processing inequality [5].

The IB framework allows us to naturally extend this concept of relevance to any joint distribution of X and Y , not necessarily ones of exponential form, in a constructive computational manner. In this framework, built on Kullback's information theoretic characterization of sufficiency [11], one can find compact representations T of a sample X that maximize mutual information about the parameter variable Y , corresponding to sufficiency for Y , and minimize $I(X; T)$, corresponding to the minimality of the statistic. However, unlike the original concepts of sufficient statistic and minimal sufficient statistic, the IB framework provides a soft tradeoff between these two objectives.

It can easily be seen that as β grows to infinity, if T is not restricted then $I(Y; T)$ converges to $I(X; Y)$ and T converges to a minimal sufficient statistic. The following theorem formalizes this insight. Similar formulations of this theorem can be gleaned from [11] and [5]. The full proof is presented for completeness in [19].

Theorem 5. *Let X be a sample drawn according to a distribution determined by the random variable Y . The set of solutions to*

$$\min_T I(X; T) \quad \text{s.t.} \quad I(Y; T) = \max_{T'} I(Y; T')$$

is exactly the set of minimal sufficient statistics for Y based on the sample X .

The IB framework thus provides a natural generalization of the concept of a sufficient statistic, where by setting β to lower values, different degrees of approximate minimal sufficient statistics can be found, characterized by the fraction of mutual information they maintain on the Y . Furthermore, such approximate minimal sufficient statistics exist for any joint distribution $p(X, Y)$ in a continuous hierarchy that is fully captured by the set of optimal IB solutions for all values of β . These solutions lie on the information curve of the distribution.

6 Proofs

6.1 Proof of Thm. 1

Let $\mathcal{S}$ be a sample of size m , and let T be a probabilistic function of X into an arbitrary finite target space, defined by $p(t|x)$ for all $x \in \mathcal{X}$ and $t \in \mathcal{T}$.

To prove the theorem, we bound the deviations of the information estimations from their expectation: $|\hat{I}(X; T) - \mathbb{E}[\hat{I}(X; T)]|$ and $|\hat{I}(Y; T) - \mathbb{E}[\hat{I}(Y; T)]|$, and then use a bound on the expected bias of entropy estimation.

To bound the deviation of the information estimates, we use McDiarmid's inequality [14], in a manner similar to [1]. For this we must bound the change in value of each of the entropy estimates when a single instance in $\mathcal{S}$ is arbitrarily changed. A useful and easily proven inequality in that regard is the following: for any natural m and for any $a \in [0, 1 - 1/m]$ and $\Delta \leq 1/m$,

$$\left| (a + \Delta) \log(a + \Delta) - a \log(a) \right| \leq \frac{\log(m)}{m}. \quad (10)$$

With this inequality, a careful application of McDiarmid's inequality leads to the following lemma. The proof of the lemma can be found in [19].

Lemma 1. *For any $\delta_1 > 0$, with probability of at least $1 - \delta_1$ over the sample, we have that*

$$|\hat{I}(X; T) - \mathbb{E}[\hat{I}(X; T)]| \leq \frac{(|\mathcal{T}| \log(m) + \log(|\mathcal{T}|)) \sqrt{\log(2/\delta_1)}}{\sqrt{2m}}. \quad (11)$$

Similarly, with a probability of at least $1 - \delta_2$,

$$|\hat{I}(Y; T) - \mathbb{E}[\hat{I}(Y; T)]| \leq \frac{(3|\mathcal{T}| + 2) \log(m) \sqrt{\log(2/\delta_2)}}{\sqrt{2m}}. \quad (12)$$

Lemma 1 provides bounds on the deviation of the $\hat{I}(X;T)$, $\hat{I}(Y;T)$ from their expected values. In order to relate these to the true values of the mutual information $I(X;T)$ and $I(Y;T)$, we use the following bias bound from [15].

Lemma 2 (Paninski, 2003). *For a random variable X , with the plug-in estimate $\hat{H}(\cdot)$ on its entropy, based on an i.i.d sample of size m , we have that*

$$|\mathbb{E}[\hat{H}(X) - H(X)]| \leq \log \left(1 + \frac{|\mathcal{X}| - 1}{m} \right) \leq \frac{|\mathcal{X}| - 1}{m}.$$

From Lemma 2, we get that the quantities $|\mathbb{E}[H(T) - H(T)]|$, $|\mathbb{E}[H(Y) - H(Y)]|$, and $|\mathbb{E}[H(Y, T) - H(Y, T)]|$ are upper bounded by $(|\mathcal{T}| - 1)/m$, $(|\mathcal{Y}| - 1)/m$ and $(|\mathcal{Y}||\mathcal{T}| - 1)/m$ respectively. Combining these with Eq. (11) and Eq. (12), and setting $\delta_1 = \delta_2 = \delta/2$, we get the bounds in Thm. 1.

6.2 Proof of Thm. 2

The idea of the proof is as follows. We bound the quantities $|I(X;T) - \hat{I}(X;T)|$ and $|I(Y;T) - \hat{I}(Y;T)|$ with deterministic bounds that depend on the empirical distribution and on the true underlying distribution. These bounds are factorized, in the sense that quantities that depend on the empirical sample are separated from quantities that depend on the characteristics of T . Quantities of the first type can be bounded by concentration of measure theorems, while quantities of the second type can be left dependent on the T we choose.

The deterministic bounds are summarized in the following lemma. The proof of this lemma is purely technical, and may be found in [19].

Lemma 3. *The following two inequalities hold:*

$$\begin{aligned} |I(X;T) - \hat{I}(X;T)| &\leq \sum_t \|\mathbf{p}(x) - \hat{\mathbf{p}}(x)\| \cdot \phi \left(\sqrt{V(\mathbf{p}(T=t|x))} \right) \quad (13) \\ &\quad + \|\mathbf{p}(x) - \hat{\mathbf{p}}(x)\| \cdot \sqrt{V(\mathbf{H}(T|x))}, \end{aligned}$$

$$\begin{aligned} |I(Y;T) - \hat{I}(Y;T)| &\leq \sum_t \|\mathbf{p}(x) - \hat{\mathbf{p}}(x)\| \cdot \phi \left(\sqrt{V(\mathbf{p}(T=t|x))} \right) \quad (14) \\ &\quad + \sum_y p(y) \sum_t \phi \left(\|\hat{\mathbf{p}}(x|y) - \mathbf{p}(x|y)\| \cdot \sqrt{V(\mathbf{p}(T=t|x))} \right) \\ &\quad + \|\mathbf{p}(y) - \hat{\mathbf{p}}(y)\| \cdot \sqrt{V(\hat{\mathbf{H}}(T|y))}. \end{aligned}$$

In order to transform the bounds in Eq. (13) and Eq. (14) to bounds that do not depend on $p(x)$, we can use concentration of measure arguments on L_2 norms of random vectors, such as the following one based on an argument in section 4.1 of [6]: Let ρ be a distribution vector of arbitrary (possible countably infinite)

cardinality, and let $\hat{\rho}$ be an empirical estimation of ρ based on a sample of size m . Then with a probability of at least $1 - \delta$ over the samples,

$$\|\rho - \hat{\rho}\|_2 \leq \frac{2 + \sqrt{2 \log(1/\delta)}}{\sqrt{m}}. \quad (15)$$

We apply this concentration bound to $\|\mathbf{p}(x) - \hat{\mathbf{p}}(x)\|$, $\|\mathbf{p}(y) - \hat{\mathbf{p}}(y)\|$, and to $\|\hat{\mathbf{p}}(x|y) - \mathbf{p}(x|y)\|$ for any y in Eq. (13) and Eq. (14). To make sure the bounds hold simultaneously over these $|\mathcal{Y}| + 2$ quantities, we replace δ in Eq. (15) by $\delta/(|\mathcal{Y}| + 2)$. Note that the union bound is taken with respect to the marginal distributions of $\hat{\mathbf{p}}(x)$, $\hat{\mathbf{p}}(y)$ and $\hat{\mathbf{p}}(x|y)$, which do not depend on the T chosen. Thus, the following bounds hold with a probability of $1 - \delta$, for all T :

$$\begin{aligned} |I(X; T) - \hat{I}(X; T)| &\leq (2 + \sqrt{2 \log((|\mathcal{Y}| + 2)/\delta)}) \sqrt{\frac{V(\mathbf{H}(T|x))}{m}} \\ &\quad + \sum_t \phi \left((2 + \sqrt{2 \log((|\mathcal{Y}| + 2)/\delta)}) \sqrt{\frac{V(\mathbf{p}(T = t|x))}{m}} \right), \\ |I(Y; T) - \hat{I}(Y; T)| &\leq (2 + \sqrt{2 \log((|\mathcal{Y}| + 2)/\delta)}) \sqrt{\frac{V(\hat{\mathbf{H}}(T|y))}{m}} \\ &\quad + 2 \sum_t \phi \left((2 + \sqrt{2 \log((|\mathcal{Y}| + 2)/\delta)}) \sqrt{\frac{V(\mathbf{p}(T = t|x))}{m}} \right). \end{aligned}$$

To get the bounds in Thm. 2, we note that

$$2 + \sqrt{2 \log((|\mathcal{Y}| + 2)/\delta)} \leq \sqrt{C \log(|\mathcal{Y}|/\delta)}$$

where C is a small constant.

It is interesting to note that these bounds still hold in certain cases even if $\mathcal{X}$ is infinite. Specifically, suppose that for all $t \in \mathcal{T}$, $p(t|x)$ is some constant c_t for all but a finite number of elements of $\mathcal{X}$. If the definition of $V(\cdot)$ is replaced with $V(\mathbf{p}(T = t|x)) = \sum_x (p(T = t|x) - c_t)^2$, Then $V(\mathbf{p}(T = t|x))$ is finite and the proof above remains valid. Therefore, under these restrictive assumptions the bound is valid and meaningful even though $\mathcal{X}$ is infinite.

7 Discussion

In this paper we analyzed the information bottleneck framework from a learning theoretic perspective. This framework has been used successfully for finding efficient relevant data representations in various applications, but this is its first rigorous learning theoretic analysis. Despite the fact that the information bottleneck is all about manipulating the joint input-output distribution, we show that it can generalize quite well based on plug-in empirical estimates, even with

sample sizes much smaller than needed for reliable estimation of the joint distribution. In fact, it is exactly the reliance on the joint distribution that allows us to derive non-uniform and adaptive bounds.

Moreover, these bounds allow us to view the information bottleneck framework in the more familiar learning theoretic setting of a performance-complexity tradeoff. In particular, we provided a preliminary analysis of the role of mutual information as both a complexity regularization term and as a bound on the classification error for common supervised applications, such as document classification. This is the first step in providing a theoretical justification for many applications of interest, including a characterization of the learning scenarios for which this method is best suited. Finally, we showed how this framework extends the classical statistical concept of minimal sufficient statistics.

References

- [1] Antos, A., Kontoyiannis, I.: Convergence properties of functional estimates for discrete distributions. *Random Structures and Algorithms* 19(3–4), 163–193 (2001)
- [2] Baker, L.D., McCallum, A.K.: Distributional clustering of words for text classification. In: *Proceedings of SIGIR 1998*, pp. 96–103 (1998)
- [3] Gilad-Bachrach, R., Navot, A., Tishby, N.: An Information Theoretic Tradeoff between Complexity and Accuracy. In: *Proceedings of COLT 2003*, pp. 595–609 (2003)
- [4] Chechik, G., Globerson, A., Tishby, N., Weiss, Y.: Information Bottleneck for Gaussian Variables. *Journal of Machine Learning Research* 6, 165–188 (2005)
- [5] Cover, T.M., Thomas, J.A.: *Elements of Information Theory*. Wiley, Chichester (1991)
- [6] Cristianini, N., Shawe-Taylor, J.: *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge (2004)
- [7] Fisher, R.A.: On the Mathematical Foundation of Theoretical Statistics. *Philos. Trans. Roy. Soc. London Sec. A* 222, 309–368 (1922)
- [8] Friedman, N., Mosenzon, O., Slonim, N., Tishby, N.: Multivariate Information Bottleneck. In: *Proceedings of UAI 2001*, pp. 152–161 (2001)
- [9] Nikravesh, M., Guyon, I., Gunn, S., Zadeh, L.A. (eds.): *Feature Extraction: Foundations and Applications*. Springer, Heidelberg (2006)
- [10] Harremoës, P., Tishby, N.: The Information Bottleneck Revisited or How to Choose a Good Distortion Measure. In: *Proceedings of the IEEE Int. Symp. on Information Theory*, pp. 566–571 (2007)
- [11] Kullback, S., Leibler, R.A.: On Information and Sufficiency. *Ann. Math. Stat.* 22, 79–86 (1951)
- [12] Lehmann, E.L., Scheffé, H.: Completeness, Similar Regions and Unbiased Estimation. *Sankhya* 10, 305–340 (1950)
- [13] Lehmann, E.L.: *Testing Statistical Hypotheses*. Wiley, New-York (1959)
- [14] McDiarmid, C.: On the Method of Bounded Differences. In: Siemons, J. (ed.) *Surveys in Combinatorics*. London Mathematical Society Lecture Note Series, vol. 141, pp. 148–188. Cambridge University Press, Cambridge (1989)
- [15] Paninski, L.: Estimation of Entropy and Mutual Information. *Neural Computation* 15(6), 1191–1253 (2003)
- [16] Pereira, F.C., Tishby, N., Lee, L.: Distributional Clustering of English Words. In: *Meeting of the Association for Computational Linguistics*, pp. 183–190 (1993)

- [17] Koopman, B.: On Distributions Admitting a Sufficient Statistic. *Trans. Amer. math. Soc.* 39, 399–409 (1936)
- [18] Rockafellar, R.T.: *Convex Analysis*. Princeton University Press, Princeton (1970)
- [19] Shamir, O., Sabato, S., Tishby, N.: Learning and Generalization with the Information Bottleneck,
www.cs.huji.ac.il/~ohads03/ShamirSabatoTishbyALT2008_full.pdf
- [20] Slonim, N.: *The Information Bottleneck: Theory and Applications*. PhD thesis. Hebrew University, Jerusalem (2003)
- [21] Slonim, N., Singh, G., Atwal, S., Tkacik, G., Bialek, W.: Information-based Clustering. *Proc. Natl. Acad. Sci. U.S.A* (December 2005)
- [22] Slonim, N., Tishby, N.: The Power of Word Clusters for Text Classification. In: 23rd European Colloquium on Information Retrieval Research (2001)
- [23] Tishby, N., Pereira, F.C., Bialek, W.: The Information Bottleneck Method. In: The 37th Allerton Conference on Communication, Control, and Computing (1999)
- [24] Tishby, N., Slonim, N.: Data clustering by Markovian relaxation and the information bottleneck method. In: *Proceedings of NIPS 2000*, pp. 640–646 (2000)

Growth Optimal Investment with Transaction Costs^{*}

László Györfi and István Vajda

Department of Computer Science and Information Theory,
Budapest University of Technology and Economics,
Magyar Tudósok Körútja 2., Budapest, Hungary, H-1117
`{gyorfi,vajda}@szit.bme.hu`

Abstract. Discrete time infinite horizon growth optimal investment in stock markets with transactions costs is considered. The stock processes are modelled by homogeneous Markov processes. Assuming that the distribution of the market process is known, we show two recursive investment strategies such that, in the long run, the growth rate on trajectories (in "liminf" sense) is greater than or equal to the growth rate of any other investment strategy with probability 1.

1 Introduction

The purpose of this paper is to investigate sequential investment strategies for financial markets such that the strategies are allowed to use information collected from the past of the market and determine, at the beginning of a trading period, a portfolio, that is, a way to distribute their current capital among the available assets. The goal of the investor is to maximize his wealth on the long run. If there is no transaction cost and the price relatives form a stationary and ergodic process the best strategy (called log-optimum strategy) can be constructed in full knowledge of the distribution of the entire process, see Algoet and Cover [2].

Papers dealing with growth optimal investment with transaction costs in discrete time setting are seldom. Cover and Iyengar [13] formulated the problem of horse race markets, where in every market period one of the assets has positive pay off and all the others pay nothing. Their model included proportional transaction costs and they used a long run expected average reward criterion. There are results for more general markets as well. Iyengar [12] investigated growth optimal investment with several assets assuming independent and identically distributed (i.i.d.) sequence of asset returns. Bobryk and Stettner [4] considered the case of portfolio selection with consumption, when there are two assets, a bank account and a stock. Furthermore, long run expected discounted reward and i.i.d asset returns were assumed. In the case of discrete time, the most far reaching study was Schäfer [16] who considered the maximization of the long run expected growth rate with several assets and proportional transaction costs,

^{*} This research was supported by the Computer and Automation Research Institute of the Hungarian Academy of Sciences.

when the asset returns follow a stationary Markov process. In contrast to the previous literature we assume only that the stock processes are modelled by homogeneous (there is no requirement regarding the initial distribution) Markov processes. We extend the usual framework of analyzing expected growth rate by showing two recursive investment strategies such that, in the long run, the growth rate on trajectories is greater than or equal to the growth rate of any other investment strategy with probability 1.

The rest of the paper is organized as follows. In Section 2 we introduce the market model and describe the modelling of transaction costs. In Section 3 we formulate the underlying Markov control problem, and Section 4 defines optimal portfolio selection strategies. Following the Summary in Section 5, the proofs are given in Section 6.

2 Mathematical Setup: Investment with Transaction Cost

Consider a market consisting of d assets. The evolution of the market in time is represented by a sequence of market vectors $\mathbf{s}_1, \mathbf{s}_2, \dots \in \mathbb{R}_+^d$, where $\mathbf{s}_i = (s_i^{(1)}, \dots, s_i^{(d)})$ such that the j -th component $s_i^{(j)}$ of $\mathbf{s}_i$ denotes the price of the j -th asset at the end of the i -th trading period. ($s_0^{(j)} = 1$.)

In order to apply the usual prediction techniques for time series analysis one has to transform the sequence $\{\mathbf{s}_i\}$ into a more or less stationary sequence of return vectors $\{\mathbf{x}_i\}$ as follows: $\mathbf{x}_i = (x_i^{(1)}, \dots, x_i^{(d)})$ such that $x_i^{(j)} = \frac{s_i^{(j)}}{s_{i-1}^{(j)}}$. Thus,

the j -th component $x_i^{(j)}$ of the return vector $\mathbf{x}_i$ denotes the amount obtained after investing a unit capital in the j -th asset on the i -th trading period.

The investor is allowed to diversify his capital at the beginning of each trading period according to a portfolio vector $\mathbf{b} = (b^{(1)}, \dots, b^{(d)})^T$. The j -th component $b^{(j)}$ of $\mathbf{b}$ denotes the proportion of the investor's capital invested in asset j . Throughout the paper we assume that the portfolio vector $\mathbf{b}$ has nonnegative components with $\sum_{j=1}^d b^{(j)} = 1$. The fact that $\sum_{j=1}^d b^{(j)} = 1$ means that the investment strategy is self financing and consumption of capital is excluded. The non-negativity of the components of $\mathbf{b}$ means that short selling and buying stocks on margin are not permitted. To make the analysis feasible, some simplifying assumptions are used that need to be taken into account. We assume that assets are arbitrarily divisible and all assets are available in unbounded quantities at the current price at any given trading period. We also assume that the behavior of the market is not affected by the actions of the investor using the strategies under investigation.

For $j \leq i$ we abbreviate by $\mathbf{x}_j^i$ the array of return vectors $(\mathbf{x}_j, \dots, \mathbf{x}_i)$. Denote by Δ_d the simplex of all vectors $\mathbf{b} \in \mathbb{R}_+^d$ with nonnegative components summing up to one. An *investment strategy* is a sequence $\mathbf{B}$ of functions

$$\mathbf{b}_i : (\mathbb{R}_+^d)^{i-1} \rightarrow \Delta_d, \quad i = 1, 2, \dots$$

so that $\mathbf{b}_i(\mathbf{x}_1^{i-1})$ denotes the portfolio vector chosen by the investor on the i -th trading period, upon observing the past behavior of the market. We write $\mathbf{b}(\mathbf{x}_1^{i-1}) = \mathbf{b}_i(\mathbf{x}_1^{i-1})$ to ease the notation.

Let S_n denote the gross wealth at the end of trading period n , $n = 0, 1, 2, \dots$, where without loss of generality let the investor's initial capital S_0 be 1 dollar, while N_n stands for the net wealth at the end of trading period n . Using the above notations, for the trading period n , the net wealth N_{n-1} can be invested according to the portfolio $\mathbf{b}_n$, therefore the gross wealth S_n at the end of trading period n is

$$S_n = N_{n-1} \sum_{j=1}^d b_n^{(j)} x_n^{(j)} = N_{n-1} \langle \mathbf{b}_n, \mathbf{x}_n \rangle,$$

where $\langle \cdot, \cdot \rangle$ denotes inner product.

At the beginning of a new market day $n+1$, the investor sets up his new portfolio, i.e. buys/sells stocks according to the actual portfolio vector $\mathbf{b}_{n+1}$. During this rearrangement, he has to pay transaction cost, therefore at the beginning of a new market day $n+1$ the net wealth N_n in the portfolio $\mathbf{b}_{n+1}$ is less than S_n . The rate of proportional transaction costs (commission factors) levied on one asset are denoted by $0 < c_s < 1$ and $0 < c_p < 1$, i.e., the sale of 1 dollar worth of asset i nets only $1 - c_s$ dollars, and similarly we take into account the purchase of an asset such that the purchase of 1 dollar's worth of asset i costs an extra c_p dollars. We consider the special case when the rate of costs are constant over the assets. Let's calculate the transaction cost to be paid when select the portfolio $\mathbf{b}_{n+1}$. Before rearranging the capitals, at the j -th asset there are $b_n^{(j)} x_n^{(j)} N_{n-1}$ dollars, while after rearranging we need $b_{n+1}^{(j)} N_n$ dollars. If $b_n^{(j)} x_n^{(j)} N_{n-1} \geq b_{n+1}^{(j)} N_n$ then we have to sell and $c_s \left(b_n^{(j)} x_n^{(j)} N_{n-1} - b_{n+1}^{(j)} N_n \right)$ is the transaction cost at the j -th asset is, otherwise we have to buy and $c_p \left(b_{n+1}^{(j)} N_n - b_n^{(j)} x_n^{(j)} N_{n-1} \right)$ is the transaction cost at the j -th asset.

Let x^+ denote the positive part of x . Thus, the gross wealth S_n decomposes to the sum of the net wealth and cost the following - self-financing - way $S_n = N_n + c_s \sum_{j=1}^d \left(b_n^{(j)} x_n^{(j)} N_{n-1} - b_{n+1}^{(j)} N_n \right)^+ + c_p \sum_{j=1}^d \left(b_{n+1}^{(j)} N_n - b_n^{(j)} x_n^{(j)} N_{n-1} \right)^+$.

Dividing both sides by S_n and introducing the ratio $w_n = \frac{N_n}{S_n}$, $0 < w_n < 1$, we get

$$1 = w_n + c_s \sum_{j=1}^d \left(\frac{b_n^{(j)} x_n^{(j)}}{\langle \mathbf{b}_n, \mathbf{x}_n \rangle} - b_{n+1}^{(j)} w_n \right)^+ + c_p \sum_{j=1}^d \left(b_{n+1}^{(j)} w_n - \frac{b_n^{(j)} x_n^{(j)}}{\langle \mathbf{b}_n, \mathbf{x}_n \rangle} \right)^+. \quad (1)$$

Remark 1. Equation (1) is used in the sequel. Examining this cost equation, it turns out, that for arbitrary portfolio vectors $\mathbf{b}_n$, $\mathbf{b}_{n+1}$, and return vector $\mathbf{x}_n$ there exists a unique cost factor $w_n \in [0, 1)$, i.e. the portfolio is self financing. The value of cost factor w_n at day n is determined by portfolio vectors $\mathbf{b}_n$ and $\mathbf{b}_{n+1}$ as well as by return vector $\mathbf{x}_n$, i.e. $w_n = w(\mathbf{b}_n, \mathbf{b}_{n+1}, \mathbf{x}_n)$, for some function w . If we want to rearrange our portfolio substantially, then our net wealth decreases more considerably, however, it remains positive. Note also, that the cost does not restrict the set of new portfolio vectors, i.e., the optimization algorithm searches for optimal vector $\mathbf{b}_{n+1}$ within the whole simplex Δ_d . The value of the cost factor ranges between $\frac{1-c_s}{1+c_p} \leq w_n \leq 1$.

Starting with an initial wealth $S_0 = 1$ and $w_0 = 1$, wealth S_n at the closing time of the n -th market day becomes $S_n = N_{n-1} \langle \mathbf{b}_n, \mathbf{x}_n \rangle = w_{n-1} S_{n-1} \langle \mathbf{b}_n, \mathbf{x}_n \rangle = \prod_{i=1}^n [w(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{x}_{i-1}) \langle \mathbf{b}_i, \mathbf{x}_i \rangle]$. Introduce the notation

$$g(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{x}_{i-1}, \mathbf{x}_i) = \log(w(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{x}_{i-1}) \langle \mathbf{b}_i, \mathbf{x}_i \rangle), \quad (2)$$

then the average growth rate becomes

$$\frac{1}{n} \log S_n = \frac{1}{n} \sum_{i=1}^n \log(w(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{x}_{i-1}) \langle \mathbf{b}_i, \mathbf{x}_i \rangle) = \frac{1}{n} \sum_{i=1}^n g(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{x}_{i-1}, \mathbf{x}_i) \quad (3)$$

Our aim is to maximize this average growth rate.

In the sequel $\mathbf{x}_i$ will be random variable and is denoted by $\mathbf{X}_i$. Let's use the decomposition

$$\frac{1}{n} \log S_n = I_n + J_n, \quad (4)$$

where I_n is $\frac{1}{n} \sum_{i=1}^n (g(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}, \mathbf{X}_i) - \mathbb{E}\{g(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}, \mathbf{X}_i) | \mathbf{X}_1^{i-1}\})$ and $J_n = \frac{1}{n} \sum_{i=1}^n \mathbb{E}\{g(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}, \mathbf{X}_i) | \mathbf{X}_1^{i-1}\}$. I_n is an average of martingale differences. Under mild conditions on the support of the distribution of $\mathbf{X}$, $g(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}, \mathbf{X}_i)$ is bounded, therefore I_n is an average of bounded martingale differences, which converges to 0 almost surely, since according to the Chow Theorem (cf. Theorem 3.3.1 in Stout [18]) $\sum_{i=1}^{\infty} \frac{\mathbb{E}\{g(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}, \mathbf{X}_i)^2\}}{i^2} < \infty$ implies that $I_n \rightarrow 0$ almost surely. Thus, the asymptotic maximization of the average growth rate $\frac{1}{n} \log S_n$ is equivalent to the maximization of J_n .

If the market process $\{\mathbf{X}_i\}$ is a *homogeneous and first order Markov process* then, for appropriate portfolio selection $\{\mathbf{b}_i\}$, we have that

$$\begin{aligned} & \mathbb{E}\{g(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}, \mathbf{X}_i) | \mathbf{X}_1^{i-1}\} \\ &= \mathbb{E}\{\log(w(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}) \langle \mathbf{b}_i, \mathbf{X}_i \rangle) | \mathbf{X}_1^{i-1}\} \\ &= \log w(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}) + \mathbb{E}\{\log \langle \mathbf{b}_i, \mathbf{X}_i \rangle | \mathbf{X}_1^{i-1}\} \\ &= \log w(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}) + \mathbb{E}\{\log \langle \mathbf{b}_i, \mathbf{X}_i \rangle | \mathbf{b}_i, \mathbf{X}_{i-1}\} \\ &\stackrel{\text{def}}{=} v(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}), \end{aligned}$$

therefore the maximization of the average growth rate $\frac{1}{n} \log S_n$ is asymptotically equivalent to the maximization of

$$J_n = \frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_{i-1}, \mathbf{b}_i, \mathbf{X}_{i-1}). \quad (5)$$

Remark 2. Without transaction cost, the fundamental limits, determined in Algoet and Cover [2] reveal that the so-called *log-optimum portfolio* $\mathbf{B}^* = \{\mathbf{b}^*(\cdot)\}$ is the best possible choice. More precisely, in trading period n let $\mathbf{b}^*(\cdot)$ be such that $\mathbf{b}_n^*(\mathbf{X}_1^{n-1}) = \arg \max_{\mathbf{b}(\cdot)} \mathbb{E}\{\log \langle \mathbf{b}(\mathbf{X}_1^{n-1}), \mathbf{X}_n \rangle | \mathbf{X}_1^{n-1}\}$. If $S_n^* = S_n(\mathbf{B}^*)$ denotes the capital achieved by a log-optimum portfolio strategy $\mathbf{B}^*$, after n trading periods, then for any other investment strategy $\mathbf{B}$ with capital $S_n = S_n(\mathbf{B})$ and

for any stationary and ergodic return process $\{\mathbf{X}_n\}_{-\infty}^{\infty}$, $\liminf_{n \rightarrow \infty} \frac{1}{n} \log \frac{S_n^*}{S_n} \geq 0$ almost surely. Note that for first order Markovian return process $\mathbf{b}_n^*(\mathbf{X}_1^{n-1}) = \mathbf{b}_n^*(\mathbf{X}_{n-1}) = \arg \max_{\mathbf{b}(\cdot)} \mathbb{E} \{ \log \langle \mathbf{b}(\mathbf{X}_{n-1}), \mathbf{X}_n \rangle | \mathbf{X}_{n-1} \}$.

Before introducing to optimal strategies we show to empirical suboptimal algorithms with some experimental results.

Algorithm 1. For transaction cost, one may apply the portfolio $\mathbf{b}_n^*(\mathbf{X}_{n-1})$, and after calculating the portfolio subtract the transaction cost. Let's consider it's empirical counterpart in case of unknown market process. Apply the kernel based log-optimal portfolio selection introduced by Györfi, Lugosi and Udina [8] as follows: define an infinite array of experts $\mathbf{B}^{(\ell)} = \{\mathbf{b}^{(\ell)}(\cdot)\}$, where ℓ is a positive integer. For fixed positive integer ℓ , choose the radius $r_\ell > 0$ such that $\lim_{\ell \rightarrow \infty} r_\ell = 0$. Then, for $n > 1$, define the expert $\mathbf{b}^{(\ell)}$ as follows. Put

$$\mathbf{b}_n^{(\ell)} = \arg \max_{\mathbf{b} \in \Delta_d} \sum_{\{i < n: \|\mathbf{x}_{i-1} - \mathbf{x}_{n-1}\| \leq r_\ell\}} \ln \langle \mathbf{b}, \mathbf{x}_i \rangle, \quad (6)$$

if the sum is non-void, and $\mathbf{b}_0 = (1/d, \dots, 1/d)$ otherwise, where $\|\cdot\|$ denotes the Euclidean norm. These experts are aggregated as follows: let $\{q_\ell\}$ be a probability distribution over the set of all positive integers ℓ . Consider two types of aggregations: By aggregating with the wealth the initial capital $S_0 = 1$ is distributed among the expert according to the distribution $\{q_\ell\}$, and the expert makes the portfolio selection and pays for transaction cost individually. If $S_n(\mathbf{B}^{(\ell)})$ is the capital accumulated by the elementary strategy $\mathbf{B}^{(\ell)}$ the investor's wealth after period n , aggregations $S_n = \sum_\ell q_\ell S_n(\mathbf{B}^{(\ell)})$. In the second case the aggregated portfolio pays for transaction cost. Here $S_n(\mathbf{B}^{(\ell)})$ is again the capital accumulated by the elementary strategy $\mathbf{B}^{(\ell)}$ after n periods. Then, after period n , the investor's aggregated portfolio becomes $\mathbf{b}_n = \frac{\sum_\ell q_\ell S_{n-1}(\mathbf{B}^{(\ell)}) \mathbf{b}_n^{(\ell)}}{\sum_\ell q_\ell S_{n-1}(\mathbf{B}^{(\ell)})}$. The investor's capital is $S_n = S_{n-1}(\mathbf{b}_n, \mathbf{x}_n) w(\mathbf{b}_{n-1}, \mathbf{b}_n, \mathbf{x}_{n-1})$, so the aggregated portfolio pays for the transaction cost.

Algorithm 2. Our second suboptimal strategy is a one-step optimization as follows: put $\mathbf{b}_1 = \{1/d, \dots, 1/d\}$ and for $i \geq 1$, $\mathbf{b}_{i+1} = \arg \max_{\mathbf{b}'} v(\mathbf{b}_i, \mathbf{b}', \mathbf{X}_i)$. Obviously, this portfolio has no global optimality property. Let's consider its empirical counterpart in case of unknown distribution. Put $\mathbf{b}_1 = \{1/d, \dots, 1/d\}$ and for $n \geq 1$,

$$\mathbf{b}_n^{(\ell)} = \arg \max_{\mathbf{b} \in \Delta_d} \sum_{\{i < n: \|\mathbf{x}_{i-1} - \mathbf{x}_{n-1}\| \leq r_\ell\}} (\ln \langle \mathbf{b}, \mathbf{x}_i \rangle + \ln w(\mathbf{b}_{n-1}, \mathbf{b}, \mathbf{x}_{n-1})), \quad (7)$$

if the sum is non-void, and $\mathbf{b}_0 = (1/d, \dots, 1/d)$ otherwise. These elementary portfolios are mixed as in Algorithm 1.

The investment strategies are tested on a data set containing 23 stocks and has length 44 years - 11178 trading days ending in 2006 - achievable at www.szit.bme.hu/oti/portfolio. The proposed empirical portfolio selection algorithms use an infinite set of experts. In the experiment we selected $L = 10$. Choose $\{q_\ell\} = 1/L$ over the experts in use, and the radius $r_\ell^2 = 0.0001 \cdot d \cdot \ell$, $r_\ell^2 = 0.0002 \cdot d +$

$0.00002 \cdot d \cdot \ell$, for $\ell = 1, \dots, L$. Table 1, [10] summarizes the average annual yield achieved by each expert at the last period when investing one unit for the kernel-based log-optimal portfolio. Experts are indexed by $\ell = 1 \dots 10$ in rows. The second column contains the average annual yields of experts for kernel based log-optimal portfolio if there is no transaction cost, and in this case the results of the two aggregations are the same: 124%. The third and fourth columns contain the average annual yields of experts for kernel based log-optimal portfolio if the commission factor is $c = 0.0015$.

In the sequel we prove to optimal strategies in case of known distribution as these algorithms were not optimal. The empirical optimal counterparts of them are still missing.

3 The Related Markov Control Problem

The problem of optimal investment with proportional transaction costs has been essentially formulated in continuous time only, in the classical articles Davis and Norman [6], Taksar et al. [19] and Shreve and Soner [17], etc. Taksar, Klass and Assaf [19] investigate optimal investment in a continuous time market with two assets and with proportional transaction costs driven by a Wiener process and using long run expected reward criteria. Akien, Sulem and Taksar [1] extend these results to the case of several risky assets.

Most of the above mentioned papers use some kind of method from stochastic optimal control theory. Without exception all the papers consider optimality in expected reward. None of these papers give result on almost sure optimality. In this paper we present two portfolio selection strategies, and for a Markovian market we prove their almost sure optimality.

Discrete time portfolio optimization with transaction cost is a special case of the general Markov control processes (MCP). A discrete time Markov control process

Table 1. The average annual yields of the individual experts and of the aggregations with $c = 0.0015$

ℓ	$c = 0$	Algorithm 1	Algorithm 2
1	68%	-6%	27%
2	87%	3%	38%
3	94%	7%	45%
4	94%	6%	49%
5	108%	14%	54%
6	118%	19%	60%
7	122%	21%	69%
8	128%	25%	73%
9	131%	27%	71%
10	131%	28%	72%
Aggregation with wealth	124%	24%	68%
Aggregation with portfolio	124%	29%	73%

is defined by a five tuple $(S, A, U(s), Q, r)$ (cf. [11]). S is a Borel space, called the state space, the action space A is Borel, too, the space of admissible actions $U(s)$ is a Borel subset of A . Let the set K be $\{(s, a) : s \in S, a \in U(s)\}$. The transition law is a stochastic kernel $Q(\cdot|s, a)$ on S given K , and $r(s, a)$ is the reward function.

The evolution of the process is the following. Let S_t denote the state at time t , action A_t is chosen at that time. Let $S_t = s$ and $A_t = a$, then the reward is $r(s, a)$, and the process moves to S_{t+1} according to the kernel $Q(\cdot|s, a)$. A control policy is a sequence $\pi = \{\pi_n\}$ of deterministic functions, which can also be stochastic kernels on A given the past states and actions.

Two reward criteria are considered. The expected long run average reward for π is defined by $J(\pi) = \liminf_{n \rightarrow \infty} \frac{1}{n} \sum_{t=0}^{n-1} \mathbb{E}^\pi r(S_t, A_t)$. The sample-path average reward is defined as $J(\pi) := \liminf_{n \rightarrow \infty} \frac{1}{n} \sum_{t=0}^{n-1} r(S_t, A_t)$. In the theory of MCP most of the results correspond to expected long run average reward, while just a few present result for the sample-path criterion. Such sample-path results can be found in [3] for bounded rewards and in [20], [14] for unbounded rewards. For Markov control processes, the main aim is to approach the maximum asymptotic reward: $J^* = \sup_\pi J(\pi)$, which leads to a dynamic programming problem.

For portfolio optimization with transaction cost, we formulate the corresponding Markov control problem. Assume that there exist $0 < a_1 < 1 < a_2 < \infty$ such that

$$\mathbf{X}_i \in [a_1, a_2]^d.$$

1. Let us define the state space as: $S := \{(\mathbf{b}, \mathbf{x}) | \mathbf{b} \in \Delta_d, \mathbf{x} \in [a_1, a_2]^d\}$.
2. The action space is $A := \Delta_d$.
3. For the set of admissible actions we get $U(\mathbf{b}, \mathbf{x}) := \Delta_d$.
4. The stochastic kernel is $Q(d(\mathbf{b}', \mathbf{x}') | (\mathbf{b}, \mathbf{x}), \mathbf{b}') := P(d\mathbf{x}' | \mathbf{x}) := \mathbb{P}\{d\mathbf{X}_2 = d\mathbf{x}' | \mathbf{X}_1 = \mathbf{x}\}$ the transition probability distribution of the Markov market process, describing the asset returns. Note, that this corresponds to the assumption, that the market behaviour is not affected by the investor.
5. The reward function is: $r((\mathbf{b}, \mathbf{x}), \mathbf{b}') = v(\mathbf{b}, \mathbf{b}', \mathbf{x})$.
6. The sample-path average reward criterion is the following:

$$\liminf_{n \rightarrow \infty} \frac{1}{n} \sum_{t=1}^n r((\mathbf{b}_{t-1}, \mathbf{x}_{t-1}), \mathbf{b}_t) = \liminf_{n \rightarrow \infty} \frac{1}{n} \sum_{t=1}^n v(\mathbf{b}_{t-1}, \mathbf{b}_t, \mathbf{x}_{t-1})$$

$$= \liminf_{n \rightarrow \infty} J_n.$$

Remark 3. It should be noted that the methods of MCP literature, more precisely the theorems in [3], [20], [14] can't be applied in our case. However, we do use the formalism, the results on the existence of the solution of discounted Bellman equations, and the basic idea of vanishing discount approach.

4 Optimal Portfolio Selection Algorithms

We introduce two optimal portfolio selection strategies. Let $0 < \delta < 1$ denote a discount factor. We apply a kind of vanishing discount approach, formulated by the discounted Bellman equation:

$$F_\delta(\mathbf{b}, \mathbf{x}) = \max_{\mathbf{b}'} \{v(\mathbf{b}, \mathbf{b}', \mathbf{x}) + (1 - \delta)\mathbb{E}\{F_\delta(\mathbf{b}', \mathbf{X}_2) | \mathbf{X}_1 = \mathbf{x}\}\}. \quad (8)$$

It can be shown that this discounted Bellman equation has a solution (cf. Hernández-Lerma, Lasserie [11], Schäfer [16]).

Strategy 1. Our first portfolio selection strategy is the following put $\mathbf{b}_1^* = \{1/d, \dots, 1/d\}$ and

$$\mathbf{b}_{i+1}^* = \arg \max_{\mathbf{b}'} \{v(\mathbf{b}_i^*, \mathbf{b}', \mathbf{X}_i) + (1 - \delta_i) \mathbb{E}\{F_{\delta_i}(\mathbf{b}', \mathbf{X}_{i+1}) | \mathbf{X}_i\}\}, \quad (9)$$

for $1 \leq i$, where $0 < \delta_i < 1$ is a discount factor such that $\delta_i \downarrow 0$.

Remark 4. A strategy similar to (9) was defined by Schäfer [16]. He introduces an additional asset to settle the transaction costs when the portfolio is restructured.

Remark 5. A portfolio selection $\{\mathbf{b}_i\}$ is called recursive if it has the form $\mathbf{b}_i = \mathbf{b}_i(\mathbf{x}_1^{i-1}) = \mathbf{b}_i(\mathbf{b}_{i-1}, \mathbf{x}_{i-1})$. Obviously, the portfolio $\{\mathbf{b}_i^*\}$ is recursive. The recursion in the definition of the portfolio $\{\mathbf{b}_i^*\}$ is not time invariant, i.e., it is a *non-stationary portfolio selection rule*.

Now, we claim our result on the optimality of Strategy 1 with respect to a sample-path average criterion:

Theorem 1. *Assume*

- (i) *that $\{\mathbf{X}_i\}$ is a homogeneous and first order Markov process,*
- (ii) *and there exist $0 < a_1 < 1 < a_2 < \infty$ such that $a_1 \leq X^{(j)} \leq a_2$ for all $j = 1, \dots, d$.*

Choose the discount factor $\delta_i \downarrow 0$ such that $(\delta_i - \delta_{i+1})/\delta_{i+1}^2 \rightarrow 0$ as $i \rightarrow \infty$, and $\sum_{n=1}^{\infty} \frac{1}{n^2 \delta_n^2} < \infty$. Then, for Strategy 1, the portfolio $\{\mathbf{b}_i^*\}$ with capital S_n^* is optimal in the sense that for any portfolio strategy $\{\mathbf{b}_i\}$ with capital S_n ,

$$\liminf_{n \rightarrow \infty} \left(\frac{1}{n} \log S_n^* - \frac{1}{n} \log S_n \right) \geq 0 \text{ a.s.}$$

Remark 6. $\liminf_{n \rightarrow \infty} \mathbb{E} \left\{ \frac{1}{n} \log S_n^* - \frac{1}{n} \log S_n \right\} \geq 0$, according to Theorem 4.2.1 in Schäfer [16], i.e., the portfolio $\{\mathbf{b}_i^*\}$ is optimal in expectation. Theorem 1 states that the portfolio strategy $\{\mathbf{b}_i^*\}$ is sample-path optimal, too, i.e., it is optimal with probability one.

Remark 7. For the choice $\delta_i = i^{-\epsilon}$, with $\epsilon < 1/2$, the conditions of Theorem 1 are satisfied.

Remark 8. For the standard stock market problems, the condition (i) is satisfied with $a_1 = 0.9$ and $a_2 = 1.1$, (cf. Fernholz [7]).

Strategy 2. Next, we introduce a portfolio with *stationary* (time invariant) recursion such that this portfolio is a sample-path optimal policy, too. For any integer $1 \leq k$, put $\mathbf{b}_1^{(k)} = \{1/d, \dots, 1/d\}$ and

$$\mathbf{b}_{i+1}^{(k)} = \arg \max_{\mathbf{b}'} \{v(\mathbf{b}_i^{(k)}, \mathbf{b}', \mathbf{X}_i) + (1 - \delta_k) \mathbb{E}\{F_{\delta_k}(\mathbf{b}', \mathbf{X}_{i+1}) | \mathbf{X}_i\}\}, \quad (10)$$

for $1 \leq i$. The portfolio $\mathbf{B}^{(k)} = \{\mathbf{b}_i^{(k)}\}$ is called the portfolio of expert k with capital $S_n(\mathbf{B}^{(k)})$. We combine the experts borrowing a current technique from machine learning, the exponential weighing (cf. Cesa-Bianchi and Lugosi [5]). Combine the experts as follows: let $\{q_k\} \geq 0$ be a probability distribution on $1 \leq k$ and aggregate the experts with the wealth as described earlier by Algorithm 1, so $\tilde{S}_n(\tilde{\mathbf{B}}) = \sum_k q_k S_n(\mathbf{B}^{(k)})$.

Theorem 2. Assume (i) and (ii) of Theorem 1. Choose the discount factor $\delta_i \downarrow 0$ as $i \rightarrow \infty$. Then, for Strategy 2,

$$\lim_{n \rightarrow \infty} \left(\frac{1}{n} \log S_n^* - \frac{1}{n} \log \tilde{S}_n \right) = 0 \text{ a.s.}$$

5 Summary

We considered discrete time infinite horizon growth optimal investment with several assets in stock markets with proportional transactions costs. The asset returns followed homogeneous Markov processes. Using techniques from dynamic programming and machine learning two recursive investment strategies were shown, such that, in the long run, the growth rate on trajectories were greater than or equal to the growth rate of any other investment strategy with probability 1. An important direction of our future work is to construct an empirical version of the second stationary rule, i.e., to get a data driven portfolio selection (cf. Györfi *et al.* [8], Györfi and Schäfer [9]) when the distribution of the market process is unknown.

6 Proofs

Proof of Theorem 1. Introduce the following notation: $F_i(\mathbf{b}, \mathbf{x}) = F_{\delta_i}(\mathbf{b}, \mathbf{x})$. We have to show that

$$\liminf_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n (g(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i, \mathbf{X}_{i+1}) - g(\mathbf{b}_i, \mathbf{b}_{i+1}, \mathbf{X}_i, \mathbf{X}_{i+1})) \geq 0$$

a.s. Because of the martingale difference argument in Section 2, one has

$$\begin{aligned} & \liminf_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n (g(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i, \mathbf{X}_{i+1}) - g(\mathbf{b}_i, \mathbf{b}_{i+1}, \mathbf{X}_i, \mathbf{X}_{i+1})) \\ &= \liminf_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n (v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i) - v(\mathbf{b}_i, \mathbf{b}_{i+1}, \mathbf{X}_i)) \end{aligned}$$

a.s. therefore we have to prove that

$$\liminf_{n \rightarrow \infty} \left(\frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i) - \frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_i, \mathbf{b}_{i+1}, \mathbf{X}_i) \right) \geq 0 \quad (11)$$

a.s. (9) implies that

$$F_i(\mathbf{b}_i^*, \mathbf{X}_i) = v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i) + (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) | \mathbf{b}_{i+1}^*, \mathbf{X}_i\}, \quad (12)$$

while for any portfolio $\{\mathbf{b}_i\}$,

$$F_i(\mathbf{b}_i, \mathbf{X}_i) \geq v(\mathbf{b}_i, \mathbf{b}_{i+1}, \mathbf{X}_i) + (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{b}_{i+1}, \mathbf{X}_i\}. \quad (13)$$

Because of (12) and (13), we get that $\frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i)$

$$\begin{aligned} &= \frac{1}{n} \sum_{i=1}^n (F_i(\mathbf{b}_i^*, \mathbf{X}_i) - (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) | \mathbf{b}_{i+1}^*, \mathbf{X}_i\}) \\ &= \frac{1}{n} \sum_{i=1}^n (F_i(\mathbf{b}_i^*, \mathbf{X}_i) - (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\}) \end{aligned}$$

and

$$\begin{aligned} \frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_i, \mathbf{b}_{i+1}, \mathbf{X}_i) &\leq \frac{1}{n} \sum_{i=1}^n (F_i(\mathbf{b}_i, \mathbf{X}_i) - (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{b}_{i+1}, \mathbf{X}_i\}) \\ &= \frac{1}{n} \sum_{i=1}^n (F_i(\mathbf{b}_i, \mathbf{X}_i) - (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\}), \end{aligned}$$

therefore

$$\begin{aligned} &\frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i) - \frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_i, \mathbf{b}_{i+1}, \mathbf{X}_i) \\ &\geq \frac{1}{n} \sum_{i=1}^n (F_i(\mathbf{b}_i^*, \mathbf{X}_i) - (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\}) \\ &\quad - \frac{1}{n} \sum_{i=1}^n (F_i(\mathbf{b}_i, \mathbf{X}_i) - (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\}). \end{aligned}$$

Apply the following identity

$$\begin{aligned} &(1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} - F_i(\mathbf{b}_i, \mathbf{X}_i) \\ &= \mathbb{E}\{F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} - F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) \\ &\quad + F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) - F_i(\mathbf{b}_i, \mathbf{X}_i) \\ &\quad - \delta_i \mathbb{E}\{F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} \\ &= a_i + b_i + c_i. \end{aligned}$$

Because of $F_i(\mathbf{b}, \mathbf{x}) = \max_{\mathbf{b}'} \{v(\mathbf{b}, \mathbf{b}', \mathbf{x}) + (1 - \delta_i) \mathbb{E}(F_i(\mathbf{b}', \mathbf{X}_{i+1}) | \mathbf{X}_i = \mathbf{x})\}$ we have that $\|F_i\|_\infty \leq \|v\|_\infty + (1 - \delta_i) \|F_i\|_\infty$, therefore $\|F_i\|_\infty \leq \frac{\|v\|_\infty}{\delta_i}$ (cf. Lemma 4.2.3 in Schäfer [16]). As $\{a_i\}$ is a sequence of martingale differences such that

$|a_i| \leq 2\|F_i\|_\infty \leq \frac{2}{\delta_i}\|v\|_\infty$, therefore, because of $\sum_n \frac{1}{n^2\delta_n^2} < \infty$, the Chow Theorem implies that $\frac{1}{n} \sum_{i=1}^n a_i \rightarrow 0$ (14) a.s. (cf. Stout [18]).

Similarly to the bounding above, we have the equality

$$F_i(\mathbf{b}, \mathbf{x}) = \max_{\mathbf{b}'} \{v(\mathbf{b}, \mathbf{b}', \mathbf{x}) + (1 - \delta_i)\mathbb{E}(F_i(\mathbf{b}', \mathbf{X}_{i+1})|\mathbf{X}_i = \mathbf{x})\}$$

and the inequality

$$\begin{aligned} F_{i+1}(\mathbf{b}, \mathbf{x}) &= \max_{\mathbf{b}''} \{v(\mathbf{b}, \mathbf{b}'', \mathbf{x}) + (1 - \delta_{i+1})\mathbb{E}(F_{i+1}(\mathbf{b}'', \mathbf{X}_{i+2})|\mathbf{X}_{i+1} = \mathbf{x})\} \\ &\geq v(\mathbf{b}, \mathbf{b}', \mathbf{x}) + (1 - \delta_{i+1})\mathbb{E}(F_{i+1}(\mathbf{b}', \mathbf{X}_{i+1})|\mathbf{X}_i = \mathbf{x}) \end{aligned}$$

with arbitrary $\mathbf{b}'$. Taking difference

$$\begin{aligned} &F_i(\mathbf{b}, \mathbf{x}) - F_{i+1}(\mathbf{b}, \mathbf{x}) \\ &\leq \max_{\mathbf{b}'} \{(1 - \delta_i)\mathbb{E}(F_i(\mathbf{b}', \mathbf{X}_{i+1})|\mathbf{X}_i = \mathbf{x}) \\ &\quad - (1 - \delta_{i+1})\mathbb{E}(F_{i+1}(\mathbf{b}', \mathbf{X}_{i+1})|\mathbf{X}_i = \mathbf{x})\} \\ &\leq (1 - \delta_i)\|F_i - F_{i+1}\|_\infty + (\delta_{i+1} - \delta_i) \max_{\mathbf{b}'} \mathbb{E}(F_{i+1}(\mathbf{b}', \mathbf{X}_{i+1})|\mathbf{X}_i = \mathbf{x}) \\ &\leq (1 - \delta_i)\|F_i - F_{i+1}\|_\infty + (\delta_{i+1} - \delta_i)\|F_{i+1}\|_\infty. \end{aligned}$$

So we have $\|F_i - F_{i+1}\|_\infty \leq \frac{\delta_i - \delta_{i+1}}{\delta_i}\|F_{i+1}\|_\infty$. Using that $\|F_{i+1}\|_\infty \leq \frac{\|v\|_\infty}{\delta_{i+1}}$ and assumption on δ_i 's, we get that $\|F_i - F_{i+1}\|_\infty \leq \|v\|_\infty \frac{\delta_i - \delta_{i+1}}{\delta_i^2}$ (cf. Lemma 4.2.3 in Schäfer [16]). Concerning $\{b_i\}$,

$$\begin{aligned} &\left| \frac{1}{n} \sum_{i=1}^n b_i \right| = \left| \frac{1}{n} \sum_{i=1}^n (F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) - F_i(\mathbf{b}_i, \mathbf{X}_i)) \right| \\ &\leq \left| \frac{1}{n} \sum_{i=1}^n (F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) - F_{i+1}(\mathbf{b}_{i+1}, \mathbf{X}_{i+1})) \right| \\ &\quad + \left| \frac{1}{n} \sum_{i=1}^n (F_{i+1}(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) - F_i(\mathbf{b}_i, \mathbf{X}_i)) \right| \\ &\leq \frac{1}{n} \sum_{i=1}^n \|F_i - F_{i+1}\|_\infty + \left| \frac{1}{n} (F_{n+1}(\mathbf{b}_{n+1}, \mathbf{X}_{n+1}) - F_1(\mathbf{b}_1, \mathbf{X}_1)) \right| \\ &\leq \frac{1}{n} \sum_{i=1}^n \|F_i - F_{i+1}\|_\infty + \frac{\|F_{n+1}\|_\infty + \|F_1\|_\infty}{n} \\ &\leq \|v\|_\infty \frac{1}{n} \sum_{i=1}^n \frac{|\delta_{i+1} - \delta_i|}{\delta_{i+1}^2} + \|v\|_\infty \frac{1/\delta_{n+1} + 1/\delta_1}{n} \rightarrow 0 \end{aligned} \tag{14}$$

by conditions. Concerning the proof of (11) what is left to show that

$$\limsup_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n \delta_i (\mathbb{E}\{F_i(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1})|\mathbf{X}_1^i\} - \mathbb{E}\{F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1})|\mathbf{X}_1^i\}) \leq 0$$

a.s. The definition of F_i implies that

$$\begin{aligned}
& F_i(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) - F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) \\
&= \max_{\mathbf{b}'} \{v(\mathbf{b}_{i+1}^*, \mathbf{b}', \mathbf{X}_{i+1}) + (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}', \mathbf{X}_{i+2}) | \mathbf{X}_{i+1}\}\} \\
&\quad - \max_{\mathbf{b}''} \{v(\mathbf{b}_{i+1}, \mathbf{b}'', \mathbf{X}_{i+1}) + (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}'', \mathbf{X}_{i+2}) | \mathbf{X}_{i+1}\}\} \\
&\leq \max_{\mathbf{b}'} \{v(\mathbf{b}_{i+1}^*, \mathbf{b}', \mathbf{X}_{i+1}) + (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}', \mathbf{X}_{i+2}) | \mathbf{X}_{i+1}\} \\
&\quad - v(\mathbf{b}_{i+1}, \mathbf{b}', \mathbf{X}_{i+1}) - (1 - \delta_i) \mathbb{E}\{F_i(\mathbf{b}', \mathbf{X}_{i+2}) | \mathbf{X}_{i+1}\}\} \\
&\leq \max_{\mathbf{b}'} \{v(\mathbf{b}_{i+1}^*, \mathbf{b}', \mathbf{X}_{i+1}) - v(\mathbf{b}_{i+1}, \mathbf{b}', \mathbf{X}_{i+1})\} \\
&\leq 2\|v\|_\infty,
\end{aligned}$$

therefore

$$\frac{1}{n} \sum_{i=1}^n \delta_i \mathbb{E}\{F_i(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) - F_i(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} \leq \frac{2\|v\|_\infty}{n} \sum_{i=1}^n \delta_i \rightarrow 0. \quad (15)$$

(6), (14) and (15) imply (11). ■

Proof of Theorem 2

Theorem 1 implies that $\liminf_{n \rightarrow \infty} \left(\frac{1}{n} \log S_n^* - \frac{1}{n} \log \tilde{S}_n \right) \geq 0$ a.s.. We have to show that

$$\liminf_{n \rightarrow \infty} \left(\frac{1}{n} \log \tilde{S}_n - \frac{1}{n} \log S_n^* \right) \geq 0 \quad (16)$$

a.s. By definition, $\frac{1}{n} \log \tilde{S}_n = \frac{1}{n} \log \sum_{k=1}^\infty q_k S_n(\mathbf{B}^{(k)}) \geq \frac{1}{n} \log \sup_k q_k S_n(\mathbf{B}^{(k)}) = \sup_k \left(\frac{\log q_k}{n} + \frac{1}{n} \log S_n(\mathbf{B}^{(k)}) \right)$, therefore (16) follows from the following:

$$\begin{aligned}
& \liminf_{n \rightarrow \infty} \left(\sup_k \left(\frac{\log q_k}{n} + \frac{1}{n} \sum_{i=1}^n g(\mathbf{b}_i^{(k)}, \mathbf{b}_{i+1}^{(k)}, \mathbf{X}_i, \mathbf{X}_{i+1}) \right) \right. \\
& \quad \left. - \frac{1}{n} \sum_{i=1}^n g(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i, \mathbf{X}_{i+1}) \right) \geq 0 \text{ a.s. which is equivalent to}
\end{aligned}$$

$$\liminf_{n \rightarrow \infty} \sup_k \left(\frac{\log q_k}{n} + \frac{1}{n} \sum_{i=1}^n (v(\mathbf{b}_i^{(k)}, \mathbf{b}_{i+1}^{(k)}, \mathbf{X}_i) - v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i)) \right) \geq 0 \quad (17)$$

a.s. (10) implies that

$$F_k(\mathbf{b}_i^{(k)}, \mathbf{X}_i) = v(\mathbf{b}_i^{(k)}, \mathbf{b}_{i+1}^{(k)}, \mathbf{X}_i) + (1 - \delta_k) \mathbb{E}\{F_k(\mathbf{b}_{i+1}^{(k)}, \mathbf{X}_{i+1}) | \mathbf{b}_{i+1}^{(k)}, \mathbf{X}_i\}, \quad (18)$$

while for any portfolio $\{\mathbf{b}_i\}$,

$$F_k(\mathbf{b}_i, \mathbf{X}_i) \geq v(\mathbf{b}_i, \mathbf{b}_{i+1}, \mathbf{X}_i) + (1 - \delta_k) \mathbb{E}\{F_k(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{b}_{i+1}, \mathbf{X}_i\},$$

thus for the portfolio $\{\mathbf{b}_i^*\}$

$$F_k(\mathbf{b}_i^*, \mathbf{X}_i) \geq v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i) + (1 - \delta_k) \mathbb{E}\{F_k(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) | \mathbf{b}_{i+1}^*, \mathbf{X}_i\}. \quad (19)$$

Because of (18) and (19), we get that $\frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i) \leq$

$$\begin{aligned} &\leq \frac{1}{n} \sum_{i=1}^n (F_k(\mathbf{b}_i^*, \mathbf{X}_i) - (1 - \delta_k) \mathbb{E}\{F_k(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) | \mathbf{b}_{i+1}^*, \mathbf{X}_i\}) \\ &= \frac{1}{n} \sum_{i=1}^n (F_k(\mathbf{b}_i^*, \mathbf{X}_i) - (1 - \delta_k) \mathbb{E}\{F_k(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\}) \end{aligned}$$

and $\frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_i^{(k)}, \mathbf{b}_{i+1}^{(k)}, \mathbf{X}_i) =$

$$\begin{aligned} &= \frac{1}{n} \sum_{i=1}^n \left(F_k(\mathbf{b}_i^{(k)}, \mathbf{X}_i) - (1 - \delta_k) \mathbb{E}\{F_k(\mathbf{b}_{i+1}^{(k)}, \mathbf{X}_{i+1}) | \mathbf{b}_{i+1}^{(k)}, \mathbf{X}_i\} \right) \\ &= \frac{1}{n} \sum_{i=1}^n \left(F_k(\mathbf{b}_i^{(k)}, \mathbf{X}_i) - (1 - \delta_k) \mathbb{E}\{F_k(\mathbf{b}_{i+1}^{(k)}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} \right), \end{aligned}$$

therefore

$$\begin{aligned} &\frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_i^{(k)}, \mathbf{b}_{i+1}^{(k)}, \mathbf{X}_i) - \frac{1}{n} \sum_{i=1}^n v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i) \\ &\geq \frac{1}{n} \sum_{i=1}^n \left(F_k(\mathbf{b}_i^{(k)}, \mathbf{X}_i) - (1 - \delta_k) \mathbb{E}\{F_k(\mathbf{b}_{i+1}^{(k)}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} \right) \\ &\quad - \frac{1}{n} \sum_{i=1}^n \left(F_k(\mathbf{b}_i^*, \mathbf{X}_i) - (1 - \delta_k) \mathbb{E}\{F_k(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} \right). \end{aligned}$$

Apply the following identity

$$\begin{aligned} &(1 - \delta_k) \mathbb{E}\{F_k(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} - F_k(\mathbf{b}_i, \mathbf{X}_i) \\ &= \mathbb{E}\{F_k(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} - F_k(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) + F_k(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) - F_k(\mathbf{b}_i, \mathbf{X}_i) \\ &\quad - \delta_k \mathbb{E}\{F_k(\mathbf{b}_{i+1}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} \\ &= a_i + b_i + c_i. \end{aligned}$$

Similarly to the proof of Theorem 1, the averages of a_i 's and b_i 's tend to zero a.s., so concerning (17) we have that, with probability one,

$$\begin{aligned} &\liminf_{n \rightarrow \infty} \sup_k \left(\frac{\log q_k}{n} + \frac{1}{n} \sum_{i=1}^n (v(\mathbf{b}_i^{(k)}, \mathbf{b}_{i+1}^{(k)}, \mathbf{X}_i) - v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i)) \right) \\ &\geq \sup_k \liminf_{n \rightarrow \infty} \left(\frac{\log q_k}{n} + \frac{1}{n} \sum_{i=1}^n (v(\mathbf{b}_i^{(k)}, \mathbf{b}_{i+1}^{(k)}, \mathbf{X}_i) - v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i)) \right) \\ &= \sup_k \liminf_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n (v(\mathbf{b}_i^{(k)}, \mathbf{b}_{i+1}^{(k)}, \mathbf{X}_i) - v(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \mathbf{X}_i)) \\ &= \sup_k \liminf_{n \rightarrow \infty} \frac{\delta_k}{n} \sum_{i=1}^n (\mathbb{E}\{F_k(\mathbf{b}_{i+1}^{(k)}, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\} - \mathbb{E}\{F_k(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) | \mathbf{X}_1^i\}). \end{aligned}$$

The problem left is to show that the last term is non negative a.s. Using the definition of F_k

$$\begin{aligned}
& F_k(\mathbf{b}_{i+1}^{(k)}, \mathbf{X}_{i+1}) - F_k(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1}) \\
&= \max_{\mathbf{b}'} \{v(\mathbf{b}_{i+1}^{(k)}, \mathbf{b}', \mathbf{X}_{i+1}) + (1 - \delta_k)\mathbb{E}\{F_k(\mathbf{b}', \mathbf{X}_{i+2})|\mathbf{X}_{i+1}\}\} \\
&\quad - \max_{\mathbf{b}''} \{v(\mathbf{b}_{i+1}^*, \mathbf{b}'', \mathbf{X}_{i+1}) + (1 - \delta_k)\mathbb{E}\{F_k(\mathbf{b}'', \mathbf{X}_{i+2})|\mathbf{X}_{i+1}\}\} \\
&= \max_{\mathbf{b}'} \min_{\mathbf{b}''} \{ \{v(\mathbf{b}_{i+1}^{(k)}, \mathbf{b}', \mathbf{X}_{i+1}) + (1 - \delta_k)\mathbb{E}\{F_k(\mathbf{b}', \mathbf{X}_{i+2})|\mathbf{X}_{i+1}\}\} \\
&\quad - \{v(\mathbf{b}_{i+1}^*, \mathbf{b}'', \mathbf{X}_{i+1}) + (1 - \delta_k)\mathbb{E}\{F_k(\mathbf{b}'', \mathbf{X}_{i+2})|\mathbf{X}_{i+1}\}\} \} \\
&\geq \min_{\mathbf{b}''} \{ \{v(\mathbf{b}_{i+1}^{(k)}, \mathbf{b}'', \mathbf{X}_{i+1}) + (1 - \delta_k)\mathbb{E}\{F_k(\mathbf{b}'', \mathbf{X}_{i+2})|\mathbf{X}_{i+1}\}\} \\
&\quad - \{v(\mathbf{b}_{i+1}^*, \mathbf{b}'', \mathbf{X}_{i+1}) + (1 - \delta_k)\mathbb{E}\{F_k(\mathbf{b}'', \mathbf{X}_{i+2})|\mathbf{X}_{i+1}\}\} \} \\
&= \min_{\mathbf{b}''} \{v(\mathbf{b}_{i+1}^{(k)}, \mathbf{b}'', \mathbf{X}_{i+1}) - v(\mathbf{b}_{i+1}^*, \mathbf{b}'', \mathbf{X}_{i+1})\} \\
&\geq -2\|v\|_\infty,
\end{aligned}$$

therefore

$$\begin{aligned}
\sup_k \liminf_{n \rightarrow \infty} \frac{\delta_k}{n} \sum_{i=1}^n \mathbb{E}\{F_k(\mathbf{b}_{i+1}^{(k)}, \mathbf{X}_{i+1}) - F_k(\mathbf{b}_{i+1}^*, \mathbf{X}_{i+1})|\mathbf{X}_1^i\} &\geq \sup_k \delta_k(-2\|v\|_\infty) \\
&= 0
\end{aligned}$$

a.s., and (17) is proved. ■

References

- [1] Akien, M., Sulem, A., Taksar, M.I.: Dynamic Optimization of Long-term Growth Rate for a Portfolio with Transaction Costs and Logarithmic Utility. *Mathematical Finance* 11, 153–188 (2001)
- [2] Algoet, P., Cover, T.: Asymptotic Optimality Asymptotic Equipartition Properties of Log-optimum Investments. *Annals of Probability* 16, 876–898 (1988)
- [3] Arapostathis, A., Borkar, V.S., Fernandez-Gaucherand, E., Ghosh, M.K., Marcus, S.I.: Discrete-time Controlled Markov Processes with Average Cost Criterion: a Survey. *SIAM J. Control Optimization* 31, 282–344 (1993)
- [4] Bobryk, R.V., Stettner, L.: Discrete Time Portfolio Selection with Proportional Transaction Costs. *Probability and Mathematical Statistics* 19, 235–248 (1999)
- [5] Cesa-Bianchi, N., Lugosi, G.: *Prediction, Learning, and Games*. Cambridge University Press, Cambridge (2006)
- [6] Davis, M.H.A., Norman, A.R.: Portfolio Selection with Transaction Costs. *Mathematics of Operations Research* 15, 676–713 (1990)
- [7] Fernholz, E.R.: *Stochastic Portfolio Theory*. Springer, New York (2000)
- [8] Györfi, L., Lugosi, G., Udina, F.: Nonparametric Kernel-based Sequential Investment Strategies. *Mathematical Finance* 16, 337–357 (2006)
- [9] Györfi, L., Schäfer, D.: Nonparametric Prediction. In: Suykens, J.A.K., Horváth, G., Basu, S., Micchelli, C., Vandevale, J. (eds.) *Advances in Learning Theory: Methods, Models and Applications*, pp. 339–354. IOS Press, Amsterdam (2003)

- [10] Györfi, L., Ottucsák, G.: Empirical log-optimal portfolio selections: a survey, manuscript (2007), <http://www.szit.bme.hu/oti/portfolio/articles/tgyorfi.pdf>
- [11] Hernández-Lerma, L., Lasserre, J.B.: Discrete-Time Markov Control Processes: Basic Optimality Criteria. Springer, New York (1996)
- [12] Iyengar, G.: Discrete Time Growth Optimal Investment with Costs. Working Paper (2002), <http://www.columbia.edu/gi10/Papers/stochastic.pdf>
- [13] Iyengar, G., Cover, T.: Growth Optimal Investment in Horse Race Markets with Costs. IEEE Transactions on Information Theory 46, 2675–2683 (2000)
- [14] Lasserre, J.B.: Sample-path Average Optimality for Markov Control Processes. IEEE Transactions on Automatic Control 44, 1966–1971 (1999)
- [15] Palczewski, J., Stettner, L.: Optimisation of Portfolio Growth Rate on the Market with Fixed Plus Proportional Transaction Cost. CIS to Appear a Special Issue Dedicated to Prof. T. Duncan (2006)
- [16] Schäfer, D.: Nonparametric Estimation for Financial Investment under Log-Utility. PhD Dissertation, Mathematical Institute, University Stuttgart. Shaker Verlag, Aachen (2002)
- [17] Shreve, S.E., Soner, H.M.: Optimal Investment and Consumption with Transaction Costs. Annals of Applied Probability 4, 609–692 (1994)
- [18] Stout, W.F.: Almost Sure Convergence. Academic Press, New York (1974)
- [19] Taksar, M., Klass, M., Assaf, D.: A Diffusion Model for Optimal Portfolio Selection in the Presence of Brokerage Fees. Mathematics of Operations Research 13, 277–294 (1988)
- [20] Vega-Amaya, O.: Sample-path Average Optimality of Markov Control Processes with Strictly Unbounded Costs. Applicationes Mathematicae 26, 363–381 (1999)

Online Regret Bounds for Markov Decision Processes with Deterministic Transitions

Ronald Ortner

University of Leoben, A-8700 Leoben, Austria
ronald.ortner@unileoben.ac.at

Abstract. We consider an upper confidence bound algorithm for Markov decision processes (MDPs) with deterministic transitions. For this algorithm we derive upper bounds on the *online* regret (with respect to an (ε) -optimal policy) that are logarithmic in the number of steps taken. These bounds also match known *asymptotic* bounds for the general MDP setting. We also present corresponding lower bounds. As an application, multi-armed bandits with switching cost are considered.

1 Introduction

1.1 MDPs with Deterministic Transitions

A *Markov decision process* (MDP) can be specified as follows. First, there is a finite set S of *states* and a finite set of *actions* A such that for each state s there is a nonempty set $A(s) \subset A$ of actions that are available in s . We assume that $A(s) \cap A(s') = \emptyset$ for $s \neq s'$, and that $A = \bigcup_{s \in S} A(s)$.¹ For a state $s \in S$ and an action $a \in A(s)$, a *reward* function r gives the mean $r(s, a)$ of the random rewards for choosing a in s . We assume that these random rewards are bounded in $[0, 1]$. Further, *transition probability* distributions $p(\cdot|s, a)$ determine the probability $p(s'|s, a)$ that choosing an action a in state s leads to state s' .

A policy is a function $\pi : S \rightarrow A$ that assigns each state s a fixed action $\pi(s) \in A(s)$. The *average reward of a policy* π is defined as

$$\rho_\pi(s_0) := \lim_{t \rightarrow \infty} \frac{1}{t} \cdot r(s_t, \pi(s_t)),$$

where the process starts in s_0 , and generally, s_t is a random variable for the state at step t .

In MDPs with *deterministic* transitions, for all states s and all $a \in A(s)$ we assume that $p(s'|s, a) = 1$ for a unique $s' \in S$, while $p(s''|s, a) = 0$ for all $s'' \neq s'$. Thus each action leads deterministically from one state to another (or the same) one, so that the transition structure may be considered as a directed graph (loops allowed) with vertex set S and edge set $\bigcup_{s \in S} A(s) = A$. As we assume that the action sets $A(s)$ are pairwise disjoint, the mean reward depends

¹ Actually, it is more usual to assume that the sets $A(s)$ coincide for all states s , yet for our purposes it is more useful to consider distinct action sets.

only on the action (or edge²) in this *transition (di)graph*, so that we will write $r(a)$ for the mean reward of edge a . Thus summarizing, a deterministic MDP may be considered as a directed graph where the edges are labeled with the respective mean rewards.

We introduce some terminology from graph theory. Given a graph with vertex set V and a set $E \subseteq V^2$ of directed edges, an edge $(v, v') \in E$ is said to *start* in its *initial vertex* v and *end* in its *terminal vertex* v' . We also say that (v, v') is an *outgoing edge of* v . A (directed) *path* is a sequence of edges $e_1, e_2, \dots, e_\ell$ such that for $2 \leq i \leq \ell$ the edge e_i starts in the same vertex in which edge e_{i-1} ends. Such a path is called a (directed) *cycle*, if the initial vertex of e_1 is identical to the terminal vertex of e_ℓ . Paths and cycles are called *simple*, if the initial vertices of all edges are pairwise distinct. In the following, we will often sloppily identify a simple cycle with the set of its edges.

As we assumed that $A(s) \neq \emptyset$ for all $s \in S$, each state has at least one outgoing edge, so that playing an arbitrary but fixed policy π eventually leads into a directed simple cycle $a_1^\pi, a_2^\pi, \dots, a_\ell^\pi$. A policy may induce more than one such cycle, and the cycle that is eventually reached depends on the initial state. Generally, any policy π will partition the edge set A into one or more cycles and a (possible empty) set of *transient* edges not contained in any cycle. Starting in a transient edge a leads to a cycle, so that each edge can uniquely be assigned to an induced cycle. Consequently, depending on the initial state s_0 , the average reward ρ_π of a policy π can be written as $\rho_\pi(s_0) = \frac{1}{\ell} \sum_{i=1}^{\ell} r(a_i^\pi)$, where $a_1^\pi, a_2^\pi, \dots, a_\ell^\pi$ is the respective induced cycle of π . We are interested in the optimal policy π^* that gives maximal reward ρ^* ,³ which basically means that we are looking for a cycle with maximal mean reward.

The first algorithm for finding the optimal cycle mean has been suggested by Karp [2]. His algorithm has run-time $O(|A||S|)$. As the run-time of Karp's algorithm is also $\Omega(|A||S|)$, other algorithms have been proposed [3, 4, 5, 6] which in some cases are faster. Value iteration on deterministic MDPs has been studied as well [7].

We consider the learning setting when the MDP is not known, and a learner can only observe her current state and the actions she may choose in this state. As a measure how well a learning algorithm works, we consider its *regret* after a finite number of T steps with respect to an optimal policy, defined as

$$R_T := T\rho^* - \sum_{t=1}^T r_t,$$

where r_t is the random reward received at step t . When the learner does not compete with the optimal average reward ρ^* but only with $\rho^* - \varepsilon$ for some $\varepsilon > 0$, one considers the *regret* R_T^ε with respect to an ε -optimal policy.

Note that if the transition graph of the MDP is not *strongly connected*⁴, the achievable optimal reward ρ^* will depend on the initial state (as the optimal

² In the following, we will use the terms *action* and *edge* synonymously.

³ It can be shown that allowing time-dependent policies does not increase the achievable maximal reward. This also holds in the general MDP setting (see [1]).

⁴ A digraph is called *strongly connected* if there is a directed path between any two vertices.

cycle may not be reachable from each initial state). Even if the learner may in principle reach an optimal cycle from her initial state, as she has first to explore the transition structure of the MDP, choosing a wrong action may lead into a suboptimal part of the state space that cannot be left anymore. In this case it seems fair to compete at each step with the optimal reward achievable in the strongly connected part containing the learner's current state.⁵ As we assume deterministic transitions, any learner that explores all actions (which obviously is necessary) will eventually reach a strongly connected part that cannot be left anymore. Since our proposed learning algorithm will have explored all actions after at most $|S||A|$ steps (see Proposition 1 below), in the following we may simply assume that the transition graph is strongly connected, so that ρ^* depends only on the MDP, and we may sloppily identify optimal policies with optimal cycles. The additional regret in the general case is at most $|S||A|$.

1.2 General Remarks

After exploring the transition structure, the remaining problem is to deal with the exploitation-exploration problem concerning the rewards. The situation is similar to a multi-armed bandit problem. However, dealing with deterministic MDPs that way does not give any satisfying bounds, as in general the number of cycles is exponential in $|S|$. In the following, we present an algorithm (a simple generalization of the UCB1 algorithm of Auer et al. [9]) that achieves logarithmic regret in the number of steps taken. More precisely, after T steps the regret is $O\left(\frac{\lambda|A|\log T}{\Delta}\right)$ for an MDP dependent parameter $\lambda \leq |S|$ and a gap of Δ between ρ^* and the second-best average reward of a cycle. Apart from the parameter λ , this bound corresponds to the bound in the original bandit setting as given in [9].

On the other hand, there are logarithmic regret bounds for the general (average reward) MDP setting as well. These bounds usually hold under the assumption that the MDP is *ergodic*, i.e., any two states are connected by *any* policy.⁶ The first of these bounds due to Burnetas and Katehakis [10] was recently generalized by Tewari and Bartlett [11]. This latter bound is of order⁷ $O\left(\frac{\kappa_1^2|A||S|\log T}{\Delta}\right)$ for an MDP dependent parameter κ_1 , but — as the original bound of [10] — it holds only asymptotically. Finite horizon bounds have been achieved in [12]. However, as the bound is $O\left(\frac{\kappa_2\kappa_3^2|A||S|^5\log T}{\Delta^2}\right)$ with MDP dependent parameters $\kappa_2 > \kappa_1$ and $\kappa_3 < \kappa_2$, the dependence on the parameters is worse than in the

⁵ This basically has been suggested as one possible approach for learning in multichain MDPs in [8]. By the way, the alternative suggestion of [8] to compete with $\min_s \rho^*(s)$, where $\rho^*(s)$ is the highest achievable reward when starting in s , seems to be too weak. A lucky learner may reach a part of the MDP in which the reward is larger than $\min_s \rho^*(s)$ for *any* policy. In that case, it seems to be more natural to compete with the highest reward achievable *in that part*.

⁶ Note that this assumption does not hold in our setting.

⁷ In these bounds for general MDPs, A is the set of actions available in each state, so that the $|A|$ in our bound corresponds rather to $|S||A|$ in the general MDP setting.

bounds of [11].⁸ Here we achieve finite horizon bounds that basically correspond to the bounds of [11] in the simpler setting of deterministic MDPs, yet without the ergodicity assumption.

1.3 Outline

We proceed by introducing the upper confidence bound algorithm UCYCLE for the deterministic MDP setting. In Section 3, we prove a logarithmic bound on the expected regret of UCYCLE and complement it with a bound that holds with high probability. Lower bounds are derived as well. Finally, in Section 4 we consider the setting of multi-armed bandits with switching cost as a special case of deterministic MDPs.

2 An Upper Confidence Bound Algorithm

As algorithm for the deterministic MDP setting we suggest a simple adaptation of known upper confidence bound algorithms such as UCB1 [9] (for multi-armed bandits) or UCRL [12] (for ergodic MDPs). The common idea of such algorithms is to choose an optimal policy in an optimistic but plausible model of the situation, where plausibility is represented by confidence intervals for the estimated parameters (rewards, transition probabilities) of the system.

In the case of deterministic MDPs, the upper confidence bound strategy will be applied only to the rewards. As the transitions are assumed to be deterministic (and the learner is aware of this fact), they can easily be determined with certainty. Thus, our suggested algorithm UCYCLE first investigates the transition structure of the MDP by playing each available action in each state once. Then an upper confidence bound strategy is applied to the rewards associated with each action in order to determine the cycle $\tilde{C}$ with the highest average plausible reward. Here again, plausibility means that the reward is contained in some suitable confidence interval. The optimal cycle can be computed efficiently by any of the algorithms from the literature mentioned in the introduction. After computing the optimal cycle $\tilde{C}$, the algorithm chooses the shortest route to a state in $\tilde{C}$ and remains in $\tilde{C}$ for an appropriate number of time steps (cf. discussion below). The algorithm is depicted in Figure 1 in detail.

Note that UCYCLE proceeds in episodes of increasing length. In fact, it is a tempting but bad idea to switch the cycle whenever another cycle looks more promising. The following example demonstrates that there are very simple cases where this strategy leads to linear regret.

⁸ In fact, the exponent of $|S|$ in the bounds of [12] can be reduced by using the perturbation bounds of [13] (as applied e.g. in [14]) instead of those given in [15]. Moreover the exponent of the “gap” in the denominator can be reduced as well by using the “fillet” technique demonstrated in the proof of Theorem 2 below. Still, the improved bound of $O\left(\frac{\kappa_2 \kappa_3^2 |A| |S|^3 \log T}{\Delta}\right)$ usually remains worse than that of [11].

Input: A confidence parameter $\delta \in (0, 1)$.

Initialization and determination of transition structure:

While some action has not been sampled yet **do**:

▷ **If** there is an unsampled action $a \in A(s)$ in the current state s , choose a .

▷ **Otherwise** choose shortest path to a state s' that has already been visited before and in which there is an unsampled action.⁹

For episodes $m = 1, 2, \dots$ **do**

▷ *Calculate optimal cycle for episode m :*

Determine an optimal cycle $\tilde{C}_m$ in the deterministic MDP with transition structure as determined and whose rewards for an action a are given by

$$\tilde{r}_t(a) := \hat{r}_t(a) + \sqrt{\frac{\log \frac{|A|t^4}{\delta}}{2n_t(a)}}, \quad (1)$$

where t is the overall number of steps taken so far, $\hat{r}_t(a)$ is the average reward obtained from action a , and $n_t(a)$ is the number of times action a was chosen.

▷ *Transition phase:* Take the shortest path to a state in the cycle $\tilde{C}_m$.

▷ *Cycle phase:* Play each action in $\tilde{C}_m$ for the next $\min_{a \in \tilde{C}_m} n_t(a)$ time steps,¹⁰ when episode m is terminated.

Fig. 1. The UCYCLE algorithm

Example 1. Consider the MDP shown in Figure 2, where not only the transitions but also all the rewards are assumed to be deterministic. There are obviously two optimal cycles, viz. the loops in each of the two states with optimal average reward of $\frac{1}{2}$. If we would take our upper confidence bound approach and choose the better loop at each step, then each loop would be played only twice, before the other loop has a higher upper confidence bound (due to the larger confidence interval). As switching (which hence happens each third step) gives no reward, the average reward after T steps will be at most $\frac{2}{3} \cdot \frac{1}{2}T = \frac{1}{3}T$, so that the regret

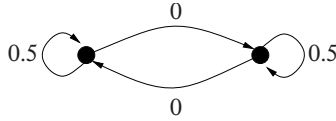


Fig. 2. Switching policies too often may lead to linear regret

⁹ The first condition guarantees that the learner need not know the state space in advance. Note that due to the condition of strong connectivity, the transition graph will be completely determined as soon as there is no such state s' . That way, only unsampled actions in the current and already visited states need to be considered in the loop, so that it is not necessary that the learner knows the number of actions in advance either.

¹⁰ That way, each action in the selected cycle $\tilde{C}_m$ is chosen the same number of times.

of this strategy is $\Omega(T)$. Note that our UCYCLE algorithm also keeps switching between the two optimal loops, but the number of switches is $O(\log T)$.

3 Online Regret Bounds

3.1 Logarithmic Upper Bounds

We start with some properties of UCYCLE. First, we bound the number of episodes and the time spent on the determination of the transition graph. Then we bound the probability that at a given step t a confidence interval fails.

Proposition 1. *It takes at most $|A||S|$ steps until UCYCLE has determined the transition structure of any deterministic MDP.*

Proof. It takes at most $|S| - 1$ steps to reach a state in which there is an action to explore, and playing that action takes another step. Thus, $|A|$ distinct actions can be explored in at most $|A||S|$ steps. $\square$

While this bound is not sharp for arbitrary $|S|$, it is easy to give examples where it is sharp at least asymptotically (for arbitrary $|S|$ and $|A| \rightarrow \infty$).

Proposition 2. *The number of episodes up to some step $T > |A|$ is upper bounded by $|A| \log_2 \frac{2T}{|A|}$.*

Proof. First note that a new episode starts only after the number of visits in one edge $a \in A$ has been doubled. Let M_a be the number of episodes which ended after the number of visits in a has been doubled,¹¹ and let T_a be the number of steps in these episodes. It is easy to see that $M_a \leq 1 + \log_2 T_a = \log_2 2T_a$. Further, $\sum_a \log_2 2T_a$ is maximal under the constraint $\sum_a T_a = T$ when $T_a = \frac{T}{|A|}$ for all a , which gives the claimed bound. $\square$

Lemma 1. *At each step t , the probability that some true mean reward is larger than the upper confidence bound value given in (1) is at most $\frac{\delta}{t^3}$, that is,*

$$\mathbb{P}\{\tilde{r}_t(a) \leq r(a) \text{ for some } a \in A\} < \frac{\delta}{t^3}.$$

Proof. For fixed $a \in A$ and $n(a) \leq t$, a standard Chernoff-Hoeffding bound (see e.g. Fact 1 in [9]) shows that

$$\mathbb{P}\left\{\hat{r}_t(a) + \sqrt{\frac{\log \frac{|A|t^4}{\delta}}{2n_t(a)}} \leq r(a)\right\} < \frac{\delta}{|A|t^4}.$$

A union bound over all actions in A and all possible values of $n(a)$ proves the lemma. $\square$

¹¹ Actually, it may happen that in an episode the number of visits is doubled in more than one edge. We assume that M_a counts only the episodes where a is the first edge for which this happens.

The error bound of Lemma 1 allows to derive the following sample complexity bound on the number of steps taken in suboptimal cycles.

Theorem 1. *The number of steps up to step T which UCYCLE in the cycle phase spends in cycles whose average reward is smaller than $\rho^* - \varepsilon$ is upper bounded by*

$$\frac{6\lambda|A|\log\frac{|A|T^4}{\delta}}{\varepsilon^2}$$

with probability at least $1 - \frac{5}{2}\delta$, where λ is the length of the largest simple cycle in the transition graph of the MDP.

Proof. Let C^* be an optimal cycle in the MDP, and let $\tilde{C}_m$ be the cycle chosen by UCYCLE in the current episode m . Denote the average reward of a cycle C in the original MDP (with the real rewards) with $\rho(C)$ and its average reward in the optimistic MDP (with rewards $\tilde{r}$) with $\tilde{\rho}(C)$. We assume throughout the proof that the confidence intervals given in (1) hold for all t , which by Lemma 1 is true with probability at least $1 - \sum_t \frac{\delta}{t^3} > 1 - \frac{5}{4}\delta$. Then

$$\rho^* - \rho(\tilde{C}_m) = \rho(C^*) - \rho(\tilde{C}_m) \leq \tilde{\rho}(C^*) - \rho(\tilde{C}_m) \leq \tilde{\rho}(\tilde{C}_m) - \rho(\tilde{C}_m)$$

with probability $1 - \frac{5}{4}\delta$. Thus, if at the beginning of an episode the estimation error $\tilde{\rho}(\tilde{C}_m) - \rho(\tilde{C}_m)$ is upper bounded by ε , UCYCLE will choose an ε -optimal cycle. This will happen in particular if $\tilde{r}_t(a) - r(a) < \varepsilon$ for all actions $a \in A$. On the other hand, this means that whenever UCYCLE chooses a cycle $\tilde{C}_m$ for which $\rho(\tilde{C}_m) < \rho^* - \varepsilon$, then there is an edge a in $\tilde{C}_m$ for which $\tilde{r}_t(a) - r(a) \geq \varepsilon$ at the initial step t of episode m . Now when each edge a was visited sufficiently often, that is, when for all $a \in A$

$$n_t(a) > \frac{2\log\frac{|A|T^4}{\delta}}{\varepsilon^2}, \quad (2)$$

then $\tilde{r}_t(a) - \hat{r}_t(a) < \frac{\varepsilon}{2}$ and (by a Chernoff-Hoeffding bound analogously to Lemma 1) also $\hat{r}_t(a) - r(a) < \frac{\varepsilon}{2}$ for all a , each with error probability at most $\frac{5}{4}\delta$. Hence, in this case $\tilde{r}_t(a) - r(a) < \varepsilon$ with probability $1 - \frac{5}{2}\delta$, and the chosen cycle is ε -optimal. We are going to determine how many steps in ε -suboptimal cycles are taken at most, until (2) holds for all actions a .

If UCYCLE chooses an ε -suboptimal cycle of length $|\tilde{C}_m|$ in an episode m , then each edge is visited exactly $\frac{\tau_m}{|\tilde{C}_m|}$ times, where τ_m is the length of episode m . For a fixed action a , let $M(a)$ be the number of episodes i in which a is part of the chosen, ε -suboptimal cycle $\tilde{C}_i(a)$, and (2) does not hold for a at the beginning of the episode. Further, let $\tau_i(a)$ be the length of the i -th respective episode. Then denoting the largest simple cycle length in the MDP by λ , we have that after the last step t' of episode $M(a) - 1$,

$$\sum_{i=1}^{M(a)-1} \frac{\tau_i(a)}{\lambda} \leq \sum_{i=1}^{M(a)-1} \frac{\tau_i(a)}{|\tilde{C}_i(a)|} \leq n_{t'}(a) \leq \frac{2\log\frac{|A|T^4}{\delta}}{\varepsilon^2}. \quad (3)$$

Within the $M(a)$ -th episode, a is finally visited sufficiently often so that (2) holds. Thus at the first step t'' of this final episode (note that a may have been played in the meantime in an optimal episode or in a transition phase)

$$n_{t''}(a) \leq \frac{2 \log \frac{|A|T^4}{\delta}}{\varepsilon^2}.$$

By the criterion when an episode ends, the number of visits in a (and indeed in all other edges as well) in this final episode can be upper bounded by $2 \cdot \frac{2 \log(|A|T^4/\delta)}{\varepsilon^2}$, so that together with (3),

$$\sum_{i=1}^{M(a)} \frac{\tau_i(a)}{\lambda} \leq \sum_{i=1}^{M(a)-1} \frac{\tau_i(a)}{\lambda} + \frac{4 \log \frac{|A|T^4}{\delta}}{\varepsilon^2} \leq \frac{6 \log \frac{|A|T^4}{\delta}}{\varepsilon^2}.$$

Consequently,

$$\sum_{i=1}^{M(a)} \tau_i(a) \leq \frac{6\lambda \log \frac{|A|T^4}{\delta}}{\varepsilon^2},$$

and summing over all actions $a \in A$ finishes the proof. $\square$

Together with Proposition 2, Theorem 1 is sufficient to yield a *high probability* bound on the regret (Theorem 3 below). For the following bound on the *expected* regret we deal with the error probabilities in a slightly more sophisticated way.

Theorem 2. *The expected regret of UCYCLE after T steps with respect to an ε -optimal policy can be upper bounded as*

$$\mathbb{E}(R_T^\varepsilon) \leq \frac{48\lambda|A| \log \frac{|A|T^4}{\delta}}{\varepsilon} + (D + \frac{10}{3}\delta)|A| \log_2 \frac{2T}{|A|} + |S||A|,$$

where λ is the largest simple cycle length and D the diameter of the transition graph, i.e. the length of the longest simple path between two vertices.

Proof. First, according to Proposition 2, the regret accumulated in the transition phases caused by switching from one cycle to another one can be upper bounded by $D|A| \log_2 \frac{2T}{|A|}$, using that by assumption at each step we suffer a loss of at most $\rho^* \leq 1$.

For the cycle phases, Theorem 1 bounds the number of steps taken in ε -suboptimal cycles with high probability. Note that the expected regret accumulated in a cycle phase of length τ when $\tilde{C}_m$ is an ε -optimal cycle is at most $\tau\varepsilon$ (this is due to the fact that episodes end only after all edges in the cycle have been visited equally often). Now we fix an $\varepsilon > 0$ and partition all suboptimal episodes¹² with respect to their expected regret:¹³ we summarize all episodes

¹² When speaking of an (ε) -suboptimal episode m we mean that the respective chosen cycle $\tilde{C}_m$ is (ε) -suboptimal.

¹³ The following technique was suggested to me by Peter Auer.

whose expected regret in the cycle phase is in the same interval $[2^{-i}, 2^{-i+1})$. For each $\varepsilon' \in [2^{-i}, 2^{-i+1})$, the number of steps in ε' -suboptimal cycles is upper bounded by $\frac{6\lambda|A|\log(|A|T^4/\delta)}{\varepsilon'^2}$ according to Theorem 1,¹⁴ so that the expected regret in the cycle phases of these episodes is upper bounded by

$$\frac{6\lambda|A|\log \frac{|A|T^4}{\delta}}{2^{-2i}} \cdot 2^{-i+1}.$$

Let $k \in \mathbb{N}$ be such that $2^{-k} \leq \varepsilon < 2^{-k+1}$. Then summing up over all $i = 0, \dots, k$ allows to upper bound the regret by

$$\sum_{i=0}^k \frac{6\lambda|A|\log \frac{|A|T^4}{\delta}}{2^{-2i}} \cdot 2^{-i+1} < 12\lambda|A|\log \left(\frac{|A|T^4}{\delta} \right) 2^{k+1} \leq \frac{48\lambda|A|\log \frac{|A|T^4}{\delta}}{\varepsilon}.$$

Finally, we have to consider the error probability with which the confidence intervals do not hold. Writing t_m for the beginning of the m -th episode, the regret for a failing confidence interval at t_m is at most $(t_{m+1} - t_m) \leq 2t_m$ (this inequality holds due to the episode termination criterion). Hence, by Lemma 1 and the bound on the number M of episodes of Proposition 2, the expected regret accumulated due to failing confidence intervals is at most

$$\begin{aligned} & 2 \sum_{m=1}^M t_m \cdot \mathbb{P}\{\text{confidence interval fails at } t_m\} \\ & \leq 2 \sum_{m=1}^M \sum_{t=1}^T t \cdot \mathbb{P}\{t_m = t \text{ and confidence interval fails at } t\} \\ & \leq 2 \sum_{m=1}^M \sum_{t=1}^T t \cdot \frac{\delta}{t^3} = 2 \sum_{m=1}^M \sum_{t=1}^T \frac{\delta}{t^2} < 2 \sum_{m=1}^M \frac{5}{3} \delta \leq \frac{10}{3} \delta |A| \log_2 \frac{2T}{|A|}. \end{aligned}$$

Summarizing we obtain

$$\mathbb{E}(R_T^\varepsilon) \leq \frac{48\lambda|A|\log \frac{|A|T^4}{\delta}}{\varepsilon} + (D + \frac{10}{3}\delta)|A|\log_2 \frac{2T}{|A|} + |S||A|,$$

now also taking into account the regret caused in the exploration phase of the transition structure according to Proposition 1. $\square$

In order to obtain high probability bounds on the regret from Theorem 1, we have to consider deviations from the average reward in each cycle.

Theorem 3. *With probability $1 - \frac{9}{2}\delta$, the regret of UCycle with respect to an ε -optimal policy after T steps can be upper bounded as*

$$R_T^\varepsilon \leq \frac{96\lambda|A|\log \frac{|A|T^4}{\delta}}{\varepsilon} + D|A|\log_2 \frac{2T}{|A|} + |S||A| + \frac{16\lambda|A|\log \frac{|A|}{\delta}}{\varepsilon}.$$

¹⁴ Actually, we do not use Theorem 1 itself, but rather refer to its proof, as we deal slightly differently with the error probabilities here.

Proof. We basically repeat the proof of Theorem 2, but in order to achieve high probability bounds on the regret with respect to an ε -optimal cycle, we consider $\frac{\varepsilon}{2}$ -optimal cycles and reserve $\frac{\varepsilon}{2}$ for the deviation from the average reward. Thus, another application of Chernoff-Hoeffding shows that in a cycle phase of length τ the probability that the random average reward is worse than the expected average reward minus $\frac{\varepsilon}{2}$ can be upper bounded by $\exp\left(-\frac{\varepsilon\tau}{2}\right)$. Now we book all episodes that are shorter than $\tau_0 := \frac{2\log(|A|/\delta)}{\varepsilon}$ (which corresponds to error probability $\frac{\delta}{|A|}$) as having maximal possible regret. Similarly to Proposition 2, the number of episodes of length $< \tau_0$ in which the number of visits of a fixed action a is doubled (cf. footnote 11) can be upper bounded by $\log_2 2\tau_0$. By the criterion for episode termination (first, visits in an action are doubled, then the cycle is completed), we may upper bound the total number of steps taken in these short episodes (and consequently also the respective regret) by

$$|A| \sum_{i=0}^{\lceil \log_2 2\tau_0 \rceil} \lambda \cdot 2^i \leq 8\lambda|A|\tau_0 = \frac{16\lambda|A|\log \frac{|A|}{\delta}}{\varepsilon}. \quad (4)$$

Similarly, the error probabilities of all longer episodes can be (by the doubling criterion for episode termination) summed up and bounded by

$$|A| \sum_{i=0}^{\lceil \log_2 \frac{2T}{|A|} \rceil} \exp\left(-\frac{\varepsilon 2^i \tau_0}{2}\right) = |A| \sum_{i=0}^{\lceil \log_2 \frac{2T}{|A|} \rceil} \left(\frac{\delta}{|A|}\right)^{2^i} < 2\delta.$$

The rest of the proof is as for Theorem 2, only with ε replaced with $\frac{\varepsilon}{2}$ and without the error term, so that one obtains including (4) the claimed regret bound, which holds with probability $1 - \frac{9}{2}\delta$. $\square$

Note that due to the different handling of the error probabilities in the proofs of Theorems 2 and 3, Theorem 3 only makes sense for sufficiently small $\delta < \frac{2}{9}$, while Theorem 2 remains sensible also for larger values of δ .

When ε is chosen sufficiently small, any ε -optimal policy will be optimal, which yields the following corollary from Theorems 2 and 3.

Corollary 1. *Let $\Delta := \rho^* - \max_{\pi: \rho_\pi < \rho^*} \rho_\pi$ be the difference between the average reward of an optimal cycle and the average reward of the best suboptimal cycle. Then*

$$\begin{aligned} \mathbb{E}(R_T) &\leq \frac{48\lambda|A|\log \frac{|A|T^4}{\delta}}{\Delta} + \left(D + \frac{10}{3}\delta\right)|A|\log_2 \frac{2T}{|A|} + |S||A|, \quad \text{and} \\ R_T &\leq \frac{96\lambda|A|\log \frac{|A|T^4}{\delta}}{\Delta} + D|A|\log_2 \frac{2T}{|A|} + |S||A| + \frac{16\lambda|A|\log \frac{|A|}{\delta}}{\Delta}, \end{aligned}$$

the latter with probability $1 - \frac{13}{2}\delta$.

Proof. The bound on the expected regret is straightforward from Theorem 2. For the high probability bound one also has to consider episodes that are Δ -good

without being optimal (which causes additional regret with respect to an *optimal* policy). This may happen if the random reward the learner obtains for a suboptimal cycle is higher than the expected reward. However, this problem can be solved using a similar strategy as in the proof of Theorem 3. We consider $\frac{\Delta}{2}$ -optimal episodes and reserve $\frac{\Delta}{2}$ for the confidence interval of the random average reward of a suboptimal cycle. Note that this is different from what we did in the proof of Theorem 3. There we had to deal with episodes in which the performance was below the average reward of the played cycle, while here we have to consider episodes where the performance is above the average reward.

Still, the argument is symmetric to the one given in the proof of Theorem 3. We consider that episodes shorter than $\frac{2\log(|A|/\delta)}{\Delta^2}$ have maximal possible regret, while the random reward of all longer episodes is larger than their expected reward by at most $\frac{\Delta}{2}$ with a total error probability $< 2\delta$. Together with Theorem 3, this results in the claimed bound which holds with probability at least $1 - \frac{13}{2}\delta$. $\square$

3.2 Lower Bounds

There are two kinds of lower bounds (on the expected regret) in the multi-armed bandit setting (cf. Section 4.1 below). First, there is a lower bound due to Mannor and Tsitsiklis [16] of $\Omega(\frac{|B|\log T}{\Delta})$ where B is the set of given arms and Δ is the difference between the best and the second-best average reward of an arm. For the case where the reward distribution is allowed to depend on the horizon T , a lower bound of $\Omega(\sqrt{|B|T})$ has been derived in [17].

It is easy to reproduce these bounds for the deterministic MDP scenario with $|B|$ being replaced with $|A|$, when there are $|S| \geq 3$ states¹⁵ and $|A| \geq 3(|S| - 1)$ actions (i.e., edges in the transition graph). This is done simply by inflating the respective multi-armed bandit problem. Figure 3 shows the basic construction of the transition graph with $|A| = 3(|S| - 1)$. Further actions may be added in each of the states. The rewards for the loops are chosen as for the arms in the multi-armed bandit problems that give the lower bounds mentioned above.

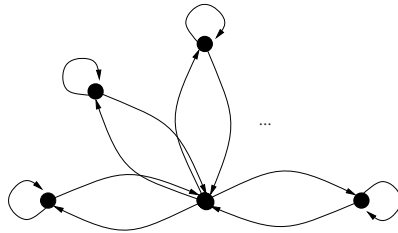


Fig. 3. The transition graph for the lower bound deterministic MDPs

¹⁵ For $|S| = 1$ one has an ordinary multi-armed bandit problem, while for $|S| = 2$ a deterministic MDP with transitions as in Figure 2 works instead of the construction given here.

All other rewards (for the transitions to different states) are set to 0. Obviously, learning such a deterministic MDP is equally hard as learning the corresponding bandit, while the regret is actually larger due to the 0-reward transitions. As the total number of edges $|A|$ is three times the number of loops $|B|$ (corresponding to the number of arms in the bandit setting), this gives the claimed lower bounds for deterministic MDPs.

4 An Application: Bandits with Switching Cost

4.1 Setting

A special case of practical relevance is the setting of (stochastic) multi-armed bandits with switching cost. In ordinary multi-armed bandit problems, a learner has to choose an *arm* from a (usually finite) set B . Choosing $b \in B$ gives random reward $\in [0, 1]$ with mean $r(b)$, and the learner tries to maximize her accumulated rewards. This corresponds to a (trivially deterministic) single state MDP.

It is a natural constraint to assume that the learner may switch arms not for free, but that she has to pay a fixed cost of $\gamma > 0$ when switching from an arm a to an arm $a' \neq a$. This can be interpreted as a negative reward of $-\gamma$ for switching arms.

Bandit settings with switching cost have mainly been considered in the economics literature (for an overview see [18]). Even though most of this literature deals with the optimization problem when the whole setting is known, there is also some work on the problem when the learner has no primary knowledge of the payoffs of the individual arms. In the wake of the seminal paper of Lai and Robbins [19], which dealt with the ordinary multi-armed bandit problem, there was an adaptation of their approach to the setting with switching costs by Agrawal et al. [20]. Their bounds later were improved by Brezzi and Lai [21]. However, as the original bounds of [19], the bounds given in [20, 21] are only *asymptotic* in the number of steps. From our results for deterministic MDPs it is easy to obtain logarithmic bounds that hold uniformly over time.

4.2 Bandits with Switching Cost as Deterministic MDPs

Translated into the deterministic MDP setting a multi-armed bandit problem with arm set B and switching cost γ corresponds to a complete digraph with $|B|$ vertices, each with loop. These loops have mean rewards according to the actions in B , while all other edges in the graph have deterministic negative reward of $-\gamma$. Note that the situation in Example 1 is an MDP corresponding to a bandit problem with switching cost. Hence, switching arms too often is also harmful in the simpler setting of bandits with switching cost.

In fact, the situation is a little bit different to the deterministic MDP setting, as in the bandit setting it is assumed that the learner knows the cost for switching. With this knowledge, it is obviously disadvantageous to choose a cycle that is not a loop in some state. Hence, a sensible adaptation of UCYCLE would choose the loop in the state that has the highest upper confidence bound value. This

corresponds to the UCB1 algorithm of Auer et al. [9] with the only difference being that increasing episodes are used (which is necessary to obtain sublinear regret as Example 1 shows). Indeed, Auer et al. [9] have already proposed an algorithm called UCB2 that also works in episodes and whose regret (including switching costs) is also logarithmic in T .

Although due to the negative switching costs, the rewards are not in $[0, 1]$, it is easy to adapt the bounds we have derived in the deterministic MDP setting. We have already argued that it is sufficient to look for optimal cycles among the loops in each state, so that λ can be chosen to be 1. Moreover, $D = 1$. However, as switching costs γ , the transition term in the bounds has to be multiplied by γ . This yields the following bounds.

Corollary 2. *The regret of UCycle in the multi-armed bandit setting with $|B|$ arms and switching cost γ can be upper bounded as*

$$\begin{aligned}\mathbb{E}(R_T) &\leq \frac{48|B| \log \frac{|B|T^4}{\delta}}{\Delta} + \left(\gamma + \frac{10}{3}\delta\right)|B| \log_2 \frac{2T}{|B|}, \quad \text{and} \\ R_T &\leq \frac{96|B| \log \frac{|B|T^4}{\delta}}{\Delta} + \gamma|B| \log_2 \frac{2T}{|B|} + \frac{16|B| \log \frac{|B|}{\delta}}{\Delta},\end{aligned}$$

the latter with probability $1 - \frac{13}{2}\delta$.

Indeed, in the bandit setting a more refined analysis is possible, so that one easily achieves bounds of the form

$$\sum_{b \in B: r(b) < r^*} \frac{\text{const} \cdot \log \frac{T}{\delta}}{r^* - r(b)} + \gamma|B| \log_2 \frac{2T}{|B|}, \quad \text{where } r^* := \max_{b \in B} r(b)$$

as given in [9] (apart from the switching cost term) by adapting the proof to the episode setting (which gives slightly worse constants in the main term than in [9]). As all this is straightforward, we neither bother to give the precise bounds nor further details concerning the proof.

Of course, the deterministic MDP setting also allows to deal with settings with individual switching costs or where switching between certain arms is not allowed. In these more general settings one trivially obtains corresponding bounds with γ replaced by the cost of the most expensive switch between any two arms. This switch need not be performed in a single step, as it may be cheaper to switch from b to b' via a sequence of other arms.¹⁶

¹⁶ Note however, that when not switching directly, the learner not only has to pay switching costs but also loses time and reward by choosing the probably suboptimal intermediate arms. There is a similar problem in the original UCycle algorithm, as taking the *shortest* path to the assumed best cycle may not be optimal. Generally, in order to solve this problem one has to consider *bias-* or *Blackwell-optimal* policies [1]. However, as this has no influence on the regret bounds, we do not consider this further.

Finally, we would like to remark that the episode strategy also works well in more general bandit settings, such as continuous bandits with Lipschitz condition. Such settings were considered e.g. in [22, 23], and it is easy to modify e.g. the proposed algorithm CAB of [22] to achieve bounds when switching costs are present. As in the settings mentioned above, the main term of the bounds remains basically the same with slightly worse constants. We note that these bounds are not logarithmic anymore and neither is the switching cost term.

5 Conclusion

Although usually there is some kind of transition (or mixing time) parameter in regret bounds for general MDPs (e.g. the κ_i in the bounds of [12, 11] mentioned above), it is not clear whether the largest simple cycle length parameter λ is necessary in regret bounds for deterministic MDPs. Interestingly, the parameter λ and the diameter D (which may be considered as an alternative transition parameter of the MDP) are in general not comparable to each other. On the one hand, complete graphs have largest possible $\lambda = |S|$ and smallest possible diameter $D = 1$. On the other hand, there are also graphs with large diameter and small λ as Figure 4 shows.

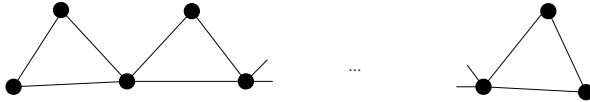


Fig. 4. Graph with $\lambda = 3$ and diameter $D = \frac{|S|-1}{2}$

A related question is what optimal bounds look like in the case of general MDPs with known transition probabilities. In particular, also in this setting it is an interesting question whether in such bounds the appearance of a transition parameter is necessary. A similar scenario has already been considered in [24]. However, in [24] the rewards are allowed to change over time, which makes learning more difficult, so that the achieved $O(\sqrt{T})$ bounds (including a transition parameter) are best possible.

Acknowledgements. The author would like to thank the reviewers for pointing out some errors and for other valuable comments. This work was supported in part by the Austrian Science Fund FWF (S9104-N13 SP4). The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements n° 216529 and n° 216886. This publication only reflects the authors' views.

References

- [1] Puterman, M.L.: Markov Decision Processes. Wiley, New York (1994)
- [2] Karp, R.M.: A characterization of the minimum cycle mean in a digraph. Discrete Math. 23(3), 309–311 (1978)

- [3] Dasdan, A., Gupta, R.: Faster maximum and minimum mean cycle algorithms for system performance analysis. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 17(10), 889–899 (1998)
- [4] Dasdan, A., Irani, S.S., Gupta, R.K.: Efficient algorithms for optimum cycle mean and optimum cost to time ratio problems. In: *Proc. 36th DAC*, pp. 37–42. ACM, New York (1999)
- [5] Hartmann, M., Orlin, J.B.: Finding minimum cost to time ratio cycles with small integral transit times. *Networks* 23(6), 567–574 (1993)
- [6] Young, N.E., Tarjan, R.E., Orlin, J.B.: Faster parametric shortest path and minimum-balance algorithms. *Networks* 21(2), 205–221 (1991)
- [7] Madani, O.: Polynomial value iteration algorithms for deterministic MDPs. In: *Proc. 18th UAI*, pp. 311–318. Morgan Kaufmann, San Francisco (2002)
- [8] Kearns, M.J., Singh, S.P.: Near-optimal reinforcement learning in polynomial time. *Mach. Learn.* 49, 209–232 (2002)
- [9] Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multi-armed bandit problem. *Mach. Learn.* 47, 235–256 (2002)
- [10] Burnetas, A.N., Katehakis, M.N.: Optimal adaptive policies for Markov decision processes. *Math. Oper. Res.* 22(1), 222–255 (1997)
- [11] Tewari, A., Bartlett, P.L.: Optimistic linear programming gives logarithmic regret for irreducible MDPs. In: *Proc. 20th NIPS* (to appear)
- [12] Auer, P., Ortner, R.: Logarithmic online regret bounds for undiscounted reinforcement learning. In: *Proc. 19th NIPS*, pp. 49–56. MIT Press, Cambridge (2006)
- [13] Hunter, J.J.: Mixing times with applications to perturbed Markov chains. *Linear Algebra Appl.* 417, 108–123 (2006)
- [14] Ortner, R.: Pseudometrics for state aggregation in average reward Markov decision processes. In: Hutter, M., Servedio, R.A., Takimoto, E. (eds.) *ALT 2007. LNCS (LNAI)*, vol. 4754, pp. 373–387. Springer, Heidelberg (2007)
- [15] Cho, G.E., Meyer, C.D.: Markov chain sensitivity measured by mean first passage times. *Linear Algebra Appl.* 316, 21–28 (2000)
- [16] Mannor, S., Tsitsiklis, J.N.: The sample complexity of exploration in the multi-armed bandit problem. *J. Mach. Learn. Res.* 5, 623–648 (2004)
- [17] Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.E.: The nonstochastic multi-armed bandit problem. *SIAM J. Comput.* 32, 48–77 (2002)
- [18] Jun, T.: A survey on the bandit problem with switching costs. *De Economist* 152, 513–541 (2004)
- [19] Lai, T., Robbins, H.: Asymptotically efficient adaptive allocation rules. *Adv. in Appl. Math.* 6, 4–22 (1985)
- [20] Agrawal, R., Hedge, M.V., Teneketzis, D.: Asymptotically efficient adaptive allocation rules for the multiarmed bandit problem with switching cost. *IEEE Trans. Automat. Control* 33(10), 899–906 (1988)
- [21] Brezzi, M., Lai, T.L.: Optimal learning and experimentation in bandit problems. *J. Econom. Dynam. Control* 27, 87–108 (2002)
- [22] Kleinberg, R.D.: Nearly tight bounds for the continuum-armed bandit problem. In: *Proc. 17th NIPS*, pp. 697–704. MIT Press, Cambridge (2004)
- [23] Auer, P., Ortner, R., Szepesvári, C.: Improved rates for the stochastic continuum-armed bandit problem. In: Bshouty, N.H., Gentile, C. (eds.) *COLT 2007. LNCS (LNAI)*, vol. 4539, pp. 454–468. Springer, Heidelberg (2007)
- [24] Even-Dar, E., Kakade, S.M., Mansour, Y.: Experts in a Markov decision process. In: *Proc. 17th NIPS*, pp. 401–408. MIT Press, Cambridge (2004)

On-Line Probability, Complexity and Randomness

Alexey Chernov¹, Alexander Shen², Nikolai Vereshchagin³, and Vladimir Vovk¹

¹ Royal Holloway, University of London, Egham, Surrey,
TW20 0EX, UK

`{chernov,vovk}@cs.rhul.ac.uk`

² LIF (Université Aix-Marseille & CNRS), Marseille and Institute
of Information Transmission Problems, Moscow

`alexander.shen@lif.univ-mrs.fr`

³ Moscow State University

`nikolay.vereshchagin@gmail.com`

Abstract. Classical probability theory considers probability distributions that assign probabilities to all events (at least in the finite case). However, there are natural situations where only part of the process is controlled by some probability distribution while for the other part we know only the set of possibilities without any probabilities assigned.

We adapt the notions of algorithmic information theory (complexity, algorithmic randomness, martingales, a priori probability) to this framework and show that many classical results are still valid.

1 On-Line Probability Distributions

Consider the following “real-life” situation. There is a tournament (say, chess or football); before each game the referee tosses a coin to decide which player will start the next game. Assuming the referee is honest, we would be surprised to learn that, say, all 100 coin tosses have produced a tail. We would be surprised also if the result of the coin tossing always turned out to be equal to some (simple) function of the results of previous games. However, it is quite possible that the results of coin tossing can be easily computed from the results of *subsequent* games. Indeed, it may well happen that the coin bit influences the results of the subsequent games and therefore can be reconstructed if these results are known.

Another similar example: if there were a rule that predicts the lucky numbers in a lottery using the previous day newspaper, we would not trust the lottery organizers. However, for the next day newspaper the situation is different (e.g., the newspaper may publish the results of the lottery).

Let X_i be the information string available before the start of i th game (say, the text of the newspaper printed just before the game) and let the bit b_i be the result of coin tossing at the start of i th game. We would like to say that for every function f and for every i the probability of the event $b_i = f(X_i)$ is $1/2$, assuming the referee is honest. And for N games the probability of the event $\forall i (b_i = f(X_i))$ equals 2^{-N} .

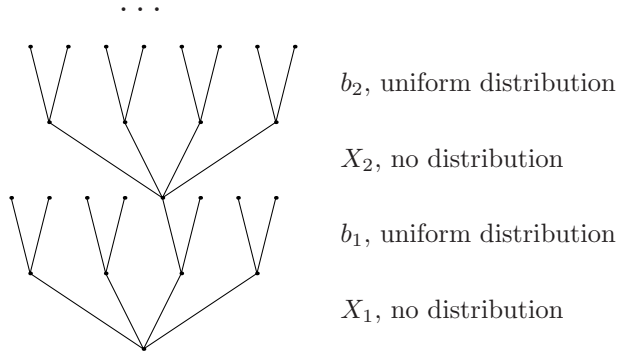


Fig. 1. The tree of possibilities

However, we cannot directly refer to classical probability theory framework in this example. Indeed, when speaking about probability of some event, one usually assumes that some probability distribution is fixed, and this distribution assigns probabilities to all possible events (at least in the finite case). In our example we do not have a probability distribution for X_i ; the only thing we have is the “conditional probability” of the event $b_i = 1$ for any condition $X_1, b_1, \dots, X_{i-1}, b_{i-1}, X_i$; this conditional probability equals $1/2$.

Formally speaking, we get a “tree of possibilities”. The sons of the root are possible values of X_1 . Each of them has two sons that correspond to two possible outcomes of the first coin tossing ($b_1 = 0$ or 1). Next level branching corresponds to the values of X_2 , then each vertex has two sons ($b_2 = 0$ or 1), etc.

In other words, tree vertices are finite sequences $(X_1, b_1, \dots, X_k, b_k)$ for even layers and $(X_1, b_1, \dots, X_k, b_k, X_{k+1})$ for odd layers; X_i are binary strings and b_i are bits. We may consider a finite tree with $2N$ layers; its leaves are sequences $(X_1, b_1, \dots, X_N, b_N)$. Or we may consider an infinite tree whose vertices are sequences of any length.

What we have is not a probability distribution but something that can be called an *on-line probability distribution* on this tree. By definition, to specify an on-line probability distribution one must fix, for each i and for all values of $X_1, b_1, \dots, X_i$, two non-negative reals with sum 1. They are called *conditional probabilities* of 0 and 1 after $X_1, b_1, \dots, X_i$ and denoted by

$$\Pr[b_i = 0 \text{ or } 1 | X_1, b_1, \dots, X_{i-1}, b_{i-1}, X_i].$$

For the case of a fair coin all these conditional probabilities are equal to $1/2$.

As usual, we can switch to unconditional probabilities (i.e., can multiply conditional probabilities on the path from the tree root). Then we arrive to the following version of the definition: an on-line probability distribution is a function P defined on tree vertices such that $P(\Lambda) = 1$ (Λ is the tree root),

$$P(X_1, b_1, \dots, X_i, b_i, X_{i+1}) = P(X_1, b_1, \dots, X_i, b_i)$$

(on vertices where no random choice is made, the function propagates without change), and

$$\begin{aligned} P(X_1, b_1, \dots, X_i, b_i, X_{i+1}) = \\ = P(X_1, b_1, \dots, X_i, b_i, X_{i+1}, 0) + P(X_1, b_1, \dots, X_i, b_i, X_{i+1}, 1). \end{aligned}$$

The intuitive meaning of $P(v)$ is the probability to arrive at v if the environment (that chooses $X_1, X_2, \dots$) wants this and makes suitable moves in its turn.

This definition makes sense both for finite and infinite trees.

Remark. A technical problem arises when some values of an on-line probability distribution are zeros: in this case conditional probabilities cannot be reconstructed from the products. However, in this case they are usually not important, so we can mostly ignore this problem.

Similar on-line probability distributions can be considered for more general trees where on the odd levels, instead of 0 and 1, we have a (countable) list of possible values of b_i .

Now let us assume that the tree is finite (has finite height and finite number of vertices on every level). Consider an event E , i.e., some set of tree leaves. We cannot define the probability of an event under a given on-line probability distribution P . However, we can define an *upper probability* of E . (It may be called a “worst case probability” if the event E is considered undesirable.) This notion can be defined in several (equivalent) ways.

Definition

(1) Consider all probability distributions on the leaves of the tree. Some of them are consistent with the given on-line probability distribution (i.e., give the same conditional probabilities for b_i when $X_1, b_1, \dots, X_i$ are given). Upper probability of E is a maximum of $\Pr[E]$ under all these distributions.

(2) Consider the following probabilistic game: a player (“adversary”, if the event is undesirable) chooses some X_1 , then b_1 is chosen at random with prescribed probabilities (condition X_1), then player chooses X_2 , then b_2 is chosen at random (according to the conditional probabilities with condition X_1, b_1, X_2), etc. The player wins if the resulting leaf belongs to the event E . The upper probability of E is the maximal probability that player wins (maximum is taken over all deterministic strategies).

(3) Let us define the *cost* of a vertex in the tree inductively starting from the leaves. For a leaf in E the cost is 1, for a leaf outside E the cost is zero. For a non-leaf vertex v where the choice of X_i is performed, the cost of v is the maximal cost of its sons; for a vertex that corresponds to the choice of b_i , the cost is the weighted sum of the sons’ costs where weights are conditional probabilities. Upper probability of E is the cost of the tree root.

(4) Let us consider *on-line martingales* with respect to P , i.e., non-negative functions V defined on tree vertices such that

$$V(X_1, b_1, \dots, X_i, b_i) = V(X_1, b_1, \dots, X_i, b_i, X_{i+1}); \quad (1)$$

$$V(X_1, b_1, \dots, X_i) = V(X_1, b_1, \dots, X_i, 0) \cdot \Pr[b_i = 0 | X_1, b_1, \dots, X_i] + \\ + V(X_1, b_1, \dots, X_i, 1) \cdot \Pr[b_i = 1 | X_1, b_1, \dots, X_i]; \quad (2)$$

these functions correspond to the player's capital in a fair game (when player observes X_i , the capital does not change; when player splits the capital between bets on $b_i = 0$ and $b_i = 1$, the winning bet is rewarded according to the conditional probabilities determined by the on-line distribution). The upper probability of E is the minimal value of $V(A)$ over all V such that $V \geq 1$ for all leaves that belong to E . In other terms, the upper probability of E is 1 divided by the fair price of the option to play such a game with initial capital 1 knowing in advance that the sequence of outcomes belongs to E .

Remark. As we have mentioned, we need some precautions for the case when some values of P are zeros, since in this case conditional probabilities are not uniquely defined. However, it is easy to see that all choices of conditional probabilities compatible with P will lead to the same value of upper probability.

Theorem 1. *All four definitions are equivalent.*

Proof. Note that player's strategy in the second definition determines a distribution on the leaves (X_i is chosen deterministically according to the strategy while b_i is chosen according to the prescribed conditional probabilities). This distribution is consistent with the given on-line distribution. So the upper probability as defined in (2) does not exceed the upper probability as defined in (1). On the other hand, any probability distribution can be considered as a mixed strategy in the game (player chooses her moves randomly using independent random bits), and the winning probability of a mixed strategy is the weighted average of the winning probabilities for pure strategies, so we get the reverse inequality. The inductive definition (3) computes the winning probability for the optimal strategy (induction on tree height).

The equivalence with the martingale definition can be proved in the same way as for the classical off-line setting (this argument goes back to Ville, see, e.g., [7]). If a martingale V starts with capital p and achieves 1 on every leaf in E , then for every probability distribution compatible with P and for every tree vertex the current value of V is an upper bound for the expectation of V if the game starts at this vertex. Therefore, $V(A)$ is an upper bound for the probability to end the game in E for every probability distribution compatible with P . The reverse inequality: the vertex cost (defined inductively) satisfies the conditions in the definition of a martingale if we replace $=$ by $\geq$ in the condition (1). Increasing this function, we can get a martingale. $\square$

Remarks

1. Note that upper probability is not additive: e.g., both the event and its negation can have upper probabilities 1, just the strategies to achieve them are different. However, it is sub-additive: the upper probability of $A \cup B$ does not exceed the sum of upper probabilities of A and B .

2. We can define *supermartingales* in the same way as martingales replacing $=$ by $\geq$ in (2). We relax the requirement (2) and not (1) since it is more natural

from the game viewpoint: getting information about X_i does not change the player's capital. It is easy to see that supermartingales may be used instead of martingales in the definition of upper probability.

3. Proving Theorem 1, we assumed that the tree is finite. However, the same argument shows that it is valid for infinite trees of finite height (and even for the trees having no infinite branches), if we use supremum instead of maximum.

Classical probability theory says that events with very small probability can be safely ignored (and when they happen, we have to reconsider our assumptions about the probability distribution). In the on-line setting we can say the same about events that have very small upper probability: believing in the probabilistic assumption, we may safely ignore the possibilities that have negligible upper probabilities, and if such an event happens, we have to reconsider the assumption.

Remarks

1. In fact upper probability (though not with this name) is used in the definition of the Arthur – Merlin class in computational complexity theory where a tree of polynomial height and a polynomially decidable event are considered and we distinguish between events of low and high upper probability.

2. It is easy to see that on-line martingales with respect to on-line probability distribution P are just the ratios Q/P where Q is some other on-line probability distribution. (Some evident precautions are needed if P can be zero somewhere.)

2 On-Line Kolmogorov Complexity KR

We can adapt the standard definition of Kolmogorov complexity (see, e.g., [3, 5] for the definition and discussion of different versions of Kolmogorov complexity) for the on-line setting. Consider a sequence $X_1, b_1, X_2, b_2, \dots, X_n, b_n$ where X_i are binary strings and b_i are bits. Look for a shortest interactive program that after getting input X_1 produces b_1 , then after getting X_2 (in addition to X_1) produces b_2 , then after getting X_3 produces b_3 etc. We call its length the *on-line decision complexity* with respect to the programming language π used, and denote it by $KR_\pi(X_1 \rightarrow b_1; X_2 \rightarrow b_2; \dots; X_n \rightarrow b_n)$. The reason for the name “decision complexity”: if all X_i are empty, we get the standard notion of decision complexity of a bit string $b_1 \dots b_n$ (the length of the shortest program that generates b_i given i). It is easy to see that a natural version of optimality theorem holds (there exists an optimal “programming language”), so the on-line decision complexity (for an optimal programming language) $KR(X_1 \rightarrow b_1; X_2 \rightarrow b_2; \dots; X_n \rightarrow b_n)$ is well defined (up to an additive $O(1)$ -term).

Theorem 2. *The on-line complexity $KR(X_1 \rightarrow b_1; \dots; X_n \rightarrow b_n)$ does not exceed the decision complexity $KR(b_1 b_2 \dots b_n)$ and is greater than the conditional complexity $KS(b_1 b_2 \dots b_n | X_1, X_2, \dots, X_n)$ up to $O(1)$ terms.*

In other terms, knowing X_i in an on-line setting may help to describe $b_1, \dots, b_n$, but knowing all X_i in advance is even better. (The proof is straight forward.)

3 On-Line A Priori Probability and KA

It is well known that Kolmogorov complexity is related to the a priori probability (maximal lower semicomputable semimeasure). The latter can be naturally defined in the on-line setting. Let us give two equivalent definitions.

Consider an interactive probabilistic machine T that has internal random bit generator. This machine gets some binary string X_1 (say, on the tape where the end of X_1 is marked by a special separator), performs a computation that uses X_1 and random bits and may produce bit b_1 (or hang). After b_1 is produced, T gets the second input string X_2 , continues its work (using fresh random bits) and may produce second output bit b_2 , etc. In other words, we write $X_1 \# X_2 \# \dots \# X_n$ on the input tape, but T cannot get access to X_i before it produces $i - 1$ output bits $b_1, \dots, b_{i-1}$.

For a given T consider a function M_T : Let $M_T(X_1, b_1, \dots, X_n, b_n)$ be the probability that T outputs $b_1, \dots, b_n$ getting $X_1, X_2, \dots, X_n$ as input (with restrictions described above). We extend the function M_T to the sequences of odd length: $M_T(X_1, b_1, \dots, X_n, b_n, X_{n+1})$ is equal to $M_T(X_1, b_1, \dots, X_n, b_n)$. We let $M_T(\Lambda) = 1$. It is easy to see that if T never hangs (or hangs with zero probability), then M_T is an on-line probability distribution. In general, M_T is an on-line semimeasure in the sense of the following

Definition. An *on-line semimeasure* is a function M that maps tree vertices to non-negative reals such that $M(\Lambda) = 1$, $M(X_1, b_1, \dots, X_i, b_i, X_{i+1}) = M(X_1, b_1, \dots, X_i, b_i)$ (on vertices where no random choice is made, the function propagates without change), and the inequality $M(X_1, b_1, \dots, X_i, b_i, X_{i+1}) \geq M(X_1, b_1, \dots, X_i, b_i, X_{i+1}, 0) + M(X_1, b_1, \dots, X_i, b_i, X_{i+1}, 1)$ holds. (We have replaced “=” by “ $\geq$ ” in the definition of an on-line probability distribution.)

It is easy to see that semimeasure M_T that corresponds to a probabilistic machine T of described type is a *lower semicomputable* function, i.e., there is an algorithm that gets its input and produces an increasing sequence of rational numbers that converges to the value of the function.

Theorem 3. *Every lower semicomputable on-line semimeasure corresponds to some probabilistic machine.*

Proof is similar to the off-line case. For an on-line semimeasure M we perform a “memory allocation”, so that for each finite sequence $X_1, b_1, \dots$ an open subset of $[0, 1]$ that has measure $M(X_1, b_1, \dots)$ is allocated. Adding X_i to the end of the sequence does not change the set; adding bits 0 and 1 replaces the corresponding set by two its disjoint subsets. (Note that these subsets may depend not only on b_i , but also on X_i .) If M is lower semicomputable, these sets can be made uniformly effectively open. Then we consider a machine T that generates a uniformly distributed random real number $\alpha \in [0, 1]$ bit by bit and generates $T(X_1, b_1, \dots, X_i) = b_i$ if the effectively open set that corresponds to $X_1, b_1, \dots, X_i, b_i$ contains α . $\square$

Theorem 4. *There exists the largest (up to $O(1)$ -factor) lower semicomputable on-line semimeasure.*

Proof. Again we can use standard trick: a universal machine first generates randomly a machine of described type in such a way that every machine appears with a positive probability, and then simulates this machine. $\square$

We call this maximal semimeasure an *on-line a priori probability* and denote it by $A(X_1, b_1, \dots, X_n, b_n)$. (If all X_i are empty strings, we get a standard a priori probability on a binary tree.) Minus logarithm of this semimeasure is called *on-line a priori complexity* and denoted by $KA(X_1 \rightarrow b_1; X_2 \rightarrow b_2; \dots; X_n \rightarrow b_n)$.

4 Relations between KR and KA

Now, when two complexities KA and KR are defined in the on-line framework, one may ask how they are connected. Their off-line versions are close to each other: it is known that $KR(x) \leq KA(x) \leq KR(x) + 2 \log KR(x)$ (up to $O(1)$ -terms) for all binary strings x .

These inequalities remain true for the on-line setting (with the same $O(1)$ -precision):

Theorem 5. $KR(\dots) \leq KA(\dots) \leq KR(\dots) + 2 \log KR(\dots)$; here “...” stands for $X_1 \rightarrow b_1; X_2 \rightarrow b_2; \dots; X_n \rightarrow b_n$.

Proof of the second inequality goes in the same way as usual; we consider a randomized algorithm that chooses machine number i with probability $1/i^2$.

The first inequality needs more care, since in the on-line case we are more restricted and need to ensure that programs are indeed on-line and do not refer to the inputs that are not yet available.

We need to allocate 2^n strings of length n to objects that have KA -complexity less than n (=have a priori probability greater than 2^{-n}). We do it inductively (first for $X_1 \rightarrow b_1$, then for $X_2 \rightarrow b_2$, etc.) and ensure a stronger requirement: if a priori probability of some object exceeds $k2^{-n}$ for some k , then there are at least k different programs of length n allocated to this object.

So we start looking at the approximations (from below) to the (a priori) probabilities of $X_1 \rightarrow 0$ and $X_1 \rightarrow 1$ (independently for each n and each X_1); when probability of $X_1 \rightarrow b_1$ exceeds $k2^{-n}$, we allocate a new (k th) program of length n that transforms X_1 to b_1 . On top of this process we look at the approximations to a priori probabilities of $X_1 \rightarrow b_1; X_2 \rightarrow b_2$ and add new programs that map X_2 to b_2 among the programs that mapped X_1 to b_1 ; we have enough programs for that since $M(X_1, b_1, X_2, 1) + M(X_1, b_1, X_2, 0) \leq M(X_1, b_1)$, so if k_1 programs are needed for the first term and k_0 are needed for the second, then there are already $k_0 + k_1$ programs allocated to $X_1 \rightarrow b_1$ to choose from. On top of that, we allocate programs for $X_1 \rightarrow b_1; X_2 \rightarrow b_2; X_3 \rightarrow b_3$ etc. $\square$

5 On-Line Randomness

Let us return to the “real-life” example and make it less real: imagine that we observe an *infinite* sequence of games and (for every i) know the bit b_i produced by the referee when i th game starts and the string X_i that is known before i th game. There are cases when we intuitively reject the fair coin assumption. Can we make the intuition more formal and define a notion “in the sequence $X_1, b_1, X_2, b_2, X_3, b_3, \dots$ the bits $b_1, b_2, \dots$ are random”? For the off-line case the most popular notion is called *Martin-Löf randomness* (ML-randomness; see [3, 6] for details). Now we want to extend it to the on-line setting.

Assume that a computable on-line probability distribution P (on the infinite tree) is fixed. Martin-Löf definition starts with a notion of an “effectively null” set. Adapting this definition to on-line setting, we need to remember that probability of events is now undefined; moreover, the notion of upper probability (that replaces it) has been defined for finite case only.

Consider the space Π of all (infinite) sequences $X_1, b_1, X_2, b_2, \dots$. A *cone* in this set is a set of all sequences with given finite prefix.

Definition. Let U be a finite union of cones. Then the *upper probability* of U with respect to P is defined as the upper probability of the corresponding event in the finite part of the tree (large enough to contain all the roots of the cones).

(It is easy to see that this probability does not change if we increase the size of the finite part of the tree. The upper probability is monotone with respect to set inclusion.)

Then we can define an on-line version of null sets.

Definition. A set $Z \subset \Pi$ is an *on-line null set* if for any $\varepsilon > 0$ there exists a sequence of cones such that: (1) the union of cones covers Z ; (2) the union of any finite number of these cones has upper probability less than ε .

Martin-Löf definition of randomness deals with effectively null sets, so our next step is to define them in an on-line setting.

Definition. A set Z is an *on-line effectively null set* if there exists an algorithm that for any given rational $\varepsilon > 0$ generates a sequence of vertices such that the corresponding cones cover Z and the union of any finite number of these cones has upper probability less than ε . (Note that we require the upper probability of the union of the cones to be small, not the sum of upper probabilities of the cones. This difference matters since upper probability, unlike classical probability, is not additive.)

Theorem 6. *There exists an on-line effectively null set that contains every other on-line effectively null set.*

Proof is similar to the off-line case. Having any algorithm that given rational $\varepsilon > 0$ generates sequences of vertices, we can “trim” it so that the union of any finite number of generated cones has upper probability less than ε . (Indeed, for a computable on-line measure the upper probability of the finite union of cones is computable, and we may quarantine new strings until they are cleared.) So

we can enumerate all the algorithms that satisfy these restrictions and then take the union of corresponding on-line effectively null sets (combining covers of size $\varepsilon/2$, $\varepsilon/4$ etc. to get the cover of size ε ; here we use the subadditivity of upper probability). $\square$

Now we can give a

Definition. Bits $b_1, b_2, \dots$ are *on-line ML-random* in a sequence $\omega = X_1, b_1, X_2, b_2, \dots$ if ω does not belong to the maximal on-line effectively null set.

In other words, $b_1, b_2, \dots$ are not random in ω if and only if $\{\omega\}$ is an on-line effectively null set (if and only if some on-line effectively null set contains ω).

6 On-Line Randomness Criterion

A classical Levin – Schnorr theorem gives a criterion of randomness in terms of complexity (in particular, a priori complexity KA) or supermartingales. Similar criterion exists for the on-line version.

Theorem 7. (Levin – Schnorr theorem, on-line version). *Assume that a computable on-line probability distribution P is fixed. Bits $b_1, b_2, \dots$ are on-line ML-random (with respect to P) in a sequence $\omega = X_1, b_1, X_2, b_2, \dots$ if and only if $KA(X_1 \rightarrow b_1; \dots, X_n \rightarrow b_n) \geq -\log_2 P(X_1, b_1, \dots, X_n, b_n) - c$ for some c and all n .*

Recalling that KA is the minus logarithm of a priori probability A , we can reformulate the criterion: bits b_i are random in $(X_1, b_1, X_2, b_2, \dots)$ if and only if the ratio $A(X_1, b_1, \dots, X_n, b_n)/P(X_1, b_1, \dots, X_n, b_n)$ has a constant upper bound. (Note that A is the maximal semimeasure and P is a measure (and therefore a semimeasure), so this ratio always has a positive lower bound.)

One more reformulation of the same result uses on-line supermartingales. As we have noted, on-line (super)martingales with respect to P are ratios Q/P where Q is an on-line (semi)measure. It allows us to reformulate the criterion as follows: bits b_i are random in a sequence $\omega = X_1, b_1, X_2, b_2, \dots$ if and only if any lower semicomputable supermartingale is bounded on prefixes of ω .

For a more advanced (and more difficult to prove) version of Theorem 7, see [8].

Proof of the on-line version of Levin – Schnorr randomness criterion follows the off-line argument with some changes: we have to be more careful since we have to deal with upper probability instead of an (additive) measure.

First, we have to show that if a sequence is not random, then the ratio A/P is unbounded on its prefixes. Since A is maximal, it is enough to construct some lower semicomputable semimeasure Q such that Q/P is unbounded.

Lemma. *Assume that some algorithm enumerates a sequence of cones $C_1, C_2, \dots$ and the upper probability of the union $C_1 \cup \dots \cup C_N$ is less than ε for some rational $\varepsilon > 0$ and for all N . Knowing this algorithm and ε , we can construct a lower semicomputable semimeasure S that exceeds P/ε at any finite sequence that belongs to one of the cones.*

Proof of the Lemma. For any vertex v let us consider the cone $C(v)$ with root v and for any N let us compute the upper probability of the intersection $C(v) \cap (C_1 \cup C_2 \cup \dots \cup C_N)$. Since P is computable, doing this for $N = 1, 2, \dots$, we get an increasing computable sequence of computable reals, and its limit is lower semicomputable. Let $S(v)$ be this limit divided by ε . This limit is not technically a semimeasure since $S(X_1, b_1, \dots, X_i, b_i)$ can be bigger than $S(X_1, b_1, \dots, X_i, b_i, X_{i+1})$. But if we increase the latter by letting $S(X_1, b_1, \dots, X_i, b_i, X_{i+1}) := S(X_1, b_1, \dots, X_i, b_i)$, and also let $S(\Lambda) = 1$, we do get a lower semicomputable semimeasure that satisfies the requirements of the Lemma. $\square$

Now we can finish the proof of the first part of Levin – Schnorr on-line randomness criterion. Let $\varepsilon_n = 2^{-2^n}$. Since ω belongs to an on-line effectively null set, we can get a sequence of cones with upper probability bounded by ε_n ; applying the Lemma to it, we get a lower semicomputable semimeasure S_n that exceeds $2^{2^n} P$ on any vertex that belongs to one of the cones. Then the sum $S = \sum_n 2^{-n} S_n$ exceeds $2^n P$ on any vertex that belongs to some of the cones generated for ε_n . By assumption ω has a prefix of this type for every n , so S/P is unbounded on prefixes of ω .

It remains to prove the second part of the theorem. For any lower semicomputable semimeasure S we have to show that the set of all sequences ω such that S/P is unbounded on the prefixes of ω is an on-line effectively null set.

For a given $\varepsilon > 0$ let us consider all the vertices where S/P exceeds $1/\varepsilon$. They can be enumerated if ε is given since S is lower semicomputable and P is computable. We need to check that the upper probability of the union of any finite number of corresponding cones is less than ε . Indeed, while computing the costs inductively in a top-down fashion, the cost is upper-bounded by ε times the value of S in the vertex. (Induction base is guaranteed by the assumption: we start with vertices where S/P is greater than $1/\varepsilon$; the induction step works since S is a semimeasure and P is a measure.) $\square$

Remarks

1. In the off-line setting the similar construction almost gives a lower semicomputable measure (with one exception: the measure of the entire space may be less than 1) or, in other terms, a martingale whose initial amount is lower semicomputable. In the on-line setting it is no more true (at least for this construction), and we get a semimeasure (or supermartingale).

2. On the other hand, the proof gives more than we claimed: if a sequence is not random, then some lower semicomputable on-line supermartingale is not only unbounded but also tends to infinity. It implies that if some lower semicomputable on-line supermartingale is unbounded on some ω , then some *other* semicomputable on-line supermartingale tends to infinity on ω .

The notion of randomness of b_i in a sequence $X_1, b_1, X_2, b_2, \dots$ lies in-between Martin-Löf randomness and Martin-Löf randomness with respect to an oracle. Indeed, it implies ML-randomness since we can consider semimeasures (supermartingales) that do not depend on X_i at all. On the other hand, each on-line

supermartingale can be transformed into a supermartingale that uses the entire sequence $X_1, X_2, \dots$ as an oracle (getting access not only to the past X_i , but also to the future ones). Both inclusions are strict for evident reasons: a ML-random sequence b_i is not on-line random if $X_i = b_i$; it is on-line random if $X_i = b_{i-1}$ but not random with oracle $X_1, X_2, \dots$.

Other observations (the proof is straightforward):

Theorem 8

(a) *If the sequence X_i is computable (or if X_i is a computable function of $X_1, b_1, \dots, X_{i-1}, b_{i-1}$) then on-line randomness is equivalent to (standard) ML-randomness with respect to induced measure where X_i are fixed.*

(b) *Changing finitely many terms among b_i or X_i does not make a random sequence non-random or vice versa, assuming that all conditional probabilities are not zeros.*

(c) *The on-line random sequence remains on-line random if we replace X_i by some Y_i that is a computable function of $b_1, X_1, \dots, b_{i-1}, X_i$.*

7 Muchnik's Paradox

In this section we consider the case of fair coin (all conditional probabilities are equal to $1/2$). Let $b_1, b_2, \dots$ be a ML-random sequence. It is easy to see that then the sequence $b_2, b_4, b_6, \dots$ is on-line ML-random if $b_1, b_3, b_5, \dots$ are used as external information. Indeed, any lower semicomputable on-line supermartingale can be transformed into a (lower semicomputable) off-line supermartingale that makes no bets on b_1, b_3, b_5 etc. For the same reason the sequence $b_1, b_3, b_5, \dots$ is on-line random inside the sequence $\Lambda, b_1, b_2, b_3, \dots$ (bits $b_2, b_4, \dots$ are treated as external information).

One may naturally expect that the reverse is also true: if both odd and even bits are unpredictable (with all previous bits used as the external information), then the entire sequence should be random. Indeed, our intuition says that if the coin tossing is performed by two referees that alternate (each of them works every second day), and both referees do their job perfectly, the resulting sequence of bits should be also perfectly random.

This would make a nice on-line version of van Lambalgen theorem that says that if a sequence $b_1, b_3, b_5, \dots$ is ML-random and at the same time $b_2, b_4, b_6, \dots$ is ML-random with oracle $b_1, b_3, b_5, \dots$, then the entire sequence $b_1, b_2, b_3, b_4, \dots$ is ML-random.

We may note also that if we replace *semicomputable* supermartingales by *computable* supermartingales, the corresponding statement becomes true. Indeed, assume that a computable supermartingale S is unbounded on some sequence. We may assume without loss of generality that it is at least 1 on every sequence (just by adding 1). At each vertex it splits the current capital in computable proportions (since the ratio of two computable numbers separated from zero is uniformly computable). So we can consider two computable supermartingales S_1 and S_2 ; one does not make any bet on odd steps and follows the proportions of S at the even steps, the other does the opposite. Then the capital of S is the

product of S_1 and S_2 , so if S is unbounded on some sequence, then at least one of S_1 and S_2 is unbounded on it.

However, all these arguments do not make the desired statement true, as An. Muchnik [4] has shown. He showed that there is a sequence which is not ML-random but still both odd and even terms are on-line random. This construction is rather delicate and we do not explain it here.

8 Selection Rules and On-Line Randomness

The classical definition of randomness (for the case of independent fair coin tossing) suggested by R. von Mises is based on selection rules: each subsequence that is selected by an “admissible selection rule” should have limit frequency $1/2$. It can also be naturally transferred into the on-line framework.

In Mises – Church definition of randomness an admissible selection rule is a total computable function that can be applied to any binary string and produces one of two answers: S (“selected”) or O (“observed”). An application of this rule to a sequence ω goes as follows: the value of selection rule on n -bit prefix of ω determines if the next bit should be selected or just observed. Another version that goes back to R.P. Daley considers partial functions as selection rules; if such a function is undefined at some prefix of ω , then the selection process hangs and the selected subsequence is finite.

Both definitions (with total and partial selection rules) can be easily extended to the on-line framework. We just allow the selection rule to use the external information that is available at the moment (i.e., all previous values of X_i). It is easy to see that we get a weaker notion of randomness (compared to on-line Martin-Löf randomness with respect to the uniform Bernoulli on-line measure, i.e., the measure where all conditional probabilities are equal to $1/2$). Moreover, the following is true:

Theorem 9. *Assume that bits b_i are on-line ML-random (with respect to the uniform Bernoulli on-line measure) in a sequence $X_1, b_1, X_2, b_2, \dots$ and a (partial) selection rule S is given that is defined on all prefixes of this sequence and selects infinitely many bits $b_{i_1}, b_{i_2}, \dots$ for increasing sequence of indices $i_1 < i_2 < \dots$. Then the selected bits are on-line ML-random in a sequence $Y_1, b_{i_1}, Y_2, b_{i_2}, \dots$ where Y_k is the prefix of the original sequence (both X_i and b_i) that precedes b_{i_k} . (This implies, as we have said, that b_{i_k} are ML-random and therefore satisfy the strong law of large numbers.)*

Proof. Indeed, a semicomputable on-line supermartingale U that plays with b_{i_k} using information that precedes them can be transformed into a supermartingale that deals with the original sequence. While selection rule is not yet defined, the supermartingale does not bet anything; if selection rule says “observe”, the supermartingale keeps the same capital not making any bets; if the selection rule says “select”, the supermartingale follows U . $\square$

9 Randomness with Respect to Classes of Measures

On-line randomness is connected with the notion of randomness with respect to effectively closed classes of measures; this notion was introduced by Levin [2] (see [1] for the detailed exposition).

Consider some class $\mathcal{S}$ of measures (probability distributions) on the Cantor space Ω of infinite sequences of zeros and ones. A set $Z \subset \Omega$ is called a $\mathcal{S}$ -null set if $P(Z) = 0$ for every $P \in \mathcal{S}$. The effective version of this definition: $Z \subset \Omega$ is an *effectively $\mathcal{S}$ -null set* if there is an algorithm that given rational $\varepsilon > 0$ produces a sequence of intervals $I_1, I_2, \dots \subset \Omega$ that covers Z such that $P(I_1 \cup I_2 \cup \dots) \leq \varepsilon$ for every $P \in \mathcal{S}$.

Levin noted that for an effectively closed class $\mathcal{S}$ the union of all effectively $\mathcal{S}$ -null set is an effectively $\mathcal{S}$ -null set. (An effectively closed class $\mathcal{S}$ is defined in a natural way: we consider a topology on the set of all measures where basic open set are the sets $\mathcal{U}_{x,p,q} = \{P | P(\Omega_x) \in (p, q)\}$ for all binary strings x and all intervals (p, q) with rational endpoints, as well as their finite intersections. A class $\mathcal{S}$ is effectively closed if its complement is a union of a computable sequence of basic open sets.) Then we say that a sequence ω is *random with respect to $\mathcal{S}$* if it does not belong to largest effectively $\mathcal{S}$ -null set.

Now we can relate the definition of the on-line randomness to Levin's definition. Consider the class $\mathcal{S}$ of measures P that are consistent with the given online distribution, i.e., have conditional probabilities $1/2$ at odd layers: $P(b_1 b_2 \dots b_{2n+1} 0) = P(b_1 b_2 \dots b_{2n+1} 1)$ for every bit string $b_1 \dots b_{2n+1}$ of odd length. It is easy to see that randomness with respect to $\mathcal{S}$ is equivalent to the on-line randomness. (The strings X_i in the definition of the on-line randomness are replaced by bits for simplicity, but this is not essential.)

10 On-Line and Prequential Randomness

The classical definition of ML-randomness with respect to a computable measure P does not really use the ordering of the bits in the sequence: if we perform a computable permutation of a sequence and change the measure accordingly, then the sequence remains random. However, for the case when the bits are generated sequentially, the standard Martin-Löf definition does not look natural. Indeed, according to the definition, we need to know the measure of any interval in the Cantor space, including the intervals that can not contain the sequence in question. If our sequence starts with, say, 01001, it seems that the value $P(\Gamma_{100})$, where Γ_z is a set of all continuations of a finite string z , should be totally irrelevant. The only thing that seems to be relevant in this sequential setting (when a sequence ω is generated bit by bit from left to right) is the values of conditional probabilities of 0 and 1 after $\omega_1 \omega_2 \dots \omega_{n-1}$, i.e., the ratios $P(\Gamma_{x0})/P(\Gamma_x)$ and $P(\Gamma_{x1})/P(\Gamma_x)$ for all x that are prefixes of ω .

This intuition is supported by the following observation: let P and P' be two computable measures that have the same conditional probabilities along some sequence ω (as defined above). If ω is ML-random with respect to P , then it

is ML-random with respect to P' . (This is an immediate corollary of Levin – Schnorr randomness criterion.)

So we come to a natural question: can we give a natural definition of randomness in such a way that only conditional probabilities along ω are used in the definition?

Imagine an adjustable random bit generator, i.e., a device for generating random bits with prescribed probabilities. Such a device gets some real number $p \in [0, 1]$ as an input and generates a bit $b \in \{0, 1\}$ claiming that this bit is a result of a “fresh” random experiment (independent of all the past information) with probability of success p . Then we can send the next probability value to the device, it generates the next random bit, and so on.

Assume that we observe the behavior of the device and have its work recorded. The record (protocol) is a sequence $p_1, b_1, p_2, b_2, \dots$ where $p_i \in [0, 1]$ and $b_i \in \{0, 1\}$. Can we say whether this random bit generator works properly or not looking at this protocol? Our intuition says that there are at least some cases when we don't trust such a generator. For example, if all p_i are greater than $1/2$ but vast majority of b_i are zeros, it is clear that something is wrong with the device. Similarly, if all p_i are, say, between $1/3$ and $2/3$ while $b_1 b_2 b_3 \dots = 010101 \dots$ (alternating zeros and ones), again we would not trust the generator.

Let us restrict ourselves to the case when all p_i are rational numbers in $(0, 1)$. Then we can define the randomness of the sequence $p_1, b_1, p_2, b_2, \dots$ in the following way. Consider an on-line probability distribution on sequences $X_1, b_1, X_2, b_2, \dots$ where strings X_i are identified with rational numbers p_i in $(0, 1)$ using some computable one-to-one correspondence, and the conditional probability $\Pr[b_i = 1 | X_1, b_1, \dots, X_i]$ equals p_i (where p_i is the rational number that corresponds to X_i). This on-line probability reflects our intuition: the probabilities p_i (strings X_i) are chosen in an arbitrary way, and the following bit b_i should follow the declared distribution.

Theorem 10. *Let p be a computable function on binary strings with positive rational values that determines a measure, i.e., $p(b_1 \dots b_n) = p(b_1 \dots b_n 0) + p(b_1 \dots b_n 1)$. Then a sequence $b_1 b_2 \dots$ is Martin-Löf random with respect to this measure if and only if the bits b_i are on-line random in a sequence $p_1, b_1, p_2, b_2, \dots$ where p_i is a conditional probability of $b_i = 1$ after the prefix $b_1 \dots b_{i-1}$, i.e., the ratio $p(b_1 \dots b_{i-1} 1) / p(b_1 \dots b_{i-1})$.*

Proof. This is a direct consequence of the supermartingale criterion. Assume that the sequence $b_1 b_2 \dots$ is not random with respect to measure p . Then there exists a lower semicomputable supermartingale with respect to p that is unbounded on the prefixes of this sequence. This supermartingale can be extended to an on-line lower semicomputable supermartingale: we let it to be zero on finite sequence $p_1, b_1, \dots, p_i, b_i$ where one of p_j differs from conditional probability of 1 after $b_1 \dots b_{j-1}$ according to p . Therefore the sequence $p_1, b_1, p_2, b_2, \dots$ is not on-line random.

On the other hand, if the sequence $p_1, b_1, \dots$ is not on-line random, there exists a lower semicomputable on-line supermartingale that is unbounded on this

sequence. The restriction of this supermartingale on the subtree that contains only the vertices compatible with p , is a lower semicomputable on the binary tree with respect to measure p . This martingale is unbounded on $b_1b_2\dots$, therefore this sequence is not Martin-Löf random with respect to measure p . $\square$

Remarks

1. We restrict our attention to the simplest case where the measure has rational values and is computable as a rational-valued function. More general definition is analyzed in [8].

2. Note that probabilities p_i play a double role in this definition. First, they determine the coefficients in the definition of on-line supermartingale; this is their “primary” role. However, they are also source of information that can be used in the computation of this supermartingale. This was not important, since if p is a computable rational-valued measure, then the conditional probabilities can be computed from other available information.

In fact, our randomness intuition is quite contradictory here. Imagine that, for example, p_i are rational numbers that converge very fast to $1/2$, e.g., $p_i = 1/2 + c_i/2^{2^i}$, where c_i is equal either to 0 or to 1. Since the convergence is very fast, we would naturally expect that randomness would be the same as for the uniform Bernoulli measure (independently of c_i). On the other hand, if we watch the random number generator of the type described and observe that generated bit b_i is always equal to c_i , this compromises the fairness of the generator. Our definition follows the second direction: b_i that is equal to c_i is *not* random.

Acknowledgements

We thank the participants of the Kolmogorov seminar (Moscow), Workshop on effective randomness (Chicago University, 2007) and Workshop on game-theoretic probability and related topics (Tokyo University, 2008), where some of these results were presented.

This work was partly supported by EPSRC grant EP/F002998/1, Sycomore ANR grant and RFBF grants 05-01-02803-CNRS-a, 06-01-00122-a. Alexey Chernov is grateful for support to J. Schmidhuber and IDSIA (Lugano, Switzerland) where part of the work was done under J. Schmidhuber’s SNF grant 200021-113364.

References

- [1] Gács, P.: Lecture notes on descriptonal complexity and randomness, <http://www.cs.bu.edu/faculty/gacs/papers/ait-notes.pdf>
- [2] Levin, L.A.: Uniform tests of randomness. Soviet Math. Dokl. 17(2), 337–340 (1976)
- [3] Li, M., Vitányi, P.: An Introduction to Kolmogorov Complexity and Its Applications, 2nd edn. Springer, New York (1997)
- [4] Muchnik, An. A., Chernov, A., Shen, A.: Algorithmic randomness and splitting of supermartingales, arxiv.org:0807.3156

- [5] Shen, A.: Algorithmic information theory and Kolmogorov complexity, Technical Report 2000-034. Uppsala Universitet publication,
<http://www.it.uu.se/research/publications/reports/2000-034>
- [6] Uspensky, V.A., Semenov, A.L., Shen, A.: Can an individual sequence of zeros and ones be random? *Russian Mathematics Surveys* 45, 121–189 (1990)
- [7] Shafer, G., Vovk, V.: *Probability and Finance: It's Only a Game*. Wiley, New York (2001)
- [8] Vovk, V., Shen, A.: Prequential randomness. In: Freund, Y., Györfi, L., Turán, G., Zeugmann, T. (eds.) *ALT 2008. LNCS(LNAI)*, vol. 5254, pp. 154–168. Springer, Heidelberg (2008)

Prequential Randomness

Vladimir Vovk¹ and Alexander Shen²

¹ Department of Computer Science, Royal Holloway, University of London
Egham, Surrey TW20 0EX, England

`vovk@cs.rhul.ac.uk`

² Laboratoire d'Informatique Fondamentale, Centre de Mathématiques et
Informatique, 39 rue Joliot-Curie, F-13453 Marseille Cedex 13, France

`alexander.shen@lif.univ-mrs.fr`

Abstract. This paper studies Dawid's prequential framework from the point of view of the algorithmic theory of randomness. The main result is that two natural notions of randomness coincide. One notion is the prequential version of the measure-theoretic definition due to Martin-Löf, and the other is the prequential version of the game-theoretic definition due to Schnorr and Levin. This is another manifestation of the close relation between the two main paradigms of randomness.

1 Introduction

We consider the following on-line learning protocol:

PROBABILITY FORECASTING OF BINARY OBSERVATIONS

FOR $n = 1, 2, \dots$:

Learner announces $p_n \in [0, 1]$.

Reality announces $y_n \in \{0, 1\}$.

END FOR.

Intuitively, p_n is Learner's subjective probability that $y_n = 1$ after having observed $y_1, \dots, y_{n-1}$ and taking account of all other relevant information available at the time of issuing the forecast. We will refer to p_n as *forecasts* and to y_n as *outcomes*.

When can we say that Learner is doing a good job of forecasting? Or as we shall say, when is the sequence of outcomes $(y_1, y_2, \dots)$ “random” w.r. to the sequence of forecasts $(p_1, p_2, \dots)$? (We further abbreviate this by saying that the sequence $(p_1, y_1, p_2, y_2, \dots)$ containing both forecasts and outcomes is random.) This paper demonstrates the equivalence of two superficially quite different answers to this question.

Standard Theory

The simplest, and well-studied, situation is where the forecasts are produced as conditional probabilities from a computable probability distribution P on

$\{0,1\}^\infty$: p_n is the conditional probability according to P that $y_n = 1$ given $y_1, \dots, y_{n-1}$. In this case it is natural to talk about the randomness of $(y_1, y_2, \dots)$ w.r. to P rather than w.r. to $(p_1, p_2, \dots)$. The two fundamental paradigms of randomness are measure-theoretic and game-theoretic. The first paradigm is usually referred to as typicality, and various flavours and modifications of the second paradigm are referred to as unpredictability, stochasticity, chaoticity, and incompressibility (see below). This terminology, however, would be counterintuitive in the general framework of this paper. (To some degree this is also true about the standard term “random”, but the reader is perhaps already accustomed to the technical meaning of this term being different from its everyday meaning.)

The game-theoretic notion of randomness is based, as can be guessed, on the idea of gaming: a sequence of outcomes is regarded random if there is no computable way to become infinitely rich betting on its elements. Similarly, the measure-theoretic notion is based on the idea of measure: a sequence is regarded random if there is no computable way to specify a set of measure zero containing this sequence. The standard game-theoretic definition of randomness is due to Schnorr [18] and Levin ([12], Theorem 3), and the standard measure-theoretic definition of randomness is due to Martin-Löf [14]. Schnorr and Levin [18, 12] also established equivalence between the two definitions.

The measure-theoretic definition of randomness w.r. to a non-computable probability distribution P was first given by Levin [12] and later developed, modified, and applied in, e.g., [13, 8, 21, 9].

Remark 1. The terms “typicality”, “chaoticity”, and “stochasticity” were used in [11]; “stochasticity” was introduced in Kolmogorov’s earlier papers, and “typicality” and “chaoticity” were introduced in [11] itself. Modern literature often talks about “unpredictability” and “incompressibility” (see, e.g., [1], Chapter 1). As we said, typicality is synonymous with what we call measure-theoretic randomness, and the other four terms are various versions of game-theoretic randomness. Unpredictability is synonymous with our game-theoretic randomness. Stochasticity is the game-theoretic notion of randomness based on von Mises’s idea of subsequence selection rules, which Ville [20] showed to be inadequate in some important respects. The synonyms “chaoticity” and “incompressibility” require that the algorithmic complexity of initial fragments of the sequence should be close to their trivial upper bound. If the complexity is defined as the minus logarithm of the *a priori* semimeasure, this is the same as unpredictability. However, other notions of complexity (such as plain, prefix, and monotonic) have also been considered, and in this case chaoticity/incompressibility is sometimes regarded as a third, information-theoretic, paradigm, based on coding; but in any case, this third paradigm is very close to the game-theoretic one, as the connections between coding and gambling are straightforward and well understood (see, e.g., [10]). Levin’s [12] (Theorem 3) representation of game-theoretic randomness is in terms of complexity (the chaoticity/incompressibility approach) and Schnorr’s [18] is in terms of martingales (the unpredictability approach).

Prequential Framework for Randomness

The standard definitions of randomness mentioned in the previous subsection depend on knowing Learner's probability model P . Our forecasting protocol, however, only involves the realized forecasts p_n , which are not assumed to be calculated from any P . This feature of the protocol greatly extends its area of application, allowing forecasts produced "on the fly". Dawid's prequential principle ([3]; it is called "M2" in [4] and "weak prequential principle" in [5, 6]) says that our evaluation of the quality of the forecasts $p_1, p_2, \dots$ in light of the observed outcomes $y_1, y_2, \dots$ should not depend on Learner's model P even if it exists and is known.

The first definition of randomness fully respecting the prequential principle was proposed by Dawid [4]. Dawid's definition, however, was based on von Mises's idea of subsequence selection rules. Dawid ([4], Section 13.2) also gave a brief description of a prequential definition based on Ville's martingales, but did not elaborate on it. Chernov *et al.* [2] give the details of the martingale definition in the case where the forecasts are only allowed to take values from a finite set. This paper provides the details of the general martingale definition, which belongs to the game-theoretic paradigm. It also gives a Bayesian definition of measure-theoretic randomness. Its main mathematical result says that the notions of measure-theoretic randomness (called measure-randomness for brevity) and of game-theoretic randomness (called game-randomness) coincide in the prequential framework.

Some Notation and Definitions

The set of all natural (i.e., positive integer) numbers is denoted $\mathbb{N}$, $\mathbb{N} := \{1, 2, \dots\}$; $\overline{\mathbb{N}}_0$ is $\mathbb{N}$ extended by adding ∞ and 0. As always, $\mathbb{Q}$ and $\mathbb{R}$ are the sets of all rational and real numbers, respectively.

Let $\Omega := \{0, 1\}^\infty$ be the set of all infinite binary sequences and $\Omega^\circ := \{0, 1\}^*$ be the set of all finite binary sequences. Set $\Pi := ([0, 1] \times \{0, 1\})^\infty$ and $\Pi^\circ := ([0, 1] \times \{0, 1\})^*$. The empty element (sequence of length zero) of both Ω° and Π° will be denoted $\square$. In our applications, the elements of Ω and Ω° will be sequences of outcomes (infinite or finite), and the elements of Π and Π° will be sequences of forecasts and outcomes (infinite or finite). The set Π will sometimes be referred to as the *prequential space*.

For $x \in \Omega^\circ$, let $\Gamma_x \subseteq \Omega$ be the set of all infinite continuations of x . Similarly, for $x \in \Pi^\circ$, $\Gamma_x \subseteq \Pi$ is the set of all infinite continuations of x . For each $\omega = (y_1, y_2, \dots) \in \Omega$ and $n \in \mathbb{N}$, set $\omega^n := (y_1, \dots, y_n)$. Similarly, for each $\pi = (p_1, y_1, p_2, y_2, \dots) \in \Pi$ and $n \in \mathbb{N}$, set $\pi^n := (p_1, y_1, \dots, p_n, y_n)$.

For understanding the intuitive meaning of our statements, the following intuitive idea of lower semicontinuity will suffice: a function $f : X \rightarrow \mathbb{R} \cup \{\infty\}$ is lower semicomputable if there is an algorithm that, for all $x \in X$ and $r \in \mathbb{R}$, will eventually tell us that $f(x) > r$ if this inequality is indeed true. (Lower semicomputable functions are not necessarily computable as the algorithm can work arbitrarily long.) Understanding the proofs requires precise definitions, as given in, e.g., Appendix A.

2 Game-Randomness

A *farthingale* is a function $V : \Pi^\diamond \rightarrow [-\infty, \infty]$ satisfying

$$\begin{aligned} V(p_1, y_1, \dots, p_{n-1}, y_{n-1}) \\ = (1 - p_n)V(p_1, y_1, \dots, p_{n-1}, y_{n-1}, p_n, 0) \\ + p_n V(p_1, y_1, \dots, p_{n-1}, y_{n-1}, p_n, 1) \end{aligned} \quad (1)$$

for all n and all $(p_1, y_1, p_2, y_2, \dots) \in \Pi$; the product 0∞ is defined to be 0. If we replace “=” by “ $\geq$ ” in (1), we get the definition of *superfarthingales*. These are prequential versions of the standard notions of martingale and supermartingale, and in our terminology we follow [6]. We will be interested mainly in non-negative farthingales and superfarthingales.

The value of a farthingale can be interpreted as the capital of a gambler betting according to the odds announced by Learner. In the case of superfarthingales, the gambler is allowed to throw away part of his capital.

Lemma 1. *Let $\mathcal{V}$ be the class of all non-negative lower semicomputable superfarthingales V with initial value $V(\square) = 1$. There exists a largest superfarthingale in $\mathcal{V}$ to within a constant factor. In other words, there exists a superfarthingale $V \in \mathcal{V}$ such that, for any other superfarthingale V' , there exists a constant $C > 0$ such that, for any $x \in \Pi^\diamond$, $V(x) \geq V'(x)/C$.*

Proof. Fix a universal computable sequence of lower semicomputable functions $f_1, f_2, \dots$ on $\Pi^\diamond$ (see Lemma 6 in Appendix A). It is easy to construct a new computable sequence of lower semicomputable functions $f'_1, f'_2, \dots$ such that each of f'_l is a superfarthingale in $\mathcal{V}$ and that $f'_l = f_l$ whenever f_l is already in $\mathcal{V}$, $l \in \mathbb{N}$. Then $\sum_{l=1}^\infty 2^{-l} f'_l$ will be a largest, to within a constant factor, superfarthingale in $\mathcal{V}$. $\square$

Let us fix a largest, to within a constant factor, superfarthingale U in $\mathcal{V}$ and call it the *universal superfarthingale*. A sequence $\pi \in \Pi$ is *game-random* if $U(\pi^n)$ stays bounded as $n \rightarrow \infty$. The following lemma gives an equivalent definition of game-random sequences.

Lemma 2. *A sequence $\pi \in \Pi$ is game-random if and only if $U(\pi^n)$ does not tend to infinity as $n \rightarrow \infty$.*

Proof. Following the proof of Lemma 3.1 in [19], we can construct a superfarthingale $V \in \mathcal{V}$ such that $\liminf_{n \rightarrow \infty} V(\pi^n) = \infty$ whenever $\sup_n U(\pi^n) = \infty$. (Therefore, $\liminf_{n \rightarrow \infty} U(\pi^n) = \infty$ whenever $\sup_n U(\pi^n) = \infty$.) Indeed, for each $m \in \mathbb{N}$, the function $U^m : \Pi^\diamond \rightarrow [0, \infty)$ defined by

$$U^m(x) := \begin{cases} 2^m & \text{if } U(y) > 2^m \text{ for some prefix } y \text{ of } x \\ U(x) & \text{otherwise} \end{cases}$$

is a superfarthingale; it is clear that it is lower semicomputable and so belongs to $\mathcal{V}$. Since $U^1, U^2, \dots$ is a computable sequence of lower semicontinuous functions, we can set

$$V := \sum_{m=1}^{\infty} 2^{-m} U^m. \quad \square$$

3 Measure-Randomness

We can also adapt the standard measure-theoretic definition of randomness to the prequential framework. First we give an informal version of the definition.

A *forecasting system* is a function $\phi : \Omega^\circ \rightarrow [0, 1]$. Let Φ be the set of all forecasting systems. For each $\phi \in \Phi$ there exists a unique probability measure $\mathbb{P}_\phi$ on Ω such that, for each $x \in \Omega^\circ$, $\mathbb{P}_\phi(\Gamma_{x1}) = \phi(x) \mathbb{P}_\phi(\Gamma_x)$. (In other words, such that $\phi(x)$ is a version of the conditional probability, according to $\mathbb{P}_\phi$, that x will be followed by 1.) The notion of a forecasting system is close to that of a probability measure on Ω : the correspondence $\phi \mapsto \mathbb{P}_\phi$ becomes an isomorphism if we only consider forecasting systems taking values in the open interval $(0, 1)$ and probability measures taking positive values on the sets Γ_x , $x \in \Omega^\circ$.

Informally, we say that a sequence $\omega \in \Omega$ is *measure-random* w.r. to a forecasting system ϕ if it is random in the sense of Martin-Löf [14] w.r. to $\mathbb{P}_\phi$ when ϕ is given as an oracle. We will formalize “given as an oracle” using some simplest notions of effective topology (see Appendix A). The following definition is a version of Levin’s “uniform test of randomness” [12, 13, 9].

Definition 1. A uniform test of randomness is a lower semicomputable function $T : \Omega \times \Phi \rightarrow \overline{\mathbb{N}}_0$ such that, for all $\phi \in \Phi$ and all $m \in \mathbb{N}$,

$$\mathbb{P}_\phi\{\omega \in \Omega \mid T(\omega, \phi) \geq m\} \leq 2^{-m}. \quad (2)$$

Intuitively, $T(\omega, \phi)$ is the amount of irregularities (measured in bits, according to (2)) discovered in ω w.r. to ϕ . The requirement of lower semicomputability means that the irregularities have to be genuine: a discovery of irregularity can never be undone. We will usually drop the adjective “uniform”.

Lemma 3. There exists a largest, to within an additive constant, test of randomness. In other words, there exists a test of randomness T such that, for any other test of randomness T' , there exists a constant C such that, for any $(\omega, \phi) \in \Omega \times \Phi$,

$$T(\omega, \phi) \geq T'(\omega, \phi) - C.$$

Proof. The proof is similar to the standard one given by Martin-Löf [14]; it will, however, crucially depend on the compactness of Φ , as in [12, 9]. For each set $G \subseteq \Omega \times \Phi$ and each $\phi \in \Phi$ we will use the notation

$$G[\phi] := \{\omega \in \Omega \mid (\omega, \phi) \in G\}$$

for the ϕ -cut of G . A convenient alternative representation of a test of randomness T is as a computable sequence of nested open sets $G_1 \supseteq G_2 \supseteq \dots$ in $\Omega \times \Phi$ such that

$$\mathbb{P}_\phi(G_m[\phi]) \leq 2^{-m} \quad (3)$$

for all $\phi \in \Phi$ and $m \in \mathbb{N}$. It is easy to see that the representations are indeed equivalent: when given T we can set $G_m := \{(\omega, \phi) \mid T(\omega, \phi) \geq m\}$, and when given $G_1, G_2, \dots$ we can set $T(\omega, \phi) := \max\{m \mid (\omega, \phi) \in G_m\}$. Such sequences $G_1, G_2, \dots$ will also be referred to as *tests of randomness*.

Let $G_{l,m}$ be a universal computable family of sequences of open sets (cf. Lemma 5 in Appendix A). Put $G'_{l,m} := \cap_{i=1}^m G_{l,i}$, so that $G'_{l,m}$ is a computable family of nested sequences of open sets containing all nested computable sequences of open sets. We can further “trim” each $G'_{l,m}$ to $G''_{l,m}$ so that:

- $\mathbb{P}_\phi(G''_{l,m}[\phi]) \leq 2^{-m}$ for all $\phi \in \Phi$;
- $G''_{l,m} = G'_{l,m}$ whenever $\mathbb{P}_\phi(G'_{l,m}[\phi]) < 2^{-m}$ for all $\phi \in \Phi$.

Indeed, let $G'_{l,m} = \cup\{U_k \mid (l, m, k) \in A\}$ be the representation of $G'_{l,m}$ as the union of basic sets. Set $H_K := \cup\{U_k \mid (l, m, k) \in A, k \leq K\}$, so that $H_1, H_2, \dots$ is a non-decreasing sequence of simple sets whose union is $G'_{l,m}$. Remember that, by (11), $\overline{H_K} \subseteq G'_{l,m}$. We may “quarantine” new H_K until they are “cleared”, i.e.,

$$\forall \phi \in \Phi : \mathbb{P}_\phi(\overline{H_K}[\phi]) < 2^{-m} \quad (4)$$

is established. The open set $G''_{l,m}$ is defined as the union of the H_K that are cleared.

Let us check that condition (4) can indeed be eventually established by a computable procedure when it is satisfied. Suppose (4) is satisfied. The set

$$S := \{\phi \in \Phi \mid \mathbb{P}_\phi(\overline{H_K}[\phi]) < 2^{-m}\}$$

is effectively open, so that we can effectively generate a sequence of basic sets $U'_k \subseteq \Phi$ whose union is S . By the compactness of Φ , already a finite number of U'_k will cover S when $S = \Phi$, and so (4) can be established in a computable manner.

Therefore, we can list all tests of randomness, in the following sense: there is a computable sequence $(G''_{l,m})_{m=1}^\infty$, $l = 1, 2, \dots$, of tests of randomness that contains all “strict” tests of randomness (i.e., those satisfying the required inequality with “ $<$ ” instead of “ $\leq$ ”; any test of randomness G_m can be made strict by redefining $G_m := G_{m+1}$, $m = 1, 2, \dots$). To obtain a largest test of randomness G_m , it suffices to set

$$G_m := \cup_{l=1}^\infty G''_{l,m+l}.$$

Indeed, the computability of the sequence of open sets G_m is obvious,

$$\mathbb{P}_\phi(G_m[\phi]) \leq \sum_{l=1}^\infty \mathbb{P}_\phi(G''_{l,m+l}[\phi]) \leq \sum_{l=1}^\infty 2^{-m-l} = 2^{-m}, \quad \forall \phi \in \Phi, \forall m \in \mathbb{N},$$

and, for each $l \in \mathbb{N}$,

$$T(\omega, \phi) = \max\{m \mid (\omega, \phi) \in G_m\} \geq \max\{m \mid (\omega, \phi) \in G''_{l,m+l}\} = T_l(\omega, \phi) - l,$$

where T is the functional representation of the test $(G_m)_{m=1}^\infty$ and T_l is the functional representation of the test $(G''_{l,m})_{m=1}^\infty$. $\square$

Let us fix a largest, to within an additive constant, test of randomness T and call it the *universal test of randomness*. A sequence $\omega \in \Omega$ is said to be *measure-random w.r. to $\phi \in \Phi$* if $T(\omega, \phi) < \infty$.

Definition 2. We say that $\pi = (p_1, y_1, p_2, y_2, \dots) \in \Pi$ is *measure-random* if there exists a forecasting system ϕ such that $(y_1, y_2, \dots)$ is *measure-random w.r. to ϕ* and ϕ agrees with π , in the sense that $p_n = \phi(y_1, \dots, y_{n-1})$ for all $n \in \mathbb{N}$.

4 Main Result

Theorem 1. A sequence $\pi \in \Pi$ is *game-random* if and only if it is *measure-random*.

This theorem will be proved in the next section. The proof will be based on Levin's [12] ideas (see also [9]). A related result is Theorem 7 in [2], which is technically much simpler but uses a less natural definition.

The philosophical significance of Theorem 1 is that it establishes the equivalence of the purely prequential and Bayesian viewpoints in the framework of the algorithmic theory of randomness. The definition of measure-randomness is Bayesian, in that Learner is modelled as a coherent decision maker, computing his forecasts by conditioning a probability measure; rejecting the forecasts is the same as rejecting all probability measures that could have produced those forecasts. The definition of game-randomness is purely prequential, in that it does not see any probability measures behind the forecasts; the latter are used for testing directly.

A simple corollary of Theorem 1 is the following observation:

Corollary 1. Let ϕ be a computable forecasting system such that $\phi(x) > 0$ for all $x \in \Omega^\circ$. A binary sequence $(y_1, y_2, \dots)$ is *random w.r. to $\mathbb{P}_\phi$* in the sense of Martin-Löf if and only if the sequence $(p_1, y_1, p_2, y_2, \dots)$ is *game-random* (equivalently, *measure-random*), where $p_n := \phi(y_1, \dots, y_{n-1})$, $n \in \mathbb{N}$.

Therefore, the prequential notions of game-randomness and measure-randomness generalize Martin-Löf's notion of randomness. Corollary 1 generalizes Theorem 10 in [2].

Remark 2. Notice that we have never assumed that the past observations $y_1, \dots, y_{n-1}$ are the only information available to Learner when choosing the forecast p_n for the next outcome y_n . Learner is allowed to (and typically does) use all kinds of "side information" in addition to the past observations. It is easy to extend all our definitions and results to the case where some of this side

information, x_n , is also known to the gambler. (As in [4], Section 9, and [2].) As an example, the definition of a superfarthingale, (1), becomes

$$\begin{aligned} V(x_1, p_1, y_1, \dots, x_{n-1}, p_{n-1}, y_{n-1}) \\ = (1 - p_n)V(x_1, p_1, y_1, \dots, x_{n-1}, p_{n-1}, y_{n-1}, x_n, p_n, 0) \\ + p_n V(x_1, p_1, y_1, \dots, x_{n-1}, p_{n-1}, y_{n-1}, x_n, p_n, 1). \end{aligned}$$

Remark 3. Since we do not record side information in the main part of this paper, the forecasting systems that we consider are never assumed computable: even if Learner computes each forecast from the past outcomes and the side information, typically the forecast cannot be computed from the past outcomes alone. It is not even obvious that the notion of a forecasting system ϕ as we defined it (a function of past outcomes) is meaningful. It involves the following controversial picture along the lines of Pearl’s ([17], Section 6.2) “local surgeries”. To elicit the value of the function ϕ on a binary sequence $y_1, \dots, y_n$, we act as follows. First we wait until Reality produces the first piece of side information x_1 and, in response, Learner produces p_1 . Then we perform a “local surgery” replacing Reality’s outcome by y_1 (if it is different from y_1). Now Reality produces x_2 and Learner produces p_2 . Another local surgery replaces the outcome by y_2 . Etc. Finally, Learner produces p_n , which is taken to be the value of ϕ on $y_1, \dots, y_n$. According to Theorem 1, this philosophically questionable approach (see, e.g., Section 4 of Pearl’s response in [16]) leads to the same notion of randomness as the philosophically immaculate approach of Section 2.

Remark 4. It is easy to see that Theorem 1 fails if in the definition of measure-typicalness we require that ϕ should range over computable forecasting systems. Indeed, take any non-computable sequence $(y_1, y_2, \dots) \in \Omega$ and consider $\pi := (y_1, y_1, y_2, y_2, \dots)$ as an element of Π . It is clear that π is game-random (no farthingale can grow on it) but no computable forecasting system agrees with π .

5 Proof of Theorem 1 and Corollary 1

The proof of the theorem will depend on a fundamental result called Ville’s inequality. Let ϕ be a forecasting system. A *martingale* w.r. to ϕ is a function $V : \Omega^\diamond \rightarrow [-\infty, \infty]$ satisfying

$$V(x) = (1 - \phi(x))V(x, 0) + \phi(x)V(x, 1) \tag{5}$$

for all $x \in \Omega^\diamond$ (with the same convention $0\infty := 0$). If we replace “=” by “ $\geq$ ” (respectively, by “ $\leq$ ”) in (5), we get the definition of a *supermartingale* (respectively, *submartingale*) w.r. to ϕ .

Proposition 1 (Ville’s inequality, [20], p. 100). *If ϕ is a forecasting system, V is a non-negative supermartingale w.r. to ϕ with initial value $V(\square) = 1$, and $C > 0$,*

$$\mathbb{P}_\phi \left\{ \omega \in \Omega \mid \sup_n V(\omega^n) \geq C \right\} \leq \frac{1}{C}.$$

Fix $\pi \in \Pi$.

Part “if” of Theorem 1

Suppose π is not game-random. Then $\pi \in G_m$ for all $m \in \mathbb{N}$, where

$$G_m := \left\{ \pi \in \Pi \mid \sup_n U(\pi^n) > 2^m \right\}$$

and U is the universal superfarthingale.

We will not distinguish between $(p_1, y_1, p_2, y_2, \dots) \in \Pi$ and the pair of sequences $((p_1, p_2, \dots), (y_1, y_2, \dots)) \in [0, 1]^\infty \times \Omega$. For $\phi \in \Phi$ and $\omega = (y_1, y_2, \dots) \in \Omega$ we set

$$\phi(\omega) := (\phi(\square), \phi(y_1), \phi(y_1, y_2), \dots) \in [0, 1]^\infty.$$

The mapping $(\omega, \phi) \mapsto \phi(\omega)$ from $\Omega \times \Phi$ to $[0, 1]^\infty$ is continuous. Therefore, the mapping $(\omega, \phi) \mapsto (\phi(\omega), \omega)$ from $\Omega \times \Phi$ to Π is also continuous. Therefore, the set

$$G'_m := \{(\omega, \phi) \mid (\phi(\omega), \omega) \in G_m\}$$

is open.

Let us check that G'_m is a test of randomness. The computability requirement follows from Lemma 7 (which is uniform in C) in Appendix A. Fix $m \in \mathbb{N}$ and $\phi \in \Phi$. To check (3), i.e., $\mathbb{P}_\phi(G'_m[\phi]) \leq 2^{-m}$ in the current notation, notice that the function $U^\phi : \Omega^\circ \rightarrow [0, \infty]$ defined by

$$U^\phi(y_1, \dots, y_n) := U(\phi(\square), y_1, \phi(y_1), y_2, \dots, \phi(y_1, \dots, y_{n-1}), y_n)$$

is a non-negative supermartingale w.r. to ϕ . Now Ville’s inequality implies

$$\begin{aligned} \mathbb{P}_\phi(G'_m[\phi]) &= \mathbb{P}_\phi\{\omega \in \Omega \mid (\omega, \phi) \in G'_m\} = \mathbb{P}_\phi\{\omega \in \Omega \mid (\phi(\omega), \omega) \in G_m\} \\ &= \mathbb{P}_\phi\left\{\omega \in \Omega \mid \sup_n U^\phi(\omega^n) > 2^m\right\} \leq 2^{-m}, \quad \forall \phi \in \Phi. \end{aligned}$$

Suppose π , assumed to be not game-random, is measure-random. Then there exists $\phi \in \Phi$ such that $\pi = (\phi(\omega), \omega)$ for some ω measure-random w.r. to ϕ . Since $\pi \in G_m$, we have $(\omega, \phi) \in G'_m$; since this is true for each $m \in \mathbb{N}$, ω is not measure-random w.r. to ϕ , and so we have arrived at a contradiction.

Part “only if” of Theorem 1

Let $G_m = \cup\{U_k \mid (m, k) \in A\}$ be a representation of the universal test of randomness via basic sets, with $A \subseteq \mathbb{N}^2$ a recursively enumerable set. Without loss of generality we can assume that each basic set U_k in this representation has the form $\Gamma_c \times \{\phi \in \Phi \mid a(x) < \phi(x) < b(x), \forall x \in \Omega^{\leq n}\}$ for some $c \in \Omega^n$, $a, b : \Omega^{\leq n} \rightarrow \mathbb{Q}$, and $n \in \mathbb{N}$. Define G'_m to be the set of all $(p_1, y_1, p_2, y_2, \dots) \in \Pi$ such that $((y_1, y_2, \dots), \phi) \in G_m$ for all ϕ that agree with $(p_1, y_1, p_2, y_2, \dots)$.

The compactness of Φ easily implies that each set $G'_m \subseteq \Pi$ is open. Indeed, suppose $\pi = (p_1, y_1, p_2, y_2, \dots) \in G'_m$. For each $\phi \in \Phi$, either ϕ disagrees with π or $((y_1, y_2, \dots), \phi) \in G_m$. In both cases there is a neighbourhood O'_ϕ of π and a neighbourhood O''_ϕ of ϕ such that either all elements of O'_ϕ disagree with

all elements of O'_ϕ or $((y'_1, y'_2, \dots), \phi') \in G_m$ for all $(p'_1, y'_1, p'_2, y'_2, \dots) \in O'_\phi$ and all $\phi' \in O''_\phi$. Since Φ is compact, there is a finite set $\phi_1, \dots, \phi_J$ such that $\cup_{j=1}^J O''_{\phi_j} = \Phi$. We can see that the neighbourhood $\cap_{j=1}^J O'_{\phi_j}$ of π is a subset of G'_m .

The same argument shows that the G'_m form a computable sequence of open sets. Let us show that there exists a non-negative superfarthingale V_m with initial value 2^{-m} or less that eventually exceeds 1 on each sequence in G'_m . (In this sense G'_m form a prequential test of randomness.)

Let $G'_m = \cup\{U_k \mid (m, k) \in A\}$ be a representation of G'_m via basic sets, where $A \subseteq \mathbb{N}^2$ is a recursively enumerable set. Let $A = \cup_{i=1}^\infty A_i$ be a representation of A as the union of a computable nested sequence $\emptyset \subset A_1 \subseteq A_2 \subseteq \dots$ of finite sets. Fix an m . For each $i \in \mathbb{N}$, define a superfarthingale W_i as follows. Let N be so large that, for all $x \in \Pi^N$ and $(m, k) \in A_i$, either $\Gamma_x \subseteq U_k$ or $\Gamma_x \cap U_k = \emptyset$. (For example, we can set N to the largest n_k in (10) over k such that $(m, k) \in A_i$.) For $n \geq N$ and $x \in \Pi^n$, set

$$W_i(x) := \begin{cases} 1 & \text{if } \Gamma_x \subseteq U_k \text{ for some } k \text{ with } (m, k) \in A_i \\ 0 & \text{otherwise.} \end{cases}$$

After that proceed by backward induction. If $W_i(x)$ is already defined for $x \in \Pi^n$, $n = N, N-1, \dots, 1$, set, for each $x \in \Pi^{n-1}$,

$$W_i(x) := \sup_{p \in [0,1]} ((1-p)W_i(x, p, 0) + pW_i(x, p, 1)). \quad (6)$$

It is clear that W_i is a superfarthingale that does not depend on the choice of N .

We will need to establish several properties of W_i . First, it is lower semicontinuous. Indeed, there is an N (e.g., the largest n_k in (10) over $(m, k) \in A_i$) such that $W_i(x)$ is lower semicontinuous when x is restricted to Π^n with $n \geq N$. (It will be even lower semicomputable when x is restricted to $\Pi^{\geq N}$.) And the operation $\sup$ preserves lower semicontinuity:

Lemma 4. *If a function $f : X \times Y \rightarrow \mathbb{R}$ defined on the product of topological spaces X and Y is lower semicontinuous, then the function $x \in X \mapsto g(x) := \sup_{y \in Y} f(x, y)$ is also lower semicontinuous.*

Proof. It suffices to notice that, for each $c \in \mathbb{R}$, $\{x \mid g(x) > c\} = \{x \mid \exists y : f(x, y) > c\}$, and projections of open sets are open. $\square$

The lower semicontinuity of W_i implies its lower semicomputability: indeed, we can restrict p to $\mathbb{Q} \cap [0, 1]$ in (6).

Let us check that $W_i(\square) \leq 2^{-m}$. Suppose that, on the contrary, $W_i(\square) > 2^{-m}$. Construct a forecasting system ϕ as follows. (The words such as “construct” and “choose” are not intended to imply computability: there are no computability restrictions in this paragraph.) For each $x \in \Omega^n$, $n = 0, 1, \dots, N-1$, choose $\phi(x)$ such that

$$\begin{aligned}
& (1 - \phi(x))W_i(x, \phi(x), 0) + \phi(x)W_i(x, \phi(x), 1) \\
& \geq \sup_{p \in [0,1]} ((1-p)W_i(x, p, 0) + pW_i(x, p, 1)) - \epsilon/N = W_i(x) - \epsilon/N,
\end{aligned}$$

where $\epsilon > 0$ satisfies $W_i(\square) > 2^{-m} + \epsilon$. For each $x \in \Omega^{\geq N}$, define $\phi(x)$ arbitrarily, say $\phi(x) := 0$. Since $(\phi(\omega), \omega) \notin G'_m$ for all $\omega \notin G_m[\phi]$, we have $W_i^\phi(\omega^N) = 0$ for all $\omega \notin G_m[\phi]$. Combining the fact that

$$\{\omega \mid W_i^\phi(\omega^N) = 1\} \subseteq G_m[\phi]$$

with the fact that the function $x \in \Omega^\circ \mapsto S(x) := W_i^\phi(x) + \epsilon n/N$, where n is the length of x , is a submartingale w.r. to ϕ , we obtain

$$\begin{aligned}
\mathbb{P}_\phi(G_m[\phi]) & \geq \mathbb{P}_\phi\{\omega \mid W_i^\phi(\omega^N) = 1\} = \mathbb{E}_\phi W_i^\phi(\omega^N) = \mathbb{E}_\phi(S(\omega^N) - \epsilon) \\
& \geq S(\square) - \epsilon = W_i^\phi(\square) - \epsilon = W_i(\square) - \epsilon > 2^{-m}, \quad (7)
\end{aligned}$$

where $\mathbb{E}_\phi$ stands for the expectation of a function of $\omega \in \Omega$ w.r. to $\mathbb{P}_\phi$. The inequality between the extreme terms of (7) fails by the definition of a test of randomness.

Define $V_m(x) := \sup_i W_i(x)$, $x \in \Pi^\circ$, to be the limit of the non-decreasing sequence of superfarthingales W_i . It is clear that V_m is also a superfarthingale and $V_m(\square) \leq 2^{-m}$. Set $V := \sum_{m=1}^\infty V_m$; this is a lower semicomputable superfarthingale with initial value $V(\square) \leq 1$ (so that $V \in \mathcal{V}$ if we redefine $V(\square) := 1$).

Now it is easy to finish the proof of the theorem. Suppose that π is not measure-random. Then $\pi \in G'_m$ for all $m \in \mathbb{N}$. Then $V(\pi^n) \rightarrow \infty$ as $n \rightarrow \infty$, and so π is not game-random.

Corollary 1

Fix a sequence $\omega = (y_1, y_2, \dots) \in \Omega$, and set $\pi := (\phi(\omega), \omega) \in \Pi$. We will prove the equivalence of the game-randomness of π and Schnorr and Levin's reformulation of Martin-Löf randomness of ω w.r. to $\mathbb{P}_\phi$. Remember that Schnorr and Levin's reformulation is that the universal lower semicomputable supermartingale w.r. to ϕ is bounded on ω^n , $n \rightarrow \infty$; we will refer to this property as the *Schnorr–Levin randomness* of ω w.r. to ϕ .

Suppose π is not game-random. Then $U(\pi^n) \rightarrow \infty$ as $n \rightarrow \infty$, where U is the universal superfarthingale. Then $U^\phi(\omega^n) \rightarrow \infty$, and since U^ϕ is a lower semicomputable supermartingale w.r. to ϕ , ω is not Schnorr–Levin random.

Now suppose ω is not Schnorr–Levin random w.r. to ϕ . Let S be the universal lower semicomputable supermartingale w.r. to ϕ . We can transform S into a superfarthingale V by multiplying it by the likelihood ratio:

$$V(p_1, y_1, \dots, p_n, y_n) := S(y_1, \dots, y_n) \frac{\mathbb{P}_\phi(\Gamma_{(y_1, \dots, y_n)})}{\prod_{i=1}^n |p_i + y_i - 1|}$$

(the expression $|p_i + y_i - 1|$ is just a convenient code for the probability assigned by Learner outputting forecast p_i to the outcome y_i : it is p_i if $y_i = 1$, and it is $1 - p_i$ if $y_i = 0$). Since $V(\pi^n) = S(\omega^n) \rightarrow \infty$ as $n \rightarrow \infty$, π is not game-random.

Our argument depends on the superfarthingale V being lower semicomputable. The lower semicomputability of V is easy to check once we finish its definition for the boundary cases. We set $V(p_1, y_1, \dots, p_n, y_n) := 0$ when $S(y_1, \dots, y_n) = 0$ (although this case is in fact impossible since S is the universal supermartingale), and we set $V(p_1, y_1, \dots, p_n, y_n) := \infty$ when $S(y_1, \dots, y_n) > 0$ but $\prod_{i=1}^n |p_i + y_i - 1| = 0$. By our assumption, $\mathbb{P}_\phi(\Gamma_{(y_1, \dots, y_n)}) > 0$. It is an open problem to get rid of this assumption or show that it is essential. (If it turns out to be essential, the next open problem is whether it can be relaxed to the assumption that the set $\phi^{-1}(\{0, 1\}) \subseteq \Omega^\circ$ is computable.)

Acknowledgements

We are grateful to A. Philip Dawid and Glenn Shafer for illuminating discussions. This work was partially supported by EPSRC (grant EP/F002998/1).

References

- [1] Bienvenu, L.: Caractérisations de l'aléatoire par les jeux: impredictibilité et stochasticté. PhD thesis, Université de Provence (2008)
- [2] Chernov, A., Shen, A., Vereshchagin, N., Vovk, V.: On-line probability, complexity and randomness. These Proceedings
- [3] Dawid, A.P.: Statistical theory: the prequential approach. *Journal of the Royal Statistical Society A* 147, 278–292 (1984)
- [4] Dawid, A.P.: Calibration-based empirical probability (with discussion). *Annals of Statistics* 13, 1251–1285 (1985)
- [5] Dawid, A.P.: Prequential analysis. In: Kotz, S., Read, C.B., Banks, D.L. (eds.) *Encyclopedia of Statistical Sciences*, vol. 1, pp. 464–470. Wiley, New York (1997)
- [6] Dawid, A.P., Vovk, V.: Prequential probability: principles and properties. *Bernoulli* 5, 125–162 (1999)
- [7] Engelking, R.: *General Topology*, 2nd edn. Heldermann, Berlin (1989)
- [8] Gács, P.: Exact expressions for some randomness tests. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik* 26, 385–394 (1980)
- [9] Gács, P.: Uniform test of algorithmic randomness over a general space. *Theoretical Computer Science* 341, 91–137 (2005)
- [10] Kelly, J.L.: A new interpretation of information rate. *Bell System Technical Journal* 35, 917–926 (1956)
- [11] Kolmogorov, A.N., Uspensky, V.A.: Algorithms and randomness. *Theory of Probability and Its Applications* 32, 389–412 (1987); *Теория вероятностей и ее применения* 32(3), 425–455 (1987) (Russian original)
- [12] Levin, L.A.: On the notion of a random sequence. *Soviet Mathematics Doklady* 14, 1413–1416 (1973); *Доклады АН СССР* 212(1), 548–550 (1973) (Russian original)
- [13] Levin, L.A.: Uniform tests of randomness. *Soviet Mathematics Doklady* 17, 337–340 (1976); *Доклады АН СССР* 227(1), 33–35, (1976) (Russian original) Erratum: 227(1), 33–35 (1976)
- [14] Martin-Löf, P.: The definition of random sequences. *Information and Control* 9, 602–619 (1966)
- [15] Martin-Löf, P.: *Notes on Constructive Mathematics*. Almqvist & Wiksell, Stockholm (1970)

- [16] Pearl, J.: Causal diagrams for empirical research (with discussion). *Biometrika* 82, 669–710 (1995)
- [17] Pearl, J.: Causation, action and counterfactuals. In: Gammerman, A. (ed.) *Computational Learning and Probabilistic Reasoning*, ch. 6, pp. 103–124. Wiley, Chichester (1996)
- [18] Schnorr, C.P.: *Zufälligkeit und Wahrscheinlichkeit*. Springer, Berlin (1971)
- [19] Shafer, G., Vovk, V.: *Probability and Finance: It's Only a Game*. Wiley, New York (2001)
- [20] Ville, J.: *Etude critique de la notion de collectif*. Gauthier-Villars, Paris (1939)
- [21] Vovk, V., V'yugin, V.V.: On the empirical validity of the Bayesian method. *Journal of Royal Statistical Society B* 55, 253–266 (1993)

A Effective Topology

In this section we will give definitions of various notions connected with computability in topological spaces, mainly following Martin-Löf [15] (see also [9], Appendix C.2). The details of the definitions become important only in the proofs. We will use the terminology of Engelking [7].

In this appendix and in some proofs in the main part of the paper we will be using the following notation for $n \in \mathbb{N}$: $\Omega^n := \{0, 1\}^n$ is the set of all finite binary sequences of length n ; $\Omega^{\leq n}$ is the set of all finite binary sequences of length at most n ; $\Pi^n := ([0, 1] \times \{0, 1\})^n$; $\Pi^{\geq n} := \bigcup_{i=n}^{\infty} ([0, 1] \times \{0, 1\})^i$.

An *effective topological space* is a second-countable topological space with a fixed numbering $(U_k)_{k=1}^{\infty}$ of its countable base. In other words, an effective topological space is a triple $(X, \mathcal{O}, (U_k)_{k=1}^{\infty})$, where $(X, \mathcal{O})$ is a topological space and $(U_k)_{k=1}^{\infty}$ is a numbering of its countable base. The family $(U_k)_{k=1}^{\infty}$ is called the *effective base* of the effective topological space, and its elements are called *basic sets*. Finite unions of basic sets are called *simple sets*. We do not distinguish between two effective topological spaces $(X, \mathcal{O}, (U_k)_{k=1}^{\infty})$ and $(X', \mathcal{O}', (U'_k)_{k=1}^{\infty})$ if $(X, \mathcal{O}) = (X', \mathcal{O}')$ and there exists a computable bijection $f : \mathbb{N} \rightarrow \mathbb{N}$ such that $U'_k = U_{f(k)}$ for all k .

Example 1 ($\mathbb{N}$). The usual discrete topology on $\mathbb{N}$ has as its base the set of all singletons $\{k\}$, $k \in \mathbb{N}$. They can serve as the effective base, $U_k := \{k\}$.

Example 2 ($\mathbb{R}$). The topology on $\mathbb{R}$ has as its base the set of all intervals (a, b) , $a < b$. To make $\mathbb{R}$ into an effective topological space, fix a computable enumeration (a_k, b_k) , $k = 1, 2, \dots$, of all intervals with rational end-points, and take $U_k := (a_k, b_k)$ as the effective base.

Example 3 (Ω). The topology on $\Omega := \{0, 1\}^{\infty}$ is the usual product topology, which makes Ω a compact topological space. To make it into an effective topological space, fix a computable bijection $f : \mathbb{N} \rightarrow \Omega^{\circ}$ and take $U_k := \Gamma_{f(k)}$ as the effective base.

Example 4 (Φ). The basic sets in Φ (the set of all forecasting systems) have the form

$$\{\phi \in \Phi \mid a(x) < \phi(x) < b(x), \forall x \in \Omega^{\leq n}\} \quad (8)$$

for some $n \in \mathbb{N}$ and $a, b : \Omega^{\leq n} \rightarrow \mathbb{Q}$. Let (n_k, a_k, b_k) , $k = 1, 2, \dots$, be a computable enumeration of all such triples (n, a, b) . Set U_k to (8) with $(n, a, b) := (n_k, a_k, b_k)$.

Example 5 (II). The topology on the prequential space Π is the standard product topology of $[0, 1] \times \{0, 1\} \times [0, 1] \times \{0, 1\} \times \dots$. The basic sets are

$$\{(p_1, y_1, p_2, y_2, \dots) \in \Pi \mid a_1 < p_1 < b_1, y_1 = c_1, \dots, a_n < p_n < b_n, y_n = c_n\} \quad (9)$$

where n ranges over $\mathbb{N}$, $a_i, b_i \in \mathbb{Q}$, and $c_i \in \{0, 1\}$, $i = 1, \dots, n$. Let

$$(n_k, a_{1,k}, b_{1,k}, c_{1,k}, \dots, a_{n_k,k}, b_{n_k,k}, c_{n_k,k}) \quad (10)$$

be a computable enumeration of all such sequences $(n, a_1, b_1, c_1, \dots, a_n, b_n, c_n)$. We can define U_k as (9) with (10) in place of $(n, a_1, b_1, c_1, \dots, a_n, b_n, c_n)$.

Let X' and X'' be two effective topological spaces with effective bases $(U'_k)_{k=1}^\infty$ and $(U''_k)_{k=1}^\infty$, respectively. The Cartesian product of X' and X'' is the product of the topological spaces X' and X'' equipped with the effective base $(U_k)_{k=1}^\infty$, where $U_{f(k', k'')} := U'_{k'} \times U''_{k''}$ and $f : \mathbb{N}^2 \rightarrow \mathbb{N}$ is a fixed computable bijection. We will be particularly interested in the product $\Omega \times \Phi$; sometimes we will need products of more than two spaces, such as $\Omega \times \Phi \times \mathbb{R} := (\Omega \times \Phi) \times \mathbb{R}$.

Let X be an effective topological space with effective base $(U_k)_{k=1}^\infty$. As described in the previous paragraph, we define the structure of an effective topological space on the power set X^n , $n \in \mathbb{N}$; let the effective base in X^n be $(U_k^n)_{k=1}^\infty$. For $n = 0$, X^0 is the trivial one-element effective topological space with all $U_k = X^0$, $k \in \mathbb{N}$. The set X^* of all finite sequences of elements of X is equipped with the topology of the direct sum of X^n , $n \geq 0$. An effective base in it can be defined by $U_{f(n,k)} := U_k^n$, where $f : (\mathbb{N} \cup \{0\}) \times \mathbb{N} \rightarrow \mathbb{N}$ is a computable bijection.

Let X be a fixed effective topological space with effective base $(U_k)_{k=1}^\infty$. An open set $G \subseteq X$ is said to be *effectively open* if it can be represented in the form $\cup\{U_k \mid k \in A\}$ for a recursively enumerable set $A \subseteq \mathbb{N}$. In the main part of this paper, for any effectively open set G we will only consider its representations $\cup\{U_k \mid k \in A\}$ such that

$$\overline{U_k} \subseteq G; \quad (11)$$

this can be done without loss of generality for all specific effective topological spaces that we will need. A *computable sequence of open sets* is a sequence of open sets $G_1, G_2, \dots$ such that there exists a recursively enumerable set $A \subseteq \mathbb{N}^2$ satisfying $G_m = \cup\{U_k \mid (m, k) \in A\}$ for all $m \in \mathbb{N}$. A *computable family of sequences of open sets* is a family $(G_{l,m})$, $l, m \in \mathbb{N}$, of sequences of open sets such that there exists a recursively enumerable set $A \subseteq \mathbb{N}^3$ satisfying $G_{l,m} = \cup\{U_k \mid (l, m, k) \in A\}$ for all l, m . The existence of a universal Turing machine immediately implies

Lemma 5. *There exists a computable family $(G_{l,m})$ of sequences of open sets such that for any computable sequence G'_m of open sets there exists $l \in \mathbb{N}$ such that $G'_m = G_{l,m}$ for all $m \in \mathbb{N}$.*

Any computable family of sequences of open sets satisfying the condition in Lemma 5 will be called a *universal computable family of sequences of open sets*.

A function $f : X \rightarrow \mathbb{R} \cup \{\infty\}$ is called *lower semicomputable* if the set $\{(x, r) \mid x \in X, r \in \mathbb{R}, f(x) > r\}$ is effectively open in $X \times \mathbb{R}$. Similarly, a function $f : X \rightarrow \overline{\mathbb{N}}_0$ is *lower semicomputable* if the set $\{(x, r) \mid x \in X, r \in \mathbb{N}, f(x) \geq r\}$ is effectively open in $X \times \mathbb{N}$. A sequence $f_1, f_2, \dots$ of lower semicomputable functions $f_l : X \rightarrow \mathbb{R} \cup \{\infty\}$ is called *computable* if the set $\{(l, x, r) \mid x \in X, r \in \mathbb{R}, f_l(x) > r\}$ is effectively open in $\mathbb{N} \times X \times \mathbb{R}$. The existence of a universal Turing machine also implies

Lemma 6. *There exists a computable sequence $f_1, f_2, \dots$ of lower semicomputable functions $f_l : X \rightarrow \mathbb{R} \cup \{\infty\}$ that contains every lower semicomputable function.*

Any computable sequence of lower semicomputable functions satisfying the condition in Lemma 6 will be called a *universal computable sequence of lower semicomputable functions*.

It is not difficult to check that the notion of lower semicomputability is an effective version of the standard topological notion of lower semicontinuity: a function $f : X \rightarrow \mathbb{R} \cup \{\infty\}$ is lower semicontinuous (in the usual sense of $\{x \in X \mid f(x) > r\}$ being open for each $r \in \mathbb{R}$) if and only if the set $\{(x, r) \mid x \in X, r \in \mathbb{R}, f(x) > r\}$ is open in the product $X \times \mathbb{R}$.

Lemma 7. *If $f : X \rightarrow [0, \infty]$ is lower semicomputable and $C \in \mathbb{N}$, the set $\{x \mid f(x) > C\}$ is effectively open.*

Proof. Let $\cup\{U_k \mid k \in A\}$, with $A \subseteq \mathbb{N}$ recursively enumerable, be a representation of the effectively open set $\{(x, r) \mid x \in X, r \in \mathbb{R}, f(x) > r\}$ as a union of basic sets in $X \times \mathbb{R}$. The set $\{x \mid f(x) > C\}$ can be represented as the union of the basic sets $\{x \in X \mid \exists r \in \mathbb{R} : (x, r) \in U_k\}$ over $k \in A$ such that $\sup\{r \mid \exists x : (x, r) \in U_k\} > C$. $\square$

A function $f : X \rightarrow \mathbb{R}$ is called *computable* if both f and $-f$ are lower semicomputable. It is easy to see that the analogue of Lemma 6 does not hold for computable functions.

Some Sufficient Conditions on an Arbitrary Class of Stochastic Processes for the Existence of a Predictor

Daniil Ryabko

INRIA Lille–Nord Europe, France
daniil@ryabko.net

Abstract. We consider the problem of sequence prediction in a probabilistic setting. Let there be given a class $\mathcal{C}$ of stochastic processes (probability measures on the set of one-way infinite sequences). We are interested in the question of what are the conditions on $\mathcal{C}$ under which there exists a predictor (also a stochastic process) for which the predicted probabilities converge to the correct ones if any of the processes in $\mathcal{C}$ is chosen to generate the data. We find some sufficient conditions on $\mathcal{C}$ under which such a predictor exists. Some of the conditions are asymptotic in nature, while others are based on the local (truncated to first observations) behaviour of the processes. The conditions lead to constructions of the predictors. In some cases we obtain rates of convergence that are optimal up to an additive logarithmic term. We emphasize that the framework is completely general: the stochastic processes considered are not required to be i.i.d., stationary, or to belong to some parametric family.

1 Introduction

Given a finite sequence $x_1, \dots, x_n$ of observations $x_i \in \mathcal{X}$, where $\mathcal{X}$ is a finite set, we want to predict what are the probabilities of observing $x_{n+1} = x$ for each $x \in \mathcal{X}$. It is assumed that the sequence is generated by some unknown stochastic process μ , a probability measure on the set of one-way infinite sequences $\mathcal{X}^\infty$. The goal is to have a predictor such that the difference between the predicted and correct probabilities goes to zero (in some sense). In general this goal is impossible to achieve if nothing is known about the measure μ generating the sequence. In other words, one cannot have a predictor whose error goes to zero for any measure μ . However, if μ is known to belong to a certain class $\mathcal{C}$ of measures, some well-known results establish the existence of a predictor.

In particular, the Laplace predictor

$$\rho_L(x_{n+1} = a | x_1, \dots, x_n) = \frac{\#\{i \leq n : x_i = a\} + 1}{n + |\mathcal{X}|}$$

predicts any i.i.d. process, that is, predicted probabilities converge to “true” probabilities if the measure generating the sequence is an i.i.d. process. Based

on similar ideas a predictor can be constructed for the class of all k -order Markov measures, and, moreover, such predictors can be combined [9] to form a predictor for the class of all stationary processes over $\mathcal{X}^\infty$. As another example, one can construct a predictor for any given countable class of measures, as shown by Solomonoff's construction of a predictor [15] for the class of all semi-computable measures.

Thus there are examples of classes of processes for which a predictor is known to exist. These examples cover some cases interesting theoretically or important from the application point of view. On the other hand, a trivial negative example is a class of all deterministic sequences (that is, each measure in the class produces a certain sequence of outcomes with probability 1) for which a predictor does not exist: for any predictor there is a measure in the class (a deterministic sequence) on which the predicted probabilities differ from the "true" ones by at least $1/2$ on every step.

The question we are addressing in this work is: in general, for which classes $\mathcal{C}$ of stochastic processes there exists a predictor that predicts every measure in the class?

Motivation. The importance of this question stems primarily from the fact that, interesting as the studied cases are, their motivation originally comes either from some specific applications or from the theoretical attractiveness of the corresponding assumptions. Since new and new applications for the problem of sequence prediction constantly come to existence, known theoretical models can be unsuitable for some of them. For example, stationary processes may model well some physical phenomena but may be less suited for the analysis of DNA sequences. If one had a tool to check feasibility of different theoretical assumptions (that is, to check whether there is a predictor that predicts every process satisfying these assumptions) one could use it to find a better model for each specific application.

Prior work. Apart from the results on the examples of classes $\mathcal{C}$ mentioned above (i.i.d., finite-memory, stationary, computable), this general question (at least for sequence prediction) has received little attention. A related question has been addressed in [6]: Whether, given a class of measures $\mathcal{C}$ and a prior ("meta"-measure) λ over this class of measures, the conditional probabilities of a Bayesian mixture of the class $\mathcal{C}$ w.r.t. λ converge to the true μ -conditional probabilities (weakly merge, in terminology of [6]) for λ -almost any measure μ in $\mathcal{C}$. The answer found in [6] is a set of necessary and sufficient conditions on the measure given by the mixture of the class $\mathcal{C}$ w.r.t. λ under which prediction is possible. The major difference from the general question we posed above is that we do not wish to assume that we have a measure on our class of measures. For large (non-parametric) classes of measures it may not be intuitive which measure over it is natural; rather, the question is whether a "natural" measure which can be used for prediction exists.

Another related question is formulated as a question about two individual measures rather than a class of measures and a predictor. Namely, one can ask

under which conditions one stochastic process predicts another. In [2] it was shown that if one measure is absolutely continuous with respect to another, than the latter predicts the former (the conditional probabilities converge in a very strong sense). In [12] a weaker form of convergence of probabilities (in particular, convergence of expected average KL divergence) is obtained under weaker assumptions.

Measuring prediction quality. As it was mentioned, we are interested in probabilities of observing $x_{n+1} = x$, $x \in \mathcal{X}$ conditional on $x_1, \dots, x_n$. Such conditional probabilities, if specified for every $x_1, \dots, x_n$ also define a probability measure over $\mathcal{X}^\infty$. Thus a predictor (for a class of stochastic processes) is also a stochastic process. The quality of prediction is measured as the discrepancy between the predicted and “true” conditional probabilities. In this work we are mainly considering the Kullback-Leibler divergence between conditional probabilities, averaged over time, which is either required to converge to zero in expectation over $x_1, \dots, x_n$ (expectation being taken with respect to the “true” measure generating the sequence), or with probability 1 (again with respect to the measure generating the sequence). Thus, we are interested in the conditions on a class $\mathcal{C}$ of measures, under which there exists a measure $\rho_{\mathcal{C}}$ such that the average KL divergence between ρ and μ conditional probabilities goes to zero for every $\mu \in \mathcal{C}$, in μ -expectation or with μ -probability 1.

The results. In the present work we exhibit some sufficient conditions on the class $\mathcal{C}$ under which this is possible; none of these conditions relies on a parametrization of any kind. The conditions presented are of two types: conditions on asymptotic behaviour of measures in $\mathcal{C}$, and on their local (restricted to first n observations) behaviour. Conditions of the first type concern separability of $\mathcal{C}$ with respect to the expected average KL divergence. We show that such separability is sufficient for the existence of a predictor.

The conditions of the second kind concern the “capacity” of the set $\mathcal{C}^n := \{\mu^n : \mu \in \mathcal{C}\}$ where μ^n is the measure μ restricted to the first n observations. Intuitively, if $\mathcal{C}^n$ is small in some sense then prediction is possible. We measure the capacity in two ways. The first way is to find the maximum probability given to each sequence $x_1, \dots, x_n$ by some measure in the class, and then take a sum over $x_1, \dots, x_n$. Denoting the obtained quantity c_n , one can show that it grows polynomially in n for some important classes of processes, such as i.i.d. or Markov processes. We show that, in general, if c_n grows subexponentially then a predictor exists that predicts any measure in $\mathcal{C}$ in expected average KL divergence. On the other hand, exponentially growing c_n are not sufficient for prediction. Under slightly stronger conditions on the speed of growth of c_n , we also establish the existence of a measure that predicts every process μ in $\mathcal{C}$ in average KL divergence with μ -probability 1 (rather than in expectation).

A more refined way to measure the capacity of $\mathcal{C}^n$ is using a concept of channel capacity from Information Theory, which was developed for a closely related problem of finding optimal codes for a class of sources. We extend corresponding results from Information Theory to show that sublinear growth of channel capacity is

sufficient for the existence of a predictor, in the sense of expected average divergence. Moreover, the obtained bounds on the divergence are optimal up to an additive logarithmic term.

2 Preliminaries

We consider stochastic processes (probability measures) on the set of one-way infinite sequences $\mathcal{X}^\infty$ where $\mathcal{X}$ is a finite set (alphabet). In the examples we will often assume $\mathcal{X} = \{0, 1\}$. The symbol μ is reserved for the “true” measure generating the sequence. We use $\mathbf{E}_\nu$ for expectation with respect to a measure ν and simply $\mathbf{E}$ for $\mathbf{E}_\mu$ (expectation with respect to the “true” measure generating the sequence).

To measure the quality of prediction we will mainly use quantities which are based on the Kullback-Leibler (KL) divergence. For two probability distributions ν_1 and ν_2 on a finite set $\mathcal{X}$ the *KL divergence* $d(\nu_1, \nu_2)$ is defined as

$$d(\nu_1, \nu_2) = \sum_{x \in \mathcal{X}} \nu_1(x) \log \frac{\nu_1(x)}{\nu_2(x)}. \quad (1)$$

The quality of prediction can be measured as time-average KL divergence between the forecast and the true probabilities. Thus for a sequence $(x_1, \dots, x_n) \in \mathcal{X}^n$ the *average KL divergence* between μ and ρ is defined as

$$\bar{d}_n(\mu, \rho | x_1, \dots, x_n) = \frac{1}{n} \sum_{t=1}^n d(\mu(\cdot | x_1, \dots, x_{t-1}), \rho(\cdot | x_1, \dots, x_{t-1})), \quad (2)$$

where $\mu(\cdot | x_1, \dots, x_{t-1})$ is the probability distribution of the t th member of the sequence conditional on $x_1, \dots, x_{t-1}$.

We say that ρ predicts μ in *average KL divergence* if

$$\bar{d}_n(\mu, \rho, x_1, \dots, x_n) \rightarrow 0 \text{ } \mu\text{-a.s.},$$

and ρ predicts μ in *expected average KL divergence* if

$$\mathbf{E}_\mu \bar{d}_n(\mu, \rho, x_1, \dots, x_n) \rightarrow 0.$$

We also define *asymptotic expected KL divergence* between measures μ_1 and μ_2 as

$$D(\mu, \rho) = \limsup_{n \rightarrow \infty} \mathbf{E}_\mu \bar{d}_n(\mu, \rho, x_1, \dots, x_{n-1}).$$

We will often omit the argument $x_1, \dots, x_n$ from our notation.

3 Main Results

Asymptotic conditions. Call a class $\mathcal{C}$ of stochastic processes *separable with respect to (asymptotic expected KL divergence) D* if there is a countable set $M \subset \mathcal{C}$ with the following property: For every $\mu \in \mathcal{C}$ and every $\varepsilon > 0$ there is $\mu_\varepsilon \in M$ such that $D(\mu, \mu_\varepsilon) \leq \varepsilon$.

Theorem 1. *If $\mathcal{C}$ is separable with respect to D then there exists a measure ρ such that ρ predicts every $\mu \in \mathcal{C}$ in expected average KL divergence*

$$D(\mu, \rho) = 0$$

for every $\mu \in \mathcal{C}$.

The predictor ρ constructed in the proof (below) is a weighted average of the countable dense set whose existence is asserted in the statement. This idea of taking a weighted mixture ρ of a countable class M and then bounding the KL divergence of a measure outside M with ρ by its divergence with a measure in M plus a constant ($\log^{-1}$ of the weight), has appeared in many places, starting from [9]; in particular the general scheme appears in [5, Section 3.2.8]. For parametric families of processes [5] also exhibits the idea of constructing a predictor by taking a mixture over a countable dense (with respect to the parametrisation) subset.

Proof. Let $w_k, k \in \mathbb{N}$ be a sequence of positive reals that sum to 1, e.g. $w_k = 2^{-k}$. Since the set M is countable we can introduce $\mu_i, i \in \mathbb{N}$ such that $M = \{\mu_i : i \in \mathbb{N}\}$. Define the predictor ρ as $\rho = \sum_{i \in \mathbb{N}} w_i \mu_i$. We have to show that

$$\lim_{n \rightarrow \infty} \mathbf{E}_\mu d_n(\mu, \rho) = 0$$

for every $\mu \in \mathcal{C}$. Fix any $\mu \in \mathcal{C}$ and $\varepsilon > 0$. Find $\mu_k \in M$ such that $D(\mu, \mu_k) \leq \varepsilon$. Introduce the symbol $\mathbf{E}^t$ for μ -expectation over x_t conditional on $x_1, \dots, x_{t-1}$. We have

$$\begin{aligned} \mathbf{E}_\mu \bar{d}_n(\mu, \rho) &= \frac{1}{n} \mathbf{E} \sum_{t=1}^n \sum_{x_t \in \mathcal{X}} \mu(x_t | x_1, \dots, x_{t-1}) \log \frac{\mu(x_t | x_1, \dots, x_{t-1})}{\rho(x_t | x_1, \dots, x_{t-1})} \\ &= \frac{1}{n} \sum_{t=1}^n \mathbf{E} \mathbf{E}^t \log \frac{\mu(x_t | x_1, \dots, x_{t-1})}{\rho(x_t | x_1, \dots, x_{t-1})} = \frac{1}{n} \mathbf{E} \log \prod_{t=1}^n \frac{\mu(x_t | x_1, \dots, x_{t-1})}{\rho(x_t | x_1, \dots, x_{t-1})} \\ &= \frac{1}{n} \mathbf{E} \log \frac{\mu(x_1, \dots, x_n)}{\rho(x_1, \dots, x_n)} \leq \frac{1}{n} \mathbf{E} \log \frac{\mu(x_1, \dots, x_n)}{w_k \mu_k(x_1, \dots, x_n)} \\ &= \frac{\log w_k^{-1}}{n} + \frac{1}{n} \mathbf{E} \log \frac{\mu(x_1, \dots, x_n)}{\mu_k(x_1, \dots, x_n)} \\ &= \frac{\log w_k^{-1}}{n} + \mathbf{E}_\mu \bar{d}_n(\mu, \mu_k), \quad (3) \end{aligned}$$

from which we conclude that

$$\limsup_{n \rightarrow \infty} \mathbf{E}_\mu \bar{d}_n(\mu, \rho) \leq \limsup_{n \rightarrow \infty} \mathbf{E}_\mu \bar{d}_n(\mu, \mu_k) \leq \varepsilon.$$

Since this holds for every ε , and since KL divergence is always non-negative, we get the statement $\lim_{n \rightarrow \infty} \mathbf{E}_\mu \bar{d}_n(\mu, \rho) = 0$. $\square$

Example: countable classes. A trivial but interesting example in which the conditions of Theorem 1 are satisfied is when the class $\mathcal{C}$ itself is countable. A well-studied case is when $\mathcal{C}$ is the class of all (semi-)computable measures ([15], see also [5]).

Example: i.i.d. Another simple example is given by the class $\mathcal{C}_B$ of all i.i.d. processes, with $\mathcal{X} = \{0, 1\}$, indexed by the parameter $p \in [0, 1]$; that is, $\mu_p(x_n = 0) = p$ for all n independently of each other. In this case, it is easy to check that the subset of all processes with rational parameters is dense in $\mathcal{C}_B$ with respect to the expected average KL divergence, since $\bar{d}_n(\mu_p, \mu_q) = d(\mu_p, \mu_q)$ and the latter is continuous in p and q .

Example: Finite-memory, stationary. The same holds for the class of stationary finite memory processes: each process with memory k is parametrized by a finite number of parameters — the conditional probabilities of observing $x_{k+1} = x \in \mathcal{X}$ given $x_1, \dots, x_k$ and the initial distribution. The set of processes with rational values of the parameters is dense with respect to the expected average divergence. Since any stationary ergodic process can be arbitrary well approximated (in the sense of asymptotic expected average KL divergence $D(\mu, \rho)$, where $\limsup$ actually becomes $\lim$) by finite-memory processes, in particular by those with rational parameters, we can conclude that the class of stationary ergodic sources is separable with respect to expected average KL divergence. Thus, applying Theorem 1 we can derive the result of [9] which says that there exists a predictor for the class of all stationary ergodic processes.

Conditions based on local behaviour of measures. Next we provide some sufficient conditions for the existence of a predictor based on local characteristics of the class of measures.

For a class $\mathcal{C}$ of stochastic processes and a sequence $(x_1, \dots, x_n) \in \mathcal{X}^n$ introduce the coefficients

$$c_{x_1, \dots, x_n}(\mathcal{C}) = \sup_{\mu \in \mathcal{C}} \mu(x_1, \dots, x_n). \quad (4)$$

Define also the normalizer

$$c_n(\mathcal{C}) = \sum_{(x_1, \dots, x_n) \in \mathcal{X}^n} c_{x_1, \dots, x_n}(\mathcal{C}). \quad (5)$$

A *normalized maximum likelihood* estimator λ is defined as

$$\lambda_{\mathcal{C}}(x_1, \dots, x_n) = \frac{1}{c_n(\mathcal{C})} c_{x_1, \dots, x_n}(\mathcal{C}). \quad (6)$$

For finite spaces (that is, for fixed n) normalized maximum likelihood estimators have been studied in e.g. [14, 17], in the context of Information Theory. However, the family $\lambda_{\mathcal{C}}(x_1, \dots, x_n)$ (indexed by n) in general does not define a stochastic process over $\mathcal{X}^\infty$ ($\lambda_{\mathcal{C}}$ are not consistent for different n); thus, in particular, using average KL divergence for measuring prediction quality would not make sense, since

$$d_n(\mu(\cdot | x_1, \dots, x_{n-1}), \lambda_{\mathcal{C}}(\cdot | x_1, \dots, x_{n-1}))$$

can be negative, as the following example shows.

Example: negative d_n for NML estimates. Let the processes $\mu_i, i \in \{1, \dots, 4\}$ be defined on the steps $n = 1, 2$ as follows. $\mu_1(00) = \mu_2(01) = \mu_4(11) = 1$, while

$\mu_3(01) = \mu_3(00) = 1/2$. We have $\lambda_C(1) = \lambda_C(0) = 1/2$, while $\lambda_C(00) = \lambda_C(01) = \lambda_C(11) = 1/3$. If we define $\lambda_C(x|y) = \lambda_C(yx)/\lambda_C(y)$ we get $\lambda_C(1|0) = \lambda_C(0|0) = 2/3$. Then $d_2(\mu_3(\cdot|0), \lambda_C(\cdot|0)) = \log 3/4 < 0$.

Yet, by taking an appropriate mixture, it is still possible to construct a predictor (a stochastic process) based on λ that predicts the measures in the class not only in expectation but, under certain conditions, also with probability 1.

Definition 1 (predictor ρ_c). Let $w := \sum_{k=1}^{\infty} \frac{1}{k^2}$ and let $w_k := \frac{1}{wk^2}$. Define a measure μ_k as follows. On first k steps it is defined as λ_C , and for $n > k$ it outputs only zeros with probability 1; so, $\mu_k(x_1, \dots, x_k) = \lambda_C(x_1, \dots, x_k)$ and $\mu_k(x_n = 0) = 1$ for $n > k$. Define the measure ρ_c as

$$\rho_c = \sum_{k=1}^{\infty} w_k \mu_k. \quad (7)$$

Thus we have taken the normalized maximum likelihood estimates λ_n for each n and continued them arbitrarily (actually by a deterministic sequence) to obtain a sequence of measures on $\mathcal{X}^{\infty}$ that can be summed.

Theorem 2. For a class $\mathcal{C}$ of stochastic processes the predictor ρ_c defined above satisfies

$$\mathbf{E}_{\mu} \bar{d}_n(\mu, \rho_c) \leq \frac{\log c_n(\mathcal{C})}{n} + O\left(\frac{\log n}{n}\right); \quad (8)$$

in particular if

$$\log c_n(\mathcal{C}) = o(n). \quad (9)$$

then ρ_c predicts every $\mu \in \mathcal{C}$ in expected average KL divergence. If the coefficients $c_n(\mathcal{C})$ are such that

$$\sum_{n=1}^{\infty} \frac{\log^2 c_n(\mathcal{C})}{n^2} < \infty \quad (10)$$

then ρ_c predicts every $\mu \in \mathcal{C}$ in average KL divergence (with μ -probability 1).

Proof. Since for each $x_1, \dots, x_n$ and each $\mu \in \mathcal{C}$ we have

$$\rho_c(x_1, \dots, x_n) \geq wn^2 c_n(\mathcal{C}) \mu(x_1, \dots, x_n)$$

the theorem can be deduced from [12, Theorems 4,5]; we give here a full proof for the sake of completeness.

For the first statement, we have (similarly to the proof of Theorem 1)

$$\begin{aligned} \mathbf{E}_{\mu} \bar{d}_n(\mu, \rho_c) &= \frac{1}{n} \mathbf{E} \log \frac{\mu(x_1, \dots, x_n)}{\rho_c(x_1, \dots, x_n)} \leq \frac{1}{n} \mathbf{E} \log \frac{\mu(x_1, \dots, x_n)}{wn\mu_n(x_1, \dots, x_n)} \\ &\leq \frac{1}{n} \log \frac{c_n(\mathcal{C})}{w_n} = \frac{1}{n} (\log c_n(\mathcal{C}) + 2 \log n + \log w). \end{aligned} \quad (11)$$

In order to prove the second statement, we first introduce a short-hand notation $x_{1..n}$ for $x_1, \dots, x_n$. Consider the random variables

$$l_n = \log \frac{\mu(x_n | x_{1..n-1})}{\rho_c(x_n | x_{1..n-1})}$$

and

$$\bar{l}_n = \frac{1}{n} \sum_{t=1}^n l_t.$$

Observe that $d_n = \mathbf{E}^n l_n$, so that the random variables $m_n := l_n - d_n$ form a martingale difference sequence (that is, $\mathbf{E}^n m_n = 0$) with respect to the standard filtration defined by $x_1, \dots, x_n, \dots$. Let also $\bar{m}_n = \frac{1}{n} \sum_{t=1}^n m_t$. We will show that $\bar{m}_n \rightarrow 0$ μ -a.s. and $\bar{l}_n \rightarrow 0$ μ -a.s. which implies $\bar{d}_n \rightarrow 0$ μ -a.s.

Note that

$$\bar{l}_n = \frac{1}{n} \log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} \leq \frac{\log w_n^{-1} c_n(\mathcal{C})}{n} \rightarrow 0.$$

Thus to show that $\bar{l}_n$ goes to 0 we need to bound it from below. It is easy to see that $n\bar{l}_n$ is (μ -a.s.) bounded from below by a constant, since $\frac{\rho_c(x_{1..n})}{\mu(x_{1..n})}$ is a positive μ -supermartingale, which converges to a finite limit μ -a.s. by Doob's submartingale convergence theorem, see e.g. [13, p.508].

Next we will show that $\bar{m}_n \rightarrow 0$ μ -a.s. We have

$$\begin{aligned} m_n &= \log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} - \log \frac{\mu(x_{1..n-1})}{\rho_c(x_{1..n-1})} - \mathbf{E}^n \log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} + \mathbf{E}^n \log \frac{\mu(x_{1..n-1})}{\rho_c(x_{1..n-1})} \\ &= \log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} - \mathbf{E}^n \log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})}. \end{aligned}$$

Let $f(n)$ be some function monotonically increasing to infinity such that

$$\sum_{n=1}^{\infty} \frac{(\log w_n^{-1} c_n(\mathcal{C}) + f(n))^2}{n^2} < \infty \quad (12)$$

(e.g. choose $f(n) = \log n$). For a sequence of random variables λ_n define

$$(\lambda_n)^{+(f)} = \begin{cases} \lambda_n & \text{if } \lambda_n \geq -f(n) \\ 0 & \text{otherwise} \end{cases}$$

and $\lambda_n^{-(f)} = \lambda_n - \lambda_n^{+(f)}$. Introduce also

$$m_n^+ = \left(\log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} \right)^{+(f)} - \mathbf{E}^n \left(\log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} \right)^{+(f)},$$

$m_n^- = m_n - m_n^+$ and the averages $\bar{m}_n^+$ and $\bar{m}_n^-$. Observe that m_n^+ is a martingale difference sequence. Hence to establish the convergence $\bar{m}_n^+ \rightarrow 0$ we can use

the martingale strong law of large numbers [13, p.501], which states that, for a martingale difference sequence γ_n , if $\mathbf{E}(n\bar{\gamma}_n)^2 < \infty$ and $\sum_{n=1}^{\infty} \mathbf{E}\gamma_n^2/n^2 < \infty$ then $\bar{\gamma}_n \rightarrow 0$ a.s. Indeed, for m_n^+ the first condition is trivially satisfied (since the expectation in question is a finite sum of finite numbers), and the second follows from the fact that

$$|m_n^+| \leq \log w_n^{-1} c_n(\mathcal{C}) + f(n)$$

and (12).

Furthermore, we have

$$m_n^- = \left(\log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} \right)^{-(f)} - \mathbf{E}^n \left(\log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} \right)^{-(f)}.$$

As it was mentioned before, $\log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})}$ converges μ -a.s. either to (positive) infinity or to a finite number. Hence

$$\left(\log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} \right)^{-(f)}$$

is non-zero only a finite number of times, and so its average goes to zero. To see that $\mathbf{E}^n \left(\log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} \right)^{-(f)} \rightarrow 0$ we write

$$\begin{aligned} \mathbf{E}^{n+1} \left(\log \frac{\mu(x_{1..n+1})}{\rho_c(x_{1..n+1})} \right)^{-(f)} \\ = \sum_{x_n \in \mathcal{X}} \mu(x_{n+1}|x_{1..n}) \left(\log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} + \log \frac{\mu(x_{n+1}|x_{1..n})}{\rho_c(x_{n+1}|x_{1..n})} \right)^{-(f)} \\ \geq \sum_{x_n \in \mathcal{X}} \mu(x_{n+1}|x_{1..n}) \left(\log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} + \log \mu(x_{n+1}|x_{1..n}) \right)^{-(f)} \end{aligned}$$

and note that the first term in brackets is bounded from below, and so for the sum in brackets to be less than $-f(n+1)$ (which is unbounded) the second term $\log \mu(x_n|x_{1..n})$ has to go to $-\infty$, but then the expectation goes to zero since $\lim_{u \rightarrow 0} u \log u = 0$.

Thus we conclude that $\bar{m}_n^- \rightarrow 0$ μ -a.s., which together with $\bar{m}_n^+ \rightarrow 0$ μ -a.s. implies $\bar{m}_n \rightarrow 0$ μ -a.s., which, finally, together with $\bar{l}_n \rightarrow 0$ μ -a.s. implies $\bar{d}_n \rightarrow 0$ μ -a.s. $\square$

Example: finite-memory. To illustrate the applicability of the theorem we first consider the class of i.i.d. processes $\mathcal{C}_B$ over the binary alphabet $\mathcal{X} = \{0, 1\}$. It is easy to see that for each $x_1, \dots, x_n$

$$\sup_{\mu \in \mathcal{C}_B} \mu(x_1, \dots, x_n) = p^k (1-p)^{n-k}$$

where $k = \#\{i \leq n : x_i = 0\}$ is the number of 0s in $x_1, \dots, x_n$ and $p = k/n$. For the constants $c_n(\mathcal{C})$ we can get the bound $c_n(\mathcal{C}) \leq n + 1$. In general, for the class $\mathcal{C}_k$ of processes with memory k over a finite space $\mathcal{X}$ we get polynomial $c_n(\mathcal{C})$ (see e.g. [11]).

Thus, with respect to finite-memory processes, the conditions of Theorem 2 leave ample space for the growth of $c_n(\mathcal{C})$: the condition (9) allows any subexponential growth of $c_n(\mathcal{C})$ and the condition (10) allows for example $c_n(\mathcal{C}) = 2^{\sqrt{n}/\log n}$.

Example: exponential coefficients are not sufficient. Observe that the condition (9) cannot be relaxed further, in the sense that exponential coefficients c_n are not sufficient for prediction. Indeed, for the class of all deterministic processes (that is, each process from the class produces some fixed sequence of observations with probability 1) we have $c_n = 2^n$, while obviously for this class a predictor does not exist.

Optimal rates of convergence. A natural question that arises with respect to the bound (8) is whether it is optimal. This question is closely related to the optimality of the normalized maximum likelihood estimates used in the construction of the predictor. In general, since such estimates are not optimal neither are the rates of convergence in (8). To obtain (close to) optimal rates one has to consider a different measure of capacity.

To do so, we make the following connection to a problem in Information Theory. Let $\mathcal{P}(\mathcal{X}^\infty)$ be the set of all stochastic processes (probability measures) on the set $\mathcal{X}^\infty$, and let $\mathcal{P}(\mathcal{X})$ be the set of probability distributions over a (finite) set $\mathcal{X}$. For a class $\mathcal{C}$ of measures we are interested in a predictor that has a small (or minimal) worst-case (with respect to the class $\mathcal{C}$) probability of error. Thus, we are interested in the quantity

$$\inf_{\rho \in \mathcal{P}(\mathcal{X}^\infty)} \sup_{\mu \in \mathcal{C}} D(\mu, \rho), \quad (13)$$

where the infimum is taken over all stochastic processes ρ , and D is the asymptotic expected average KL divergence. (In particular, we are interested in the conditions under which the quantity in (13) equals zero.) This problem has been studied for the case when the probability measures are over a finite set $\mathcal{X}$, and D is replaced simply by the KL divergence d between the measures. Thus, the problem was to find the probability measure ρ (if it exists) on which the following minimax is attained

$$R(A) := \inf_{\rho \in \mathcal{P}(\mathcal{X})} \sup_{\mu \in A} d(\mu, \rho), \quad (14)$$

where $A \subset \mathcal{P}(\mathcal{X})$. This problem is closely related to the problem of finding the best code for the class of sources A , which was its original motivation. The normalized maximum likelihood distribution considered above does not in general lead to the optimum solution for this problem. The optimum solution is obtained through the result that relates the minimax (14) to the so-called channel

capacity. For a set A of measures on a finite set $\mathcal{X}$ the *channel capacity* of A is defined as

$$C(A) := \sup_{P \in \mathcal{P}_0(A)} \sum_{\mu \in S(P)} P(\mu) d(\mu, \rho_P), \quad (15)$$

where $\mathcal{P}_0(A)$ is the set of all probability distributions on A that have a finite support $S(P)$, and $\rho_P = \sum_{\mu \in S(P)} P(\mu) \mu$. It is shown in [8, 3] that $C(A) = R(A)$, thus reducing the problem of finding a minimax to an optimization problem. Moreover, an algorithm [1] exists for approximating $C(A)$ numerically and solving the optimization problem for the important case when A is the convex hull of a finite set. For probability measures over infinite spaces this result ($R(A) = C(A)$) was generalized in [4], but the divergence between probability distributions is measured by KL divergence (and not average KL divergence), which gives infinite $R(A)$ for most of the cases interesting from the sequence prediction point of view (e.g. for the class of i.i.d. processes).

However, truncating measures in a class $\mathcal{C}$ to the first n observations, we can use the results about channel capacity to analyze the predictive properties of the class. Moreover, the rates of convergence that can be obtained along these lines are close to optimal. In order to pass from measures minimizing the divergence for each individual n to a process that minimizes the divergence for all n we use the same idea as when constructing the process ρ_C .

Theorem 3. *Let $\mathcal{C}$ be a class of measures over $\mathcal{X}^\infty$ and $\mathcal{C}^n$ be the class of measures from $\mathcal{C}$ restricted to $\mathcal{X}^n$. There exists a measure ρ_C such that*

$$\mathbf{E}_\mu \bar{d}_n(\mu, \rho_C, x_1, \dots, x_n) \leq \frac{C(\mathcal{C}^n)}{n} + O\left(\frac{\log n}{n}\right). \quad (16)$$

(in particular, if $C(\mathcal{C}^n)/n \rightarrow 0$ then ρ_C predicts every $\mu \in \mathcal{C}$ in expected average KL divergence). Moreover, for any measure ρ_C and every $\varepsilon > 0$ there exists $\mu \in \mathcal{C}$ such that

$$\mathbf{E}_\mu \bar{d}_n(\mu, \rho_C, x_1, \dots, x_n) \geq \frac{C(\mathcal{C}^n)}{n} - \varepsilon.$$

Proof. As shown in [3], for each n there exists a sequence ν_k^n , $k \in \mathbb{N}$ of measures on $\mathcal{X}^n$ such that

$$\lim_{k \rightarrow \infty} \sup_{\mu^n \in \mathcal{C}^n} d(\mu^n, \nu_k^n) \rightarrow C(\mathcal{C}^n).$$

For each $n \in \mathbb{N}$ find an index k_n such that

$$\left| \sup_{\mu^n \in \mathcal{C}^n} d(\mu^n, \nu_{k_n}^n) - C(\mathcal{C}^n) \right| \leq 1.$$

Define the measure ρ_n as follows. On the first n symbols it coincides with $\nu_{k_n}^n$ and $\rho_n(x_m = 0) = 1$ for $m > n$. Finally, set $\rho_C = \sum_{n=1}^{\infty} w_n \rho_n$, where $w_k = \frac{1}{wn^2}$, $w = \sum_{n=1}^{\infty} \frac{1}{n^2}$. We have to show that $\lim_{n \rightarrow \infty} \mathbf{E}_\mu \bar{d}_n(\mu, \rho_C) = 0$ for every $\mu \in \mathcal{C}$. Indeed,

$$\begin{aligned}
\mathbf{E}_\mu \bar{d}_n(\mu, \rho_C) &= \frac{1}{n} \mathbf{E}_\mu \log \frac{\mu(x_{1..n})}{\rho_C(x_{1..n})} \\
&\leq \frac{\log w_k^{-1}}{n} + \frac{1}{n} \mathbf{E}_\mu \log \frac{\mu(x_{1..n})}{\rho_n(x_{1..n})} \leq \frac{\log w + 2 \log n}{n} + \frac{1}{n} d(\mu^n, \rho_n) \\
&\leq o(1) + \frac{C(\mathcal{C}^n)}{n} + \frac{1}{n}. \quad (17)
\end{aligned}$$

The second statement follows from the fact [8, 3] that $C(\mathcal{C}^n) = R(\mathcal{C}^n)$ (cf. (14)). $\square$

Thus, if the channel capacity $C(\mathcal{C}^n)$ grows sublinearly, a predictor can be constructed for the class of processes $\mathcal{C}$. In this case the problem of constructing the predictor is reduced to finding the channel capacities for different n and finding the corresponding measures on which it is attained or approached.

As an **example** we can mention, again, the class of all i.i.d. processes, whose channel capacity $C(\mathcal{C}_B^n)$ is $O(\log n)$, see e.g. [7].

We also remark that the requirement of a sublinear channel capacity cannot be relaxed, in the sense that a linear channel capacity is not sufficient for prediction, since it is the maximal possible capacity for a set of measures on $\mathcal{X}^n$.

4 Discussion

As far as **algorithmic realizability** of the predictors proposed is concerned, we should first note that when an input parameter of an “algorithm” is an arbitrary class of stochastic processes, one can hardly talk about algorithms for real computers. Rather, the predictors constructed have to be regarded as *reductions* of the problem of finding a predictor for a given class of stochastic processes to the conceptually much easier problems of approximating certain suprema and infinite sums. Here an analogy can be made with the classification problem. In certain cases the problem of finding a good classifier can be reduced to the problem of finding a classifier from a given class that best fits the data (minimizes empirical risk [16]). Conceptually this is a much simpler problem; however, in some cases it can be intractable (see e.g. [10]). In general, for each particular class of classifiers a separate algorithm should be constructed to find efficiently a classifier that fits the data. Indeed, efficient solutions (such as support vector machines [16]) exist for many important cases.

Returning to our problem, Theorem 1 states that if in a class $\mathcal{C}$ of measures there is a countable dense subset M , then a predictor can be constructed whose average expected error goes to zero. Moreover, such a predictor can be obtained as a weighted sum of measures from M (with any positive weights that sum to 1). Thus the problem of finding a predictor is reduced to two (simpler) problems: finding a dense subset and taking an infinite sum. We can further show that in some cases the latter problem can be reduced to a version of the former, that is, it is not necessary to take an infinite sum if one can find a finite ε -net.

Proposition 1. *Let a class $\mathcal{C}$ of stochastic processes be such that for some ε there exists a finite or countable subset $M_\varepsilon \subset \mathcal{C}$ with the following property. For any $\mu \in \mathcal{C}$ there exists $\mu' \in M_\varepsilon$ such that $D(\mu, \mu') \leq \varepsilon$. Then for the measure*

$$\rho = \sum_{\nu \in M_\varepsilon} w_\nu \nu,$$

where w_ν are any positive reals (that sum to 1), we have $D(\mu, \rho) \leq \varepsilon$ for any $\mu \in \mathcal{C}$.

This statement can be proven in exactly the same way as Theorem 1.

The predictors constructed in the proofs of Theorems 2 and 3 also involve summation over an infinite set. However, in these cases it is immediately apparent from the constructions of the predictors that for prediction on n th step it is sufficient to take sums up to n , and the bounds on expected average divergence (8) and (16) still hold. Thus the problem of finding a predictor is reduced to the problem of approximating a finite number of suprema.

Namely, for the case of normalized maximum likelihood predictor of Theorem 2 the quantity (4) have to be evaluated for each n and each $x_1, \dots, x_n$. For the predictor based on channel capacity one has to find the measure on which channel capacity (15) is attained (possibly up to a certain ε_n) for each n . Again, for some important cases efficient algorithms for solving this problem exist, such as [1] for the case when the set $\mathcal{C}^n$ is a convex hull of a finite number of measures.

One more question we discuss is **other possible ways of measuring the quality of prediction**. In this paper we were considering KL divergence (1) averaged over time (2), and have developed predictors on which this divergence tends to zero either in expectation or with probability 1. Other possible ways of measuring divergence include the absolute distance

$$a_n(\mu, \rho | x_{1..n-1}) = \sum_{x \in \mathcal{X}} |\mu(x_n = x | x_{1..n-1}) - \rho(x_n = x | x_{1..n-1})|,$$

the squared absolute distance

$$s_n(\mu, \rho | x_{1..n-1}) = \sum_{x \in \mathcal{X}} (\mu(x_n = x | x_{1..n-1}) - \rho(x_n = x | x_{1..n-1}))^2,$$

and the Hellinger distance

$$h_n(\mu, \rho | x_{1..n-1}) = \sum_{x \in \mathcal{X}} (\sqrt{\mu(x_n = x | x_{1..n-1})} - \sqrt{\rho(x_n = x | x_{1..n-1})})^2.$$

Analogously with the average KL divergence (2) we can define average absolute distance $\bar{a}_n$, average squared absolute distance $\bar{s}_n$ and average Hellinger distance $\bar{h}_n$. Using Pinsker's inequality $a_t^2 \leq 2d_t$ one can easily show that all convergence results and upper bounds stated in terms of KL divergence also hold for the measures of divergence just introduced (see e.g. Lemma 1 of [12] for details).

Proposition 2. *The statements concerning convergence and upper of Theorems 1, 2 and 3 hold true if the average KL distance $\bar{d}_n$ is replaced by either of the following: average absolute distance $\bar{a}_n$, average squared absolute distance $\bar{s}_n$ and average Hellinger distance $\bar{h}_n$.*

Another interesting problem would be to investigate a stronger notion of predictive quality: without averaging over time. For example, under which conditions on a class $\mathcal{C}$ of measures there exist a predictor ρ for which

$$d_n(\mu, \rho | x_1, \dots, x_{n-1})$$

goes to zero with μ probability 1 for every μ .

These questions, along with the problem of finding efficient algorithmic solutions for cases of practical interest, constitute an agenda for further investigation.

References

- [1] Arimoto, S.: An algorithm for computing the capacity of arbitrary discrete memoryless channels. *IEEE Transactions on Information Theory* IT-18, 14–20 (1972)
- [2] Blackwell, D., Dubins, L.: Merging of opinions with increasing information. *Annals of Mathematical Statistics* 33, 882–887 (1962)
- [3] Gallager, R.: *Source Coding With Side Information and Universal Coding* (unpublished) M.I.T. LIDS-P-937 (1976) (revised 1979)
- [4] Haussler, D.: A General Minimax Result for Relative Entropy. *IEEE Transactions on Information Theory* 43(4), 1276–1280 (1997)
- [5] Hutter, M.: *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Springer, Berlin (2005)
- [6] Jackson, M., Kalai, E., Smorodinsky, R.: Bayesian representation of stochastic processes under learning: de Finetti revisited. *Econometrica* 67(4), 794–875 (1999)
- [7] Krichevsky, R.: *Universal Compression and Retrieval*. Kluwer Academic Publishers, Dordrecht (1993)
- [8] Ryabko, B.: Coding of a source with unknown but ordered probabilities. *Problems of Information Transmission* 15(2), 134–138 (1979)
- [9] Ryabko, B.: Prediction of random sequences and universal coding. *Problems of Information Transmission* 24, 87–96 (1988)
- [10] Ryabko, D.: On sample complexity for computational classification problems. *Algorithmica* 49(1), 69–77 (2007)
- [11] Ryabko, D., Hutter, M.: On sequence prediction for arbitrary measures. In: *Proc. 2007 IEEE International Symposium on Information Theory, Nice, France*, pp. 2346–2350 (2007)
- [12] Ryabko, D., Hutter, M.: Predicting Non-Stationary Processes. *Applied Mathematics Letters* 21(5), 477–482 (2008)
- [13] Shiryaev, A.: *Probability*. Springer, Heidelberg (1996)
- [14] Shtarkov, Yu.: Universal sequential coding of single messages. *Problems of Information Transmission* 23, 3–17 (1988)
- [15] Solomonoff, R.J.: Complexity-based induction systems: comparisons and convergence theorems. *IEEE Trans. Information Theory* IT-24, 422–432 (1978)
- [16] Vapnik, V.: *Statistical Learning Theory*. John Wiley & Sons, Inc., New York (1998)
- [17] Xie, Q., Barron, A.: Asymptotic minimax regret for data compression, gambling, and prediction. *IEEE Transactions on Information Theory* 46(2), 431–445 (1997)

Nonparametric Independence Tests: Space Partitioning and Kernel Approaches

Arthur Gretton¹ and László Györfi²

¹ MPI for Biological Cybernetics, Spemannstr. 38, 72076 Tübingen, Germany
`arthur@tuebingen.mpg.de`

² Budapest University of Technology and Economics,
H-1521 Stoczek u. 2, Budapest, Hungary
`gyorfi@szit.bme.hu`

Abstract. Three simple and explicit procedures for testing the independence of two multi-dimensional random variables are described. Two of the associated test statistics (L_1 , log-likelihood) are defined when the empirical distribution of the variables is restricted to finite partitions. A third test statistic is defined as a kernel-based independence measure. All tests reject the null hypothesis of independence if the test statistics become large. The large deviation and limit distribution properties of all three test statistics are given. Following from these results, distribution-free strong consistent tests of independence are derived, as are asymptotically α -level tests. The performance of the tests is evaluated experimentally on benchmark data.

Consider a sample of $\mathbb{R}^d \times \mathbb{R}^{d'}$ -valued random vectors $(X_1, Y_1), \dots, (X_n, Y_n)$ with independent and identically distributed (i.i.d.) pairs defined on the same probability space. The distribution of (X, Y) is denoted by ν , while μ_1 and μ_2 stand for the distributions of X and Y , respectively. We are interested in testing the null hypothesis that X and Y are independent,

$$\mathcal{H}_0 : \nu = \mu_1 \times \mu_2,$$

while making minimal assumptions regarding the distribution.

We consider two main approaches to independence testing. The first is to partition the underlying space, and to evaluate the test statistic on the resulting discrete empirical measures. Consistency of the test must then be verified as the partition is refined for increasing sample size. Previous multivariate hypothesis tests in this framework, using the L_1 divergence measure, include homogeneity tests (to determine whether two random variables have the same distribution, by Biau and Györfi [1]); and goodness-of-fit tests (for whether a random variable has a particular distribution, by Györfi and van der Meulen [2], and Beirlant et al. [3]). The log-likelihood has also been employed on discretised spaces as a statistic for goodness-of-fit testing [4]. We provide generalizations of both the L_1 and log-likelihood based tests to the problem of testing independence, representing to our knowledge the first application of these techniques to independence testing.

We obtain two kinds of tests for each statistic: strong consistent tests¹ based on large deviation bounds, which make no assumptions about the distribution; and tests based on the asymptotic distribution of the test statistic, which assume only that the distribution is nonatomic. We also present a conjecture regarding the form taken by an asymptotic test based on the Pearson χ^2 statistic, using the goodness-of-fit results in [4] (further related test statistics include the power divergence family of Read and Cressie [6], although we do not study them here). The advantage of our test procedures is that, besides being explicit and easy to carry out, they require very few assumptions on the partition sequences, are consistent, and have distribution-independent thresholds.

Our second approach to independence testing is kernel-based. In this case, our test statistic has a number of different interpretations: as an L_2 distance between Parzen window estimates [7], as a smoothed difference between empirical characteristic functions [8, 9], or as the Hilbert-Schmidt norm of a cross-covariance operator mapping between functions of the random variables [10, 11]. Each test differs from the others regarding the conditions required of the kernels: the Parzen window statistic requires the kernel bandwidth to decrease with increasing sample size, and has a different limiting distribution to the remaining two statistics; while the Hilbert-Schmidt approach uses a fixed bandwidth, and can be thought of as a generalization of the characteristic function-based test. We provide two new results: a strong consistent test of independence based on a tighter large deviation bound than that in [10], and an empirical comparison of the limiting distributions of the kernel-based statistic.

Additional independence testing approaches also exist in the statistics literature. For $d = d' = 1$, an early nonparametric test for independence, due to Hoeffding, Blum, Kiefer, and Rosenblatt [12, 13], is based on the notion of differences between the joint distribution function and the product of the marginals. The associated independence test is consistent under appropriate assumptions. Two difficulties arise when using this statistic in a test, however. First, quantiles of the null distribution are difficult to estimate. Second, and more importantly, the quality of the empirical distribution function estimates becomes poor as the dimensionality of the spaces $\mathbb{R}^d$ and $\mathbb{R}^{d'}$ increases, which limits the utility of the statistic in a multivariate setting. Further approaches to independence testing can be used when particular assumptions are made on the form of the distributions, for instance that they should exhibit symmetry. We do not address these approaches in the present study.

The paper is organized as follows. Section 1 describes the large deviation and limit distribution properties of the L_1 -test statistic. The large deviation result is used to formulate a distribution-free strong consistent test of independence, which rejects the null hypothesis if the test statistic becomes large. The limit distribution is used in an asymptotically α -level test, which is consistent when the distribution is nonatomic. Both a distribution-free strong consistent test

¹ A strong consistent test means that both on $\mathcal{H}_0$ and on its complement the test makes a.s. no error after a random sample size. This concept is close to the definition of discernability introduced by Dembo and Peres [5]. See [1] for further discussion.

and an asymptotically α -level test are presented for the log-likelihood statistic in Section 2, and a conjecture for an asymptotically α -level test based on the Pearson χ^2 statistic is described in Section 3. Section 4 contains a review of kernel-based independence statistics, and the associated hypothesis tests for both the fixed-bandwidth and variable-bandwidth cases. Finally, a numerical comparison between the tests is given in Section 5. More detailed proofs and further discussion may be found in an associated technical report [14].

1 L_1 -Based Statistic

Denote by ν_n , $\mu_{n,1}$ and $\mu_{n,2}$ the empirical measures associated with the samples $(X_1, Y_1), \dots, (X_n, Y_n)$, $X_1, \dots, X_n$, and $Y_1, \dots, Y_n$, respectively, so that

$$\begin{aligned}\nu_n(A \times B) &= n^{-1} \# \{i : (X_i, Y_i) \in A \times B, i = 1, \dots, n\}, \\ \mu_{n,1}(A) &= n^{-1} \# \{i : X_i \in A, i = 1, \dots, n\}, \quad \text{and} \\ \mu_{n,2}(B) &= n^{-1} \# \{i : Y_i \in B, i = 1, \dots, n\},\end{aligned}$$

for any Borel subsets A and B . Given the finite partitions $\mathcal{P}_n = \{A_{n,1}, \dots, A_{n,m_n}\}$ of $\mathbb{R}^d$ and $\mathcal{Q}_n = \{B_{n,1}, \dots, B_{n,m'_n}\}$ of $\mathbb{R}^{d'}$, we define the L_1 test statistic comparing ν_n and $\mu_{n,1} \times \mu_{n,2}$ as

$$L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) = \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} |\nu_n(A \times B) - \mu_{n,1}(A) \cdot \mu_{n,2}(B)|.$$

In the following two sections, we derive the large deviation and limit distribution properties of this L_1 statistic, and the associated independence tests.

1.1 Large Deviation Properties

For testing a simple hypothesis versus a composite alternative, Györfi and van der Meulen [2] introduced a related goodness of fit test statistic L_n defined as

$$L_n(\mu_n, \mu) = \sum_{A \in \mathcal{P}_n} |\mu_n(A) - \mu(A)|.$$

Beirlant [15], and Biau and Györfi [1] proved that, for all $0 < \varepsilon$,

$$\mathbf{P}\{L_n(\mu_n, \mu) > \varepsilon\} \leq 2^{m_n} e^{-n\varepsilon^2/2}. \quad (1)$$

We now show that a similar result follows quite straightforwardly for the L_1 independence statistic.

Theorem 1. *Under $\mathcal{H}_0$, for all $0 < \varepsilon_1$, $0 < \varepsilon_2$ and $0 < \varepsilon_3$,*

$$\mathbf{P}\{L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > \varepsilon_1 + \varepsilon_2 + \varepsilon_3\} \leq 2^{m_n \cdot m'_n} e^{-n\varepsilon_1^2/2} + 2^{m_n} e^{-n\varepsilon_2^2/2} + 2^{m'_n} e^{-n\varepsilon_3^2/2}.$$

Proof. We bound $L_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$ according to

$$\begin{aligned} L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) &\leq \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} |\nu_n(A \times B) - \nu(A \times B)| \\ &\quad + \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} |\nu(A \times B) - \mu_1(A) \cdot \mu_2(B)| \\ &\quad + \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} |\mu_1(A) \cdot \mu_2(B) - \mu_{n,1}(A) \cdot \mu_{n,2}(B)|. \end{aligned}$$

The central term in the sum is zero under the null hypothesis. The proof is then completed by further applications of the triangle inequality, then using (1) on the resulting terms, and applying a union bound. $\blacksquare$

Theorem 1 yields a strong consistent test of homogeneity, which rejects the null hypothesis if $L_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$ becomes large. The test is distribution-free, i.e., the probability distributions ν , μ_1 and μ_2 are completely arbitrary. The proof of the following corollary is similar to that employed for the homogeneity test by Biau and Györfi [1].

Corollary 1. *Consider the test which rejects $\mathcal{H}_0$ when*

$$L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > c_1 \left(\sqrt{\frac{m_n m'_n}{n}} + \sqrt{\frac{m_n}{n}} + \sqrt{\frac{m'_n}{n}} \right) \approx c_1 \sqrt{\frac{m_n m'_n}{n}},$$

where $c_1 > \sqrt{2 \ln 2} \approx 1.177$. Assume conditions

$$\lim_{n \rightarrow \infty} m_n m'_n / n = 0, \quad \lim_{n \rightarrow \infty} m_n / \ln n = \infty, \quad \lim_{n \rightarrow \infty} m'_n / \ln n = \infty, \quad (2)$$

are satisfied. Then under $\mathcal{H}_0$, the test makes a.s. no error after a random sample size. Moreover, if $\nu \neq \mu_1 \times \mu_2$, and for any sphere S centered at the origin,

$$\lim_{n \rightarrow \infty} \max_{A \in \mathcal{P}_n, A \cap S \neq \emptyset} \text{diam}(A) = 0 \quad \text{and} \quad \lim_{n \rightarrow \infty} \max_{B \in \mathcal{Q}_n, B \cap S \neq \emptyset} \text{diam}(B) = 0, \quad (3)$$

then after a random sample size the test makes a.s. no error.

1.2 Asymptotic Normality

Beirlant et al. [3] proved, under conditions

$$\lim_{n \rightarrow \infty} m_n = \infty, \quad \lim_{n \rightarrow \infty} m_n / n = 0, \quad \lim_{n \rightarrow \infty} \max_{j=1, \dots, m_n} \mu(A_{nj}) = 0, \quad (4)$$

that

$$\sqrt{n} (L_n(\mu_n, \mu) - \mathbf{E}\{L_n(\mu_n, \mu)\}) / \sigma \xrightarrow{\mathcal{D}} \mathcal{N}(0, 1),$$

where $\xrightarrow{\mathcal{D}}$ stands for the convergence in distribution and $\sigma^2 = 1 - 2/\pi$. We adapt this proof to the case of independence testing (see Appendix for details).

Theorem 2. *Assume that conditions (2) and*

$$\lim_{n \rightarrow \infty} \max_{A \in \mathcal{P}_n} \mu_1(A) = 0, \quad \lim_{n \rightarrow \infty} \max_{B \in \mathcal{Q}_n} \mu_2(B) = 0, \quad (5)$$

are satisfied. Then under $\mathcal{H}_0$, there exists a centering sequence $(C_n)_{n \geq 1}$ depending on ν such that

$$\sqrt{n}(L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) - C_n) / \sigma \xrightarrow{\mathcal{D}} \mathcal{N}(0, 1), \quad \text{where } \sigma^2 = 1 - 2/\pi.$$

Theorem 2 yields the asymptotic null distribution of a consistent independence test, which rejects the null hypothesis if $L_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$ becomes large. In contrast to Corollary 1, and because of condition (4), this new test is *not* distribution-free. In particular, the measures μ_1 and μ_2 have to be nonatomic. The corollary below follows from Theorem 2, replacing C_n with the upper bound

$$C_n \leq \sqrt{2m_n m'_n / (\pi n)}$$

(the original expression for C_n is provided in the Appendix, eq. (20)).

Corollary 2. *Let $\alpha \in (0, 1)$. Consider the test which rejects $\mathcal{H}_0$ when*

$$L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > c_2 \sqrt{m_n m'_n / n} + \sigma / \sqrt{n} \Phi^{-1}(1 - \alpha) \approx c_2 \sqrt{m_n m'_n / n},$$

where $\sigma^2 = 1 - 2/\pi$, $c_2 = \sqrt{2/\pi} \approx 0.798$, and Φ denotes the standard normal distribution function. Then, under the conditions of Theorem 2, the test has asymptotic significance level α . Moreover, under the additional conditions (3), the test is consistent.

2 Log-Likelihood Statistic

In the goodness-of-fit testing literature the *I-divergence* or *log-likelihood statistic*,

$$I_n(\mu_n, \mu) = 2 \sum_{j=1}^{m_n} \mu_n(A_{n,j}) \log [\mu_n(A_{n,j}) / \mu(A_{n,j})],$$

plays an important role. For testing independence, the corresponding log-likelihood test statistic is defined as

$$I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) = 2 \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \nu_n(A \times B) \log \frac{\nu_n(A \times B)}{\mu_{n,1}(A) \cdot \mu_{n,2}(B)}.$$

The large deviation and the limit distribution properties of $I_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$ can be derived from the properties of

$$I_n(\nu_n, \nu) = 2 \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \nu_n(A \times B) \log [\nu_n(A \times B) / \nu(A \times B)],$$

since under the null hypothesis it can easily be seen that

$$I_n(\nu_n, \nu) - I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) = I_n(\mu_{n,1}, \mu_1) + I_n(\mu_{n,2}, \mu_2) \geq 0.$$

For the large deviation bound, Kallenberg [16], and Quine and Robinson [17] proved that, for all $\epsilon > 0$,

$$\mathbf{P}\{I_n(\mu_n, \mu)/2 > \epsilon\} \leq \binom{n + m_n - 1}{m_n - 1} e^{-n\epsilon} \leq e^{m_n \log(n + m_n) - n\epsilon}.$$

Therefore under the condition $m_n \log n = o(n)$, which is stronger than (4),

$$\mathbf{P}\{I_n(\mu_n, \mu)/2 > \epsilon\} = e^{-n(\epsilon+o(1))}. \quad (6)$$

A test based on this result can be introduced which rejects independence if

$$I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) \geq m_n m'_n n^{-1} (2 \log(n + m_n m'_n) + 1).$$

Under $\mathcal{H}_0$, we obtain the non-asymptotic bound

$$\begin{aligned} & \mathbf{P}\{I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > m_n m'_n n^{-1} (2 \log(n + m_n m'_n) + 1)\} \\ & \leq \mathbf{P}\{I_n(\nu_n, \nu) > m_n m'_n n^{-1} (2 \log(n + m_n m'_n) + 1)\} \leq e^{-m_n m'_n}. \end{aligned}$$

Therefore condition (2) implies

$$\sum_{n=1}^{\infty} \mathbf{P}\{I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > m_n m'_n n^{-1} (2 \log(n + m_n m'_n) + 1)\} < \infty,$$

and by the Borel-Cantelli lemma we have strong consistency under the null hypothesis. Under the alternative hypothesis the proof of strong consistency follows from Pinsker's inequality,

$$L_n^2(\nu_n, \mu_{n,1} \times \mu_{n,2}) \leq I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}). \quad (7)$$

Concerning the limit distribution, Györfi and Vajda [4] proved under (4),

$$(nI_n(\mu_n, \mu) - m_n) (2m_n)^{-1/2} \xrightarrow{\mathcal{D}} \mathcal{N}(0, 1).$$

This implies that for any real valued x , under conditions (2) and (5),

$$\mathbf{P}\left\{\frac{nI_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) - m_n m'_n}{\sqrt{2m_n m'_n}} \leq x\right\} \leq \mathbf{P}\left\{\frac{nI_n(\nu_n, \nu) - m_n m'_n}{\sqrt{2m_n m'_n}} \leq x\right\} \rightarrow \Phi(x),$$

from which an asymptotically α -level test follows straightforwardly.

3 Pearson χ^2 Statistic

Another statistic for testing independence is the Pearson χ^2 test statistic,

$$\chi_n^2(\nu_n, \mu_{n,1} \times \mu_{n,2}) = \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \frac{(\nu_n(A \times B) - \mu_{n,1}(A) \cdot \mu_{n,2}(B))^2}{\mu_{n,1}(A) \cdot \mu_{n,2}(B)}.$$

For the associated goodness of fit test, Quine and Robinson [17] provide a large deviation bound for the statistic

$$\chi_n^2(\mu_n, \mu) = \sum_{j=1}^{m_n} \frac{(\mu_n(A_{n,j}) - \mu(A_{n,j}))^2}{\mu(A_{n,j})}, \quad (8)$$

from which it may be possible to obtain a strong consistent distribution-free test of independence. The asymptotic distribution of (8) under conditions (4) was addressed by Györfi and Vajda [4], who proved

$$(n\chi_n^2(\mu_n, \mu) - m_n)(2m_n)^{-1/2} \xrightarrow{\mathcal{D}} \mathcal{N}(0, 1).$$

We conjecture that under the conditions (2) and (5),

$$(n\chi_n^2(\nu_n, \mu_{n,1} \times \mu_{n,2}) - m_n m'_n)(2m_n m'_n)^{-1/2} \xrightarrow{\mathcal{D}} \mathcal{N}(0, 1),$$

from which an asymptotically α -level test follows.

4 Kernel-Based Statistic

We now present a second class of approaches to independence testing, based on a kernel statistic. We can derive this statistic in a number of ways. The most immediate interpretation, introduced by Rosenblatt [7], defines the statistic as the L_2 distance between the joint density estimate and the product of marginal density estimates. Let K and K' be density functions (called kernels) defined on $\mathbb{R}^d$ and on $\mathbb{R}^{d'}$, respectively. For the bandwidth $h > 0$, define

$$K_h(x) = \frac{1}{h^d} K\left(\frac{x}{h}\right) \quad \text{and} \quad K'_h(x) = \frac{1}{h^{d'}} K'\left(\frac{x}{h}\right).$$

The Rosenblatt-Parzen kernel density estimates of the density of (X, Y) and X are respectively

$$f_n(x, y) = \frac{1}{n} \sum_{i=1}^n K_h(x - X_i) K'_h(y - Y_i) \quad \text{and} \quad f_{n,1}(x) = \frac{1}{n} \sum_{i=1}^n K_h(x - X_i), \quad (9)$$

with $f_{n,2}(y)$ defined by analogy. Rosenblatt [7] introduced the kernel-based independence statistic

$$T_n = \int_{\mathbb{R}^d \times \mathbb{R}^{d'}} (f_n(x, y) - f_{n,1}(x) f_{n,2}(y))^2 dx dy. \quad (10)$$

Alternatively, defining

$$L_h(x) = \int_{\mathbb{R}^d} K_h(u) K_h(x - u) du = \frac{1}{h^d} \int_{\mathbb{R}^d} K(u) K(x - u) du$$

and $L'_h(x)$ by analogy, we may write the kernel test statistic

$$\begin{aligned} T_n &= \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n L_h(X_i - X_j) L'_h(Y_i - Y_j) \\ &\quad - \frac{2}{n^3} \sum_{i=1}^n \left(\sum_{j=1}^n L_h(X_i - X_j) \right) \left(\sum_{j=1}^n L'_h(Y_i - Y_j) \right) \\ &\quad + \left(\frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n L_h(X_i - X_j) \right) \left(\frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n L'_h(Y_i - Y_j) \right). \end{aligned} \quad (11)$$

Note that at independence, the expected value of the statistic is not zero, but

$$\mathbf{E}\{T_n\} = \frac{n-1}{n^2} (L_h(0) - \mathbf{E}\{L_h(X_1 - X_2)\}) (L'_h(0) - \mathbf{E}\{L'_h(Y_1 - Y_2)\}) \quad (12)$$

$$\leq n^{-1} L_h(0) L'_h(0) = (nh^d h^{d'})^{-1} \|K\|^2 \|K'\|^2. \quad (13)$$

A second interpretation of the above statistic is as a smoothed difference between the joint characteristic function and the product of the marginals [8]. The characteristic function and Rosenblatt-Parzen window statistics can be quite similar: in fact, for appropriate smoothing and kernel choices and fixed n , they may be identical [9]. For increasing n , the main differences between the approaches are that the kernel bandwidth h must decrease in the Rosenblatt test for consistency of the kernel density estimates, and the more restrictive conditions on the Rosenblatt-Parzen test statistic [7, conditions a.1-a.4].

A generalization of the statistic to include non-Euclidean domains is presented by Gretton et al. [10, 11]. The test statistic in (11) is then interpreted as a biased empirical estimate of the Hilbert-Schmidt norm of a cross-covariance operator between reproducing kernel Hilbert spaces (RKHS),² $\|C_{xy}\|_{\text{HS}}^2$. Clearly, when K_h and K'_h are continuous and square integrable densities, the induced kernels L_h and L'_h are continuous positive definite RKHS kernels. However, as long as L_h and L'_h are *characteristic kernels* (in the sense of Fukumizu et al. [18]; see also Sriperumbudur et al. [19]), then $\|C_{xy}\|_{\text{HS}}^2 = 0$ iff X and Y independent: these kernels need not be inner products of square integrable probability density functions. The Gaussian kernel is characteristic on $\mathbb{R}^d$ [18], and universal kernels (in the sense of Steinwart [20]) are characteristic on compact domains [10]. Note that universal kernels exist that may not be written as inner products of kernel density functions: see examples in [20, Section 3]. An appropriate choice of kernels allows testing of dependence in general settings, such as distributions on strings and graphs [11].

4.1 Large Deviation Property

The empirical statistic T_n was previously shown by Gretton et al. [10] to converge in probability to its expectation with rate $1/\sqrt{n}$. We now provide a more refined bound, which is tighter when the bandwidth h decreases. We will obtain our results for the statistic

$$\tilde{T}_n = \|f_n(\cdot, \cdot) - \mathbf{E}f_n(\cdot, \cdot)\|^2,$$

since under the null hypothesis,

$$\begin{aligned} \sqrt{T_n} &= \|f_n(\cdot, \cdot) - f_{n,1}(\cdot)f_{n,2}(\cdot)\| \\ &\leq \sqrt{\tilde{T}_n} + \|f_{n,1}(\cdot)\| \|f_{n,2}(\cdot) - \mathbf{E}f_{n,2}(\cdot)\| + \|f_{n,1}(\cdot) - \mathbf{E}f_{n,1}(\cdot)\| \|\mathbf{E}f_{n,2}(\cdot)\| \approx \sqrt{\tilde{T}_n}. \end{aligned} \quad (14)$$

² Given RKHSs $\mathcal{F}$ and $\mathcal{G}$, the cross-covariance operator $C_{xy} : \mathcal{G} \rightarrow \mathcal{F}$ for the measure ν is defined such that for all $f \in \mathcal{F}$ and $g \in \mathcal{G}$, $\langle f, C_{xy}g \rangle_{\mathcal{F}} = \mathbf{E}\{[f(x) - \mathbf{E}\{f(x)\}][g(y) - \mathbf{E}\{g(y)\}]\}$.

Theorem 3. For any $\epsilon > 0$,

$$\mathbf{P} \left\{ \tilde{T}_n \geq \left(\epsilon + \mathbf{E} \left\{ \sqrt{\tilde{T}_n} \right\} \right)^2 \right\} \leq e^{-n\epsilon^2 / (2L_h(0)L'_h(0))}.$$

Proof. We apply McDiarmid's inequality [21]: Let $Z_1, \dots, Z_n$ be independent random variables taking values in a set A , and assume that $f : A^n \rightarrow \mathbb{R}$ satisfies

$$\sup_{\substack{z_1, \dots, z_n, \\ z'_i \in A}} |f(z_1, \dots, z_n) - f(z_1, \dots, z_{i-1}, z'_i, z_{i+1}, \dots, z_n)| \leq c_i, \quad 1 \leq i \leq n.$$

Then, for all $\epsilon > 0$,

$$\mathbf{P} \{ f(Z_1, \dots, Z_n) - \mathbf{E}f(Z_1, \dots, Z_n) \geq \epsilon \} \leq e^{-2\epsilon^2 / \sum_{i=1}^n c_i^2}.$$

Given the function $f = \sqrt{\tilde{T}_n}$, the result follows from

$$\frac{2}{n} \|K_h(\cdot - X_1)K'_h(\cdot - Y_1)\| = \frac{2}{n} \sqrt{L_h(0)L'_h(0)} =: c_i = c_1. \quad \blacksquare$$

From these inequalities we can derive a test of independence. Given ϵ such that $n\epsilon^2 / (2L_h(0)L'_h(0)) = 2 \ln n$, and recalling (13) and (14), a strongly consistent test rejects independence if

$$T_n > \|K\|^2 \|K'\|^2 (\sqrt{4 \ln n} + 1)^2 (nh^d h^{d'})^{-1}.$$

Under the alternative hypothesis, there are two cases. If $h \rightarrow 0$ and the density f exists and is square integrable, then $T_n \rightarrow \|f - f_1 f_2\|^2 > 0$ a.s. If h is fixed, the strong law of large numbers implies $T_n \rightarrow \|C_{xy}\|_{\text{HS}}^2 > 0$ for characteristic kernels, and the test is strongly consistent. In both cases the strong consistency is not distribution-free.

4.2 Limit Distribution

We now describe the asymptotic limit distribution of the test statistic T_n in (11). We address two cases: first, when the kernel bandwidth decreases, and second, when it remains fixed.

Let us consider the case where $K_h(x)$ and $K'_h(y)$ are intended to be used in a Rosenblatt-Parzen density estimator, as in (9). The corresponding density estimates in T_n are mean square consistent if $h = h_n$ such that

$$h_n \rightarrow 0 \quad \text{and} \quad nh_n^d h_n^{d'} \rightarrow \infty. \quad (15)$$

Based on the results in [22, 23, 24], we expect T_n to be asymptotically normally distributed. For an independence test, we require $\text{var}(T_n) \approx \text{var}(\tilde{T}_n)$. If $h \rightarrow 0$,

$$\text{var}(\tilde{T}_n) \approx 2\|f\|^2 n^{-2} h^{-d} h^{-d'}. \quad (16)$$

Therefore a possible form for the asymptotic normal distribution is

$$nh^{d/2}h^{d'/2}(T_n - \mathbf{E}\{T_n\})/\sigma \xrightarrow{\mathcal{D}} \mathcal{N}(0, 1),$$

where $\sigma^2 = 2\|f\|^2$. While an α -level test may be obtained by replacing $\mathbf{E}\{T_n\}$ with its upper bound in (13), the resulting threshold is still not distribution-free, since σ depends on the unknown f . The simplest distribution-free bound for the variance, $\sigma^2 \leq \|K\|^4 \|K'\|^4 n^{-2} h^{-2d} h^{-2d'}$, is unsatisfactory since its performance as a function of h is worse than the result in (16). An improved distribution-free bound on the variance is a topic for future research: we give an empirical estimate below (eq. 18) for use in asymptotic hypothesis tests.

We now consider the case of fixed h . Following [8], the distribution of T_n under $\mathcal{H}_0$ is

$$nT_n \xrightarrow{D} \sum_{l=1}^{\infty} \lambda_l z_l^2, \quad (17)$$

where $z_l \sim \mathcal{N}(0, 1)$ i.i.d., and λ_l are the solutions to an eigenvalue problem depending on the unknown distribution of X and Y (see [11, Theorem 2]). A difficulty in using the statistic (11) in a hypothesis test therefore arises due to the form of the null distribution, which is a function of the unknown distribution over X and Y , whether or not h is fixed. In the case of h decreasing according to (15), we may use an empirical estimate of the variance of T_n under $\mathcal{H}_0$ due to Gretton et al. [11, Theorem 4]. Denoting by $\odot$ the entrywise matrix product and $A^{\cdot 2}$ the entrywise matrix power,

$$\text{var}(T_n) = \mathbf{1}^\top (\mathbf{B} - \text{diag}(\mathbf{B})) \mathbf{1}, \quad \text{where} \quad \mathbf{B} = ((\mathbf{H}\mathbf{L}\mathbf{H}) \odot (\mathbf{H}\mathbf{L}'\mathbf{H}))^{\cdot 2}, \quad (18)$$

$\mathbf{L}$ is a matrix with entries $L_h(X_i - X_j)$, $\mathbf{L}'$ is a matrix with entries $L'_h(Y_i - Y_j)$, $\mathbf{H} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}^\top$ is a centering matrix, and $\mathbf{1}$ an $n \times 1$ vector of ones.

Two approaches have been proposed in the case of fixed h to obtain quantiles of the null distribution (17) for hypothesis testing: repeated shuffling of the sample [8], and approximation by a two-parameter Gamma density [9],

$$nT_n \sim x^{\alpha-1} e^{-x/\beta} / (\beta^\alpha \Gamma(\alpha)), \quad \alpha = (\mathbf{E}\{T_n\})^2 / \text{var}(T_n), \quad \beta = n \text{var}(T_n) / \mathbf{E}\{T_n\},$$

using $\mathbf{E}\{T_n\}$ from (12). This Gamma approximation was found by [11] to perform identically on the Section 5 benchmark data to the more computationally expensive approach of Feuerverger [8]. We emphasize, however, that this approximation is a heuristic, with no guarantees on asymptotic performance.

We end this section with an empirical comparison between the Normal and two-parameter Gamma null distribution approximations, and the null CDF generated by repeated independent samples of T_n . We chose X and Y to be independent and univariate, with X having a uniform distribution and Y being a symmetric bimodal mixture of Gaussians. Both variables had zero mean and unit standard deviation. Results are plotted in Figure 1. We observe that as the kernel size increases, the Gamma approximation of T_n becomes more accurate (although it is always good for large quantiles, which is the region most important to a hypothesis test). The Normal approximation is close to the Gamma

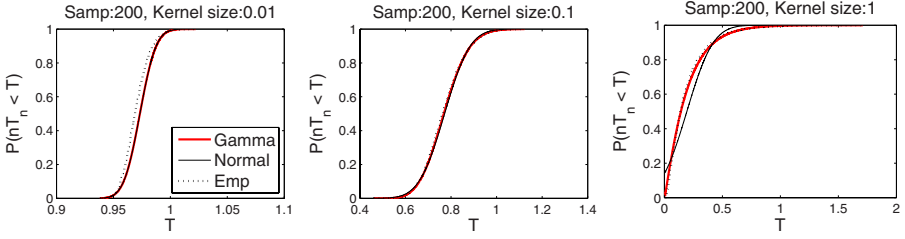


Fig. 1. Simulated cumulative distribution function of T_n (*Emp*) under $\mathcal{H}_0$ for $n = 200$, compared with the two-parameter Gamma distribution (*Gamma*) and the Normal distribution (*Normal*). The empirical CDF was obtained using 5000 independent draws of T_n .

approximation for small kernel sizes, but is less accurate for larger kernel sizes (where “small” and “large” will depend on the measure ν).

5 Experiments

In comparing the independence tests, we made use of the multidimensional benchmark data proposed by Gretton et al. [11]. We tested the independence of both one-dimensional and two-dimensional random variables (i.e. $d = d' = 1$ and $d = d' = 2$). The data were constructed as follows. First, we generated n samples of two univariate random variables, each drawn at random from the ICA benchmark densities in Figure 5 of Bach and Jordan [25]: these included super-Gaussian, sub-Gaussian, multimodal, and unimodal distributions. Second, we mixed these random variables using a rotation matrix parametrised by an angle θ , varying from 0 to $\pi/4$ (a zero angle meant the data were independent, while dependence became easier to detect as the angle increased to $\pi/4$: see the two plots in Figure 2). Third, in the case of $d = 2$, a second dimension was appended to each of the mixed variables, consisting of independent Gaussian noise of zero mean and unit standard deviation; and each resulting vector was multiplied by an independent random two-dimensional orthogonal matrix, to obtain vectors dependent across all observed dimensions. We emphasise that classical approaches (such as Spearman’s ρ or Kendall’s τ) are unable to find this dependence, since the variables are uncorrelated; nor can we recover the subspace in which the variables are dependent using PCA, since this subspace has the same second order properties as the noise. We investigated sample sizes $n = 128, 512$.

We compared three different asymptotic independence testing approaches based on space partitioning: the L_1 test, denoted *L1*; the Pearson χ^2 test *Pears*; and the log likelihood test *Like*. The number of discretisations per dimension was set at $m_n = m'_n = 4$, besides in the $n = 128, d = 2$ case, where it was set at $m_n = m'_n = 3$: in the latter case, there were too few samples per bin when a greater number of partitions were used. We divided our spaces $\mathbb{R}^d$ and $\mathbb{R}^{d'}$ into roughly equiprobable bins. Further increases in the number of partitions per dimension, where sufficient samples were present to justify this (i.e.,

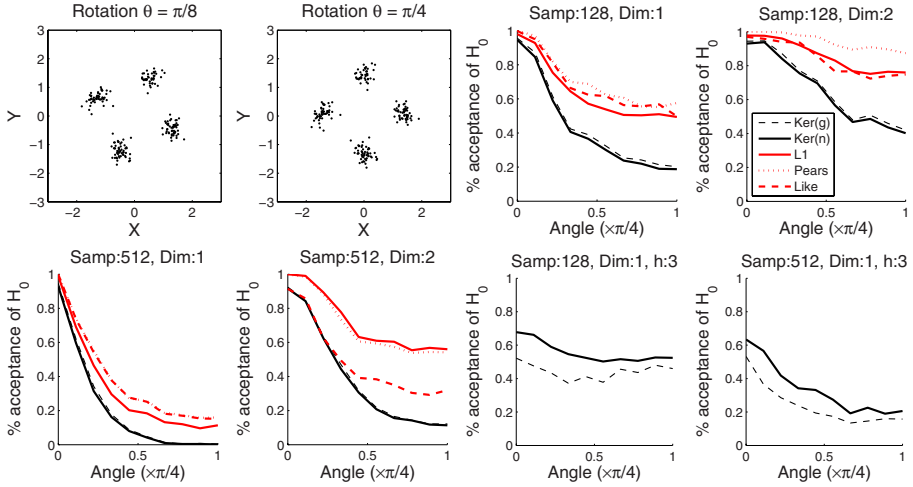


Fig. 2. Top left plots: Example dataset for $d = d' = 1$, $n = 200$, and rotation angles $\theta = \pi/8$ (left) and $\theta = \pi/4$ (right). In this case, both distributions prior to rotation are mixtures of two Gaussians. **Next four plots:** Rate of acceptance of $\mathcal{H}_0$ for the *PD*, *fCorr*, *HSICp*, and *HSICg* tests. “Samp” is the number n of samples, and “dim” is the dimension $d = d'$ of x and y . **Bottom right plots** Performance of the *Ker(g)* and *Ker(n)* tests for a large kernel size $h = 3$, and $\alpha = 0.5$, to show the difference between the Normal and two-parameter Gamma approximations to the null distribution.

the $n = 512, d = 1$ case), resulted only in very minor shifts in performance. We also compared with the kernel approach from Section 4, using both the Gamma *Ker(g)* and Normal *Ker(n)* approximations to the null distribution. Our kernels were Gaussian for both X and Y , with h and h' set to the median distances between samples of the respective variables, following Gretton et al. [11].

Results are plotted in Figure 2 (average over 500 independent generations of the data). The y -intercept on these plots corresponds to the acceptance rate of $\mathcal{H}_0$ at independence, or $1 - (\text{Type I error})$, and should be close to the design parameter of $1 - \alpha = 0.95$. Elsewhere, the plots indicate acceptance of $\mathcal{H}_0$ where the underlying variables are dependent, i.e. the Type II error. As expected, dependence becomes easier to detect as θ increases from 0 to $\pi/4$, when n increases, and when d decreases. Although no tests are reliable for small θ , several tests do well as θ approaches $\pi/4$ (besides the case of $n = 128, d = 2$). For smaller numbers of samples ($n = 128$), the L_1 test performs the same as or slightly better than the log likelihood test; the Pearson χ^2 test always performs worst. For larger numbers of samples ($n = 512$), the L_1 test has a slight advantage at $d = 1$, but the log-likelihood test displays far better performance for $d = 2$. The superior performance of the log-likelihood test compared with the χ^2 test might arise due to the different convergence properties of the two test statistics. In particular, we note the superior convergence behaviour of the goodness-of-fit statistic for the log likelihood, as compared with the χ^2 statistic, in terms of

the dependence of the latter on the number m_n of partitions used [15]. In all cases, the kernel-based test outperforms the remaining methods, and behaviour under the Normal and Gamma null distribution models is virtually identical. That said, we should bear in mind the kernel test thresholds require $\mathbf{E}\{T_n\}$ and $\text{var}(T_n)$, which are unknown and must be estimated from the data: thus, unlike the L_1 , χ^2 , and log likelihood tests, the kernel test thresholds are not distribution-independent.

It is of interest to further investigate the null distribution approximation strategies for the kernel tests. We used an artificially high kernel bandwidth $h = 3$, and a lower $\alpha = 0.5$, to make visible the performance difference. Results are shown in the final row of Figure 2. In accordance with Figure 1, the Gaussian approximation yields a larger threshold than the true CDF would require, and consequently has a Type I error below the design level α .

Acknowledgments. The research of László Györfi is supported by the Hungarian Academy of Sciences (MTA SZTAKI). This work is also supported by the IST Program of the EC, under the FP7 Network of Excellence, ICT-216886-NOE.

References

- [1] Biau, G., Györfi, L.: On the asymptotic properties of a nonparametric l_1 -test statistic of homogeneity. *IEEE Trans. Inform. Theory* 51, 3965–3973 (2005)
- [2] Györfi, L., van der Meulen, E.C.: A consistent goodness of fit test based on the total variation distance. In: Roussas, G. (ed.) *Nonparametric Functional Estimation and Related Topics*, pp. 631–645. Kluwer Academic Publishers, Dordrecht (1990)
- [3] Beirlant, J., Györfi, L., Lugosi, G.: On the asymptotic normality of the l_1 - and l_2 -errors in histogram density estimation. *Canad. J. Statist.* 22, 309–318 (1994)
- [4] Györfi, L., Vajda, I.: Asymptotic distributions for goodness of fit statistics in a sequence of multinomial models. *Stat. Prob. Lett.* 56, 57–67 (2002)
- [5] Dembo, A., Peres, Y.: A topological criterion for hypothesis testing. *Ann. Statist.* 22, 106–117 (1994)
- [6] Read, T., Cressie, N.: *Goodness-Of-Fit Statistics for Discrete Multivariate Analysis*. Springer, New York (1988)
- [7] Rosenblatt, M.: A quadratic measure of deviation of two-dimensional density estimates and a test of independence. *The Annals of Statistics* 3, 1–14 (1975)
- [8] Feuerverger, A.: A consistent test for bivariate dependence. *International Statistical Review* 61, 419–433 (1993)
- [9] Kankainen, A.: *Consistent Testing of Total Independence Based on the Empirical Characteristic Function*. PhD thesis, University of Jyväskylä (1995)
- [10] Gretton, A., Bousquet, O., Smola, A., Schölkopf, B.: Measuring statistical dependence with Hilbert-Schmidt norms. In: Jain, S., Simon, H.U., Tomita, E. (eds.) *ALT 2005. LNCS (LNAI)*, vol. 3734, pp. 63–78. Springer, Heidelberg (2005)
- [11] Gretton, A., Fukumizu, K., Teo, C.H., Song, L., Schölkopf, B., Smola, A.: A kernel statistical test of independence. In: *NIPS 20* (2008)
- [12] Hoeffding, W.: A nonparametric test for independence. *The Annals of Mathematical Statistics* 19, 546–557 (1948)
- [13] Blum, J.R., Kiefer, J., Rosenblatt, M.: Distribution free tests of independence based on the sample distribution function. *Ann. Math. Stat.* 32, 485–498 (1961)

- [14] Gretton, A., Györfi, L.: Consistent nonparametric tests of independence. Technical Report 172, MPI for Biological Cybernetics (2008)
- [15] Beirlant, J., Devroye, L., Györfi, L., Vajda, I.: Large deviations of divergence measures on partitions. *J. Statist. Plan. Inference* 93, 1–16 (2001)
- [16] Kallenberg, W.C.M.: On moderate and large deviations in multinomial distributions. *Annals of Statistics* 13, 1554–1580 (1985)
- [17] Quine, M., Robinson, J.: Efficiencies of chi-square and likelihood ratio goodness-of-fit tests. *Ann. Statist.* 13, 727–742 (1985)
- [18] Fukumizu, K., Gretton, A., Sun, X., Schölkopf, B.: Kernel measures of conditional dependence. In: *NIPS* 20 (2008)
- [19] Sriperumbudur, B.K., Gretton, A., Fukumizu, K., Lanckriet, G.R.G., Schölkopf, B.: Injective Hilbert space embeddings of probability measures. In: *COLT*, pp. 111–122 (2008)
- [20] Steinwart, I.: On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research* 2, 67–93 (2001)
- [21] McDiarmid, C.: On the method of bounded differences. In: *Survey in Combinatorics*, pp. 148–188. Cambridge University Press, Cambridge (1989)
- [22] Hall, P.: Central limit theorem for integrated square error of multivariate non-parametric density estimators. *Journal of Multivariate Analysis* 14, 1–16 (1984)
- [23] Cotterill, D.S., Csörgő, M.: On the limiting distribution of and critical values for the Hoeffding, Blum, Kiefer, Rosenblatt independence criterion. *Statistics and Decisions* 3, 1–48 (1985)
- [24] Beirlant, J., Mason, D.M.: On the asymptotic normality of l_p -norms of empirical functionals. *Math. Methods Statist.* 4, 1–19 (1995)
- [25] Bach, F.R., Jordan, M.I.: Kernel independent component analysis. *J. Mach. Learn. Res.* 3, 1–48 (2002)

A Proof of Theorem 2

The main difficulty in proving Theorem 2 is that it states the asymptotic normality of $L_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$, which is a sum of dependent random variables. To overcome this problem, we use a “Poissonization” argument originating from the fact that an empirical process is equal in distribution to the conditional distribution of a Poisson process given the sample size (see [3] for details). We begin by introducing the necessary terminology. For each $n \geq 1$, denote by N_n a $\text{Poisson}(n)$ random variable, defined on the same probability space as the sequences $(X_i)_{i \geq 1}$ and $(Y_i)_{i \geq 1}$, and independent of these sequences. Further define ν_{N_n} , $\mu_{N_n,1}$ and $\mu_{N_n,2}$ as the Poissonized version of the empirical measures associated with the samples $\{(X_i, Y_i)\}$, $\{X_i\}$ and $\{Y_i\}$, respectively,

$$\begin{aligned} n\nu_{N_n}(A \times B) &= \#\{i : (X_i, Y_i) \in A \times B, i = 1, \dots, N_n\}, \\ n\mu_{N_n,1}(A) &= \#\{i : X_i \in A, i = 1, \dots, N_n\}, \quad \text{and} \\ n\mu_{N_n,2}(B) &= \#\{i : Y_i \in B, i = 1, \dots, N_n\}, \end{aligned}$$

for any Borel subsets A and B . Clearly, $n\nu_{N_n}(A \times B)$, $n\mu_{N_n,1}(A)$, and $n\mu_{N_n,2}(B)$ are Poisson random variables. The Poissonized version $\tilde{L}_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$ of $L_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$ is then

$$\tilde{L}_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) = \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} |\nu_{N_n}(A \times B) - \mu_{N_n,1}(A) \cdot \mu_{N_n,2}(B)|.$$

Key to the proof of Theorem 2 is the following result, which extends the proposition of [3, p. 311].

Proposition 1. *Let g_{njk} ($n \geq 1$, $j = 1, \dots, m_n$, $k = 1, \dots, m'_n$) be real measurable functions, and let*

$$M_n := \sum_{j=1}^{m_n} \sum_{k=1}^{m'_n} g_{njk} (\nu_{N_n}(A_{nj} \times B_{nk}) - \mu_{N_n,1}(A_{nj})\mu_{N_n,2}(B_{nk})).$$

Assume that under the null hypothesis,

$$\mathbf{E}\{g_{njk} (\nu_{N_n}(A_{nj} \times B_{nk}) - \mu_{N_n,1}(A_{nj})\mu_{N_n,2}(B_{nk}))\} = 0,$$

and that

$$\left(M_n, \frac{N_n - n}{\sqrt{n}}\right) \xrightarrow{\mathcal{D}} \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma^2 & 0 \\ 0 & 1 \end{bmatrix}\right) \quad (19)$$

as $n \rightarrow \infty$, where σ is a positive constant and $\mathcal{N}(\mathbf{m}, \mathbf{C})$ is a normally distributed random variable with mean $\mathbf{m}$ and covariance matrix $\mathbf{C}$. Then

$$\frac{1}{\sigma} \sum_{j=1}^{m_n} \sum_{k=1}^{m'_n} g_{njk} (\nu_n(A_{nj} \times B_{nk}) - \mu_{n,1}(A_{nj})\mu_{n,2}(B_{nk})) \xrightarrow{\mathcal{D}} \mathcal{N}(0, 1).$$

The proof of the proposition is a simple extension of that by Beirlant et al. for the goodness-of-fit case [3, pp. 311–313]. We now turn to the proof of Theorem 2.

Proof (Theorem 2, sketch only). We will show the theorem with the centering constant

$$C_n = \mathbf{E}\{\tilde{L}_n(\nu_n, \mu_{n,1} \times \mu_{n,2})\} = \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \mathbf{E}\{|\nu_{N_n}(A \times B) - \mu_{N_n,1}(A)\mu_{N_n,2}(B)|\}. \quad (20)$$

Define

$$g_{njk}(x) := \sqrt{n} (|x| - \mathbf{E} |\nu_{N_n}(A_{nj} \times B_{nk}) - \mu_{N_n,1}(A_{nj})\mu_{N_n,2}(B_{nk})|).$$

Our goal is to prove that the assumption in (19) holds. In particular (see [3, 1] for details), we require a central limit result to hold for the Poissonized statistic

$$S_n := t\sqrt{n} \sum_{j=1}^{m_n} \sum_{k=1}^{m'_n} \left(|\nu_{N_n}(A_{nj} \times B_{nk}) - \mu_{N_n,1}(A_{nj})\mu_{N_n,2}(B_{nk})| - \mathbf{E} |\nu_{N_n}(A_{nj} \times B_{nk}) - \mu_{N_n,1}(A_{nj})\mu_{N_n,2}(B_{nk})| \right) + v\sqrt{n} (N_n/n - 1).$$

Once we obtain $\text{var}(S_n)$, the asymptotic normality in (19) can be proved by verifying the Lyapunov conditions as in Beirlant et al. [3]. We have that

$$N_n/n - 1 = \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \nu_{N_n}(A \times B) - \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \mu_1(A)\mu_2(B),$$

and therefore the variance of S_n is

$$\begin{aligned} \text{var}(S_n) = & t^2 n \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \text{var} |\nu_{N_n}(A \times B) - \mu_{N_n,1}(A) \mu_{N_n,2}(B)| \\ & + 2tvn \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \mathbf{E} \{ |\nu_{N_n}(A \times B) - \mu_{N_n,1}(A) \mu_{N_n,2}(B)| \\ & \cdot (\nu_{N_n}(A \times B) - \mu_1(A) \mu_2(B)) \} + v^2. \end{aligned}$$

One can check that there exist standard normal random variables $Z_{A \times B}$, Z_A , and Z_B such that

$$\begin{aligned} \nu_{N_n}(A \times B) &\stackrel{\mathcal{D}}{\approx} Z_{A \times B} \sqrt{\mu_1(A) \mu_2(B)/n} + \mu_1(A) \mu_2(B), \\ \mu_{N_n,1}(A) &\stackrel{\mathcal{D}}{\approx} Z_A \sqrt{\mu_1(A)/n} + \mu_1(A), \end{aligned}$$

with $\mu_{N_n,2}(B)$ and Z_B defined by analogy. Making these substitutions and simplifying,

$$\text{var}(S_n) \approx t^2(1 - 2/\pi) + v^2. \quad \blacksquare$$

Supermartingales in Prediction with Expert Advice

Alexey Chernov, Yuri Kalnishkan, Fedor Zhdanov, and Vladimir Vovk

Computer Learning Research Centre, Department of Computer Science
Royal Holloway, University of London, Egham, Surrey TW20 0EX, UK
`{chernov,yura,fedor,vovk}@cs.rhul.ac.uk`

Abstract. This paper compares two methods of prediction with expert advice, the Aggregating Algorithm and the Defensive Forecasting, in two different settings. The first setting is traditional, with a countable number of experts and a finite number of outcomes. Surprisingly, these two methods of fundamentally different origin lead to identical procedures. In the second setting the experts can give advice conditional on the learner's future decision. Both methods can be used in the new setting and give the same performance guarantees as in the traditional setting. However, whereas defensive forecasting can be applied directly, the AA requires substantial modifications.

1 Introduction

The framework of prediction with expert advice was introduced in the late 1980s. In contrast to statistical learning theory, the methods of prediction with expert advice work without making any statistical assumption about the source of data. The role of the assumptions is played by a “pool of experts” that the predictor competes with. For references and details, see the monograph [3].

Many methods for prediction with expert advice are known. This paper deals with two of them: the Aggregating Algorithm [15] and defensive forecasting [17]. The Aggregating Algorithm (the AA for short) is a member of the family of exponential-weights algorithms, and so implements a Bayesian-type aggregation; various optimality properties of the AA have been established [16]. Defensive forecasting is a recently developed technique that combines the ideas of game-theoretic probability [12] with Levin and Gács's ideas of neutral measure [7, 9] and Foster and Vohra's ideas of universal calibration [5].

The idea of defensive forecasting is that a prediction strategy is developed assuming that we are given probability forecasts satisfying a convenient law of probability. In game-theoretic probability, a law of probability is represented as a strategy for an imaginary opponent, Sceptic, whose capital tends to infinity (or becomes large) if the law is violated. Sceptic's capital is a supermartingale, and a well-known result (Lemma 3 of this paper) says that there is a forecasting strategy that prevents Sceptic's capital from growing, thereby forcing the law of probability. This paper gives a self-contained description of a simple version

of the method of defensive forecasting; we have no need to talk about laws of probability (for us, they are synonymous with game-theoretic supermartingales).

The older versions of defensive forecasting (see, e.g., [17]) minimize Learner’s actual loss with the help of the following trick: a probability forecasting system is constructed such that the actual losses (Learner’s and experts’) are close to the (one-step-ahead conditional) expected losses; at each step Learner minimizes the expected loss. (Therefore, the law of probability used is the conjunction of several laws of large numbers.) Defensive forecasting, as well as the AA, can be used for competitive online prediction against “pools of experts” consisting of all functions from a large functional class (see [18, 19]). However, the loss bounds proved so far are generally incomparable: for large classes (such as many Sobolev spaces), defensive forecasting is better, whereas for smaller classes (such as classes of analytical functions), the AA works better. Note that the optimality results for the AA are obtained for the case of free agents as experts, not functions from a given class, thus we need to evaluate the algorithms anew.

In this paper, the AA and defensive forecasting are compared in the simple case of a finite number of outcomes. Learner competes with a pool of experts Θ . Learner and the experts suffer some loss at each step. We are interested in the performance guarantees of the form

$$\forall \theta \in \Theta \quad L_N \leq cL_N(\theta) + a(\theta),$$

where L_N is the cumulative loss of Learner and $L_N(\theta)$ is the cumulative loss of expert θ over the first N steps, c is some constant and a depends on θ only. In this case, we prove the following fact: an exponent of the regret ($L_N - cL_N(\theta)$) is a supermartingale if and only if the AA guarantees for this c the bound above. A defensive forecasting algorithm exploiting this fact turns out to give the same predictions as the AA.

Then we consider a new setting for prediction with expert advice, where the experts are allowed to “second-guess”, that is, to give “conditional” predictions that are functions of the future Learner’s decision (cf. the notion of internal regret [6]). If the dependence is regular enough (the expert’s loss is continuous in Learner’s loss), the Defensive Forecasting algorithm works in the new setting virtually without changes and guarantees the same performance bound as in the traditional setting. The AA in its original form cannot work in the new setting, and we suggest a modified version of the AA for this case.

2 Aggregating Algorithm

We begin with formulating the setting of the prediction with expert advice and the AA. Then we give a proof of the standard performance bound for the AA, which provides an insight to the “supermartingale” nature of the bound.

A *game of prediction* consists of three components: a non-empty finite set Ω of possible outcomes, a non-empty set Γ of possible decisions, and a function $\lambda : \Omega \times \Gamma \rightarrow [0, \infty]$ called the *loss function*. For technical convenience, we identify each decision $\gamma \in \Gamma$ with the function $\omega \mapsto \lambda(\omega, \gamma)$ (and also with

a point in an $|\Omega|$ -dimensional Euclidian space with pointwise operations). Let $\Lambda = \{g \in [0, \infty]^\Omega \mid \exists \gamma \in \Gamma \forall \omega \in \Omega g(\omega) = \lambda(\omega, \gamma)\}$ be the set of *predictions*. From now on, a game is a pair (Ω, Λ) , where $\Lambda \subseteq [0, \infty]^\Omega$.

The game of prediction with expert advice is played by Learner, Reality, and Experts; the set (“pool”) of Experts is denoted by Θ . We will assume that Θ is (finite or) countable. In this paper, there is no loss of generality in assuming that Reality and all Experts are cooperative, since we are only interested in what can be achieved by Learner alone; therefore, we essentially consider a two-player game. The protocol of the game is the following.

PREDICTION WITH EXPERT ADVICE

$L_0 := 0$.

$L_0^\theta := 0, \theta \in \Theta$.

FOR $n = 1, 2, \dots$:

Experts $\theta \in \Theta$ announce $\gamma_n^\theta \in \Lambda$.

Learner announces $\gamma_n \in \Lambda$.

Reality announces $\omega_n \in \Omega$.

$L_n := L_{n-1} + \gamma_n(\omega_n)$.

$L_n^\theta := L_{n-1}^\theta + \gamma_n^\theta(\omega_n)$.

END FOR.

The goal of Learner is to keep L_n less or at least not much greater than L_n^θ , at each step n and for all $\theta \in \Theta$.

To analyse the game, we need some additional notation. A point $g \in [0, \infty]^\Omega$ is called a *superprediction* if there is $\gamma \in \Lambda$ such that $\gamma(\omega) \leq g(\omega)$ for all $\omega \in \Omega$. The last property will be denoted by $\gamma \leq g$. Let Σ_Λ be the set of all superpredictions.

The *Aggregating Algorithm* is a strategy for Learner. It has four parameters: reals $c \geq 1$ and $\eta > 0$, a *distribution* P_0 on Θ (that is, $P_0(\theta) \in [0, 1]$ for all $\theta \in \Theta$ and $\sum_{\theta \in \Theta} P_0(\theta) = 1$), and a substitution function $\sigma : \Sigma_\Lambda \rightarrow \Lambda$ such that $\sigma(g) \leq g$ for any $g \in \Sigma_\Lambda$.

At step N , the AA takes a point $g_N \in [0, \infty]^\Omega$ defined by the formula

$$g_N(\omega) = -\frac{c}{\eta} \ln \sum_{\theta \in \Theta} \frac{P_{N-1}(\theta)}{\sum_{\theta \in \Theta} P_{N-1}(\theta)} \exp(-\eta \gamma_N^\theta(\omega)),$$

where

$$P_{N-1}(\theta) = P_0(\theta) \prod_{n=1}^{N-1} \exp(-\eta \gamma_n^\theta(\omega_n))$$

is an unnormalized distribution on Θ . Then, $\gamma_N = \sigma(g_N)$ is announced as the next prediction of Learner.

The step of AA is correct if and only if g_N is a superprediction. The necessary and sufficient condition for this is

$$\exists \gamma_N \in \Lambda \forall \omega \quad \gamma_N(\omega_N) \leq -\frac{c}{\eta} \ln \sum_{\theta \in \Theta} \frac{P_{N-1}(\theta)}{\sum_{\theta \in \Theta} P_{N-1}(\theta)} \exp(-\eta \gamma_N^\theta(\omega)). \quad (1)$$

We say that the AA is *realizable* for certain c and η if the condition (1) is true regardless of $\gamma_N^\theta \in \Lambda$ and P_{N-1} (that is, regardless of P_0 , the history, and the opponents' moves). In other words, for any finite set $G \subseteq \Lambda$ and for any distribution ρ on G , it holds that

$$\exists \gamma \in \Lambda \forall \omega \quad \exp\left(-\frac{\eta}{c}\gamma(\omega)\right) \geq \sum_{g \in G} \rho(g) \exp(-\eta g(\omega)). \quad (2)$$

A detailed survey of the AA, its properties, attainable bounds and respective conditions on c and η for different games can be found in [16]. Remark only that if the AA is realizable for $c = 1$ and some η , the game is called η -mixable (and mixable if it is η -mixable for some η), and this case is of special interest.

Theorem 1 (Vovk, 1990). *If the AA is realizable for c and η , then the AA with parameters c , η , P_0 , and σ guarantees that at each step n for all experts θ*

$$L_n \leq cL_n^\theta + \frac{c}{\eta} \ln \frac{1}{P_0(\theta)}.$$

The theorem was proved in [15]. Here we reproduce the proof emphasizing the points we need in the sequel.

Proof. We need to deduce the performance bound from the condition (1). To this end, we will rewrite (1) and get a semi-invariant of AA—a value that does not grow.

First, note that if we replace $\exists \gamma_N \in \Lambda$ by $\exists \gamma_N \in \Sigma_\Lambda$ in (1), we get an equivalent statement. Indeed, $\Lambda \subseteq \Sigma_\Lambda$, thus one direction is trivial. The other direction holds by definition of a superprediction.

Second, note P_{N-1} occur in (1) only as a ratio of $P_{N-1}(\theta)$ to their sum, so we can multiply all $P_{N-1}(\theta)$ by a constant (an expression without θ). Let us define Q_{N-1} by the formula $P_0(\theta)Q_{N-1}(\theta) = P_{N-1}(\theta) \prod_{n=1}^{N-1} \exp\left(\frac{\eta}{c}\gamma_n(\omega_n)\right)$, that is,

$$Q_{N-1}(\theta) = \exp\left(\eta \sum_{n=1}^{N-1} \left(\frac{\gamma_n(\omega_n)}{c} - \gamma_n^\theta(\omega_n)\right)\right),$$

and replace $P_{N-1}(\theta)$ by $P_0(\theta)Q_{N-1}(\theta)$ in (1). The inequality transforms to

$$\sum_{\theta \in \Theta} P_0(\theta)Q_{N-1}(\theta) \geq \sum_{\theta \in \Theta} P_0(\theta)Q_{N-1}(\theta) \exp(-\eta \gamma_N^\theta(\omega)) \exp\left(\frac{\eta}{c}\gamma_N(\omega)\right).$$

Finally, we get that (1) is equivalent to the following condition:

$$\exists \gamma_N \in \Sigma_\Lambda \forall \omega \quad \sum_{\theta \in \Theta} P_0(\theta)Q_N(\theta) \leq \sum_{\theta \in \Theta} P_0(\theta)Q_{N-1}(\theta) \quad (3)$$

(for ω_N in Q_N we substitute ω).

In other words, the AA (if realizable for c and η) guarantees that after each step n the value $\sum_{\theta \in \Theta} P_0(\theta)Q_n(\theta)$ does not increase whatever ω_n is chosen by

Reality. Since $\sum_{\theta \in \Theta} P_0(\theta) Q_0(\theta) = \sum_{\theta \in \Theta} P_0(\theta) = 1$, we get $\sum_{\theta \in \Theta} P_0(\theta) Q_N(\theta) \leq 1$ and $Q_N(\theta) \leq 1/P_0(\theta)$ for each step N . To complete the proof it remains to note that

$$Q_N(\theta) = \exp \left(\eta \left(\frac{L_N}{c} - L_N^\theta \right) \right). \quad \square$$

For $c = 1$, the value $\frac{1}{\eta} \ln \sum_{\theta} P_0(\theta) Q_N(\theta)$ is known as the exponential potential (see [3, Sections 3.3, 3.5]) and plays an important role in the analysis of weighted average algorithms.

In the next section we show that the reason why condition (3) holds is essentially that the Q is a supermartingale.

3 Supermartingales

Let $\mathcal{P}(\Omega)$ be the set of all distributions on Ω . Note that since Ω is finite we can identify $\mathcal{P}(\Omega)$ with a $(|\Omega| - 1)$ -dimensional simplex in Euclidean space, with the standard distance and topology. Let E be any set (maybe, empty). A function $S: (E \times \mathcal{P}(\Omega) \times \Omega)^* \rightarrow \mathbb{R}$ is called a (game-theoretic) *supermartingale* if for any N , for any $e_1, \dots, e_N \in E$, for any $\pi_1, \dots, \pi_N \in \mathcal{P}(\Omega)$, for any $\omega_1, \dots, \omega_{N-1} \in \Omega$, it holds that

$$\begin{aligned} \sum_{\omega \in \Omega} \pi_N(\omega) S(e_1, \pi_1, \omega_1, \dots, e_{N-1}, \pi_{N-1}, \omega_{N-1}, e_N, \pi_N, \omega) \\ \leq S(e_1, \pi_1, \omega_1, \dots, e_{N-1}, \pi_{N-1}, \omega_{N-1}). \end{aligned} \quad (4)$$

Remark 1. In the context of algorithmic probability theory (e.g. [10, p. 296]), the word ‘supermartingale’ is used in the following sense. Let $\mu: \Omega^* \rightarrow [0, 1]$ be a measure on Ω^* . A function $s: \Omega^* \rightarrow \mathbb{R}_+$ is a supermartingale with respect to μ if for any N and any $\omega_1, \dots, \omega_{N-1} \in \Omega$ it holds that

$$\sum_{\omega \in \Omega} \mu(\omega \mid \omega_1, \dots, \omega_{N-1}) s(\omega_1, \dots, \omega_{N-1}, \omega) \leq s(\omega_1, \dots, \omega_{N-1}),$$

where $\mu(\omega \mid \omega_1, \dots, \omega_{N-1}) = \frac{\mu(\omega_1, \dots, \omega_{N-1}, \omega)}{\mu(\omega_1, \dots, \omega_{N-1})}$. The relation with the game-theoretic supermartingale notion is the following. Let us take any measure μ . Let e_n be any functions of $\omega_1, \dots, \omega_{n-1}$. Let $\pi_n(\omega)$ be $\mu(\omega \mid \omega_1, \dots, \omega_{n-1})$. Having substituted these functions in a game-theoretic supermartingale S , one gets a probabilistic supermartingale with respect to μ .

A supermartingale S is called *forecast-continuous* if for each N , it is continuous as a function of π_N .

3.1 Two Examples of Supermartingales

Let us consider two examples of supermartingales that naturally arise from two widely used games of prediction.

The *logarithmic loss function* is defined by

$$\lambda(\omega, \gamma) := \begin{cases} -\ln \gamma & \text{if } \omega = 1, \\ -\ln(1 - \gamma) & \text{if } \omega = 0, \end{cases}$$

where $\omega \in \{0, 1\}$ and $\gamma \in [0, 1]$ (notice that the loss function is allowed to take value ∞). The losses in the game are $L_N := \sum_{n=1}^N \lambda(\omega_n, \gamma_n)$ for Learner who predicts γ_n and $L_N^\theta := \sum_{n=1}^N \lambda(\omega_n, \gamma_n^\theta)$ for expert θ who predicts γ_n^θ . Consider an exponent of the difference of losses of Learner and the expert:

$$\exp \left(\eta \sum_{n=1}^N \left(\lambda(\omega_n, \gamma_n) - \lambda(\omega_n, \gamma_n^\theta) \right) \right).$$

Let us assign prediction $\gamma \in [0, 1]$ to each probability distribution $(1 - \gamma, \gamma)$ on $\{0, 1\}$. With this identification, the expression above can be considered as a function on $([0, 1] \times \mathcal{P}(\{0, 1\}) \times \{0, 1\})^*$.

Lemma 1. *For $\eta \in [0, 1]$, the function above is a forecast-continuous supermartingale.*

Proof. The continuity is obvious. For the supermartingale property, it suffices to check that

$$pe^{\eta(-\ln p + \ln g)} + (1 - p)e^{\eta(-\ln(1-p) + \ln(1-g))} \leq 1,$$

i.e., that $p^{1-\eta}g^\eta + (1-p)^{1-\eta}(1-g)^\eta \leq 1$ for all $p, g \in [0, 1]$ (p stands for γ_n and g stands for γ_n^θ). The last inequality immediately follows from the inequality between the geometric and arithmetic means when $\eta \in [0, 1]$. (The left-hand side of that inequality is a special case of what is known as the Hellinger integral in probability theory.) $\square$

In the game with *quadratic loss function*, $\omega \in \{0, 1\}$ and $\gamma \in [0, 1]$ as before, and the losses of Learner and expert θ are $L_N := \sum_{n=1}^N (\gamma_n - \omega_n)^2$ and $L_N^\theta := \sum_{n=1}^N (\gamma_n^\theta - \omega_n)^2$, respectively.

Lemma 2. *For $\eta \in [0, 2]$, the following function on $([0, 1] \times \mathcal{P}(\{0, 1\}) \times \{0, 1\})^*$*

$$\exp \left(\eta \sum_{n=1}^N \left((\gamma_n - \omega_n)^2 - (\gamma_n^\theta - \omega_n)^2 \right) \right)$$

is a forecast-continuous supermartingale.

Proof. It is sufficient to check that

$$pe^{\eta((p-1)^2 - (g-1)^2)} + (1-p)e^{\eta((p-0)^2 - (g-0)^2)} \leq 1$$

for all $p, g \in [0, 1]$. If we substitute $g = p + x$, the last inequality will reduce to

$$pe^{2\eta(1-p)x} + (1-p)e^{-2\eta px} \leq e^{\eta x^2}, \quad \forall x \in [-p, 1-p].$$

The last inequality is a simple corollary of Hoeffding's inequality [8, 4.16], which is true for any $h \in \mathbb{R}$ (cf. [3, Lemma A.1]). Indeed, applying Hoeffding's inequality to the random variable X that is equal to 1 with probability p and to 0 with probability $(1-p)$, we obtain $p \exp(h(1-p)) + (1-p) \exp(-hp) \leq \exp(h^2/8)$, which the substitution $h := 2\eta x$ reduces to $p \exp(2\eta(1-p)x) + (1-p) \exp(-2\eta px) \leq \exp(\eta^2 x^2/2) \leq \exp(\eta x^2)$, the last inequality assuming $\eta \leq 2$. $\square$

3.2 A Supermartingale Criterion If the AA Is Realizable

In Lemmas 1 and 2, we take a certain function of losses, and consider it as a supermartingale by having identified a prediction γ with a distribution $(1-\gamma, \gamma)$. A similar approach works also in the general case.

Let $\alpha : \mathcal{P}(\Omega) \rightarrow \Sigma_A \subseteq [0, \infty]^\Omega$ map any distribution π to a prediction α_π . Given α , a real $c \geq 1$, and a real $\eta > 0$, let us define the following function on $(\Sigma_A \times \mathcal{P}(\Omega) \times \Omega)^*$:

$$Q(e_1, \pi_1, \omega_1, \dots, e_N, \pi_N, \omega_N) = \exp \left(\eta \sum_{n=1}^N \left(\frac{\alpha_{\pi_n}(\omega_n)}{c} - e_n(\omega_n) \right) \right). \quad (5)$$

Note that this very function is used in Lemmas 1 and 2, and also it is the function $Q_N(\theta)$ from the proof of Theorem 1, with e_n standing for γ_n^θ and α_{π_n} standing for γ_n .

Our next goal (Theorems 2 and 3) is to show that the AA is realizable if and only if there exists α such that the function Q is a supermartingale.

Lemma 3. *Let S be a forecast-continuous supermartingale. For any N , for any $e_1, \dots, e_N \in E$, for any $\pi_1, \dots, \pi_{N-1} \in \mathcal{P}(\Omega)$, for any $\omega_1, \dots, \omega_{N-1} \in \Omega$, it holds that*

$$\begin{aligned} \exists \pi \in \mathcal{P}(\Omega) \forall \omega \in \Omega \quad & S(e_1, \pi_1, \omega_1, \dots, e_N, \pi, \omega) \\ & \leq S(e_1, \pi_1, \omega_1, \dots, e_{N-1}, \pi_{N-1}, \omega_{N-1}). \end{aligned}$$

A variant of this lemma was originally proved by Levin [9]. For a full proof see [7, Theorem 6] and [20, Theorem 1].

Note that the property provided by Lemma 3 is essentially the condition (3).

Theorem 2. *Let α be a mapping from $\mathcal{P}(\Omega)$ to Σ_A and $c \geq 1$ and $\eta > 0$ reals such that Q is a forecast-continuous supermartingale. Then the AA is realizable for c and η .*

Proof. Let $G \subseteq A$ be an arbitrary finite set. To prove (2) for any distribution ρ on G , we construct a supermartingale Q^ρ on $(\mathcal{P}(\Omega) \times \Omega)^*$ (the set E in the definition of supermartingale may be empty), which is a ρ -average of Q with g substituted for e_1 ($e_2, e_3, \dots$ may be arbitrary), and apply Lemma 3 for $N = 1$. Namely,

$$Q^\rho(\pi, \omega) = \sum_{g \in G} \rho(g) Q(g, \pi, \omega).$$

By the lemma, there exists $\pi \in \mathcal{P}(\Omega)$ such that $Q^\rho(\pi, \omega) \leq 1$ for all ω , that is,

$$\sum_{g \in G} \rho(g) \exp \left(\eta \left(\frac{\alpha_\pi(\omega)}{c} - g(\omega) \right) \right) \leq 1.$$

Since $\alpha_\pi \in \Sigma_A$, there is $\gamma \in A$ such that $\gamma \leq \alpha_\pi$, which completes the proof. $\square$

For the converse statement, we need three assumptions about A .

Assumption 1. A is a compact set.

Assumption 2. There is $\gamma \in A$ such that $\gamma(\omega) < \infty$ for all ω .

These assumptions are standard (see [16]). The third assumption is new and very technical. First we introduce some definitions that will be useful also in the proof of the theorem below.

For a given η , the *exp-convex hull* of Σ_A is the set $\Xi_A \supseteq \Sigma_A$ that consists of all points $g \in [0, \infty]^\Omega$ of the form

$$g(\omega) = \log_{(e^{-\eta})} \sum_{\gamma \in G} \rho(\gamma) (e^{-\eta})^{\gamma(\omega)} = -\frac{1}{\eta} \ln \sum_{\gamma \in G} \rho(\gamma) \exp(-\eta \gamma(\omega))$$

for all $\omega \in \Omega$, where G is a finite subset of Σ_A and ρ is a distribution on G .

Let Ξ' be the set of minimal elements of Ξ_A : $f \in \Xi'$ if and only if for any $g \in \Xi \forall \omega (g(\omega) \leq f(\omega))$ implies $f = g$. Notice that Ξ' is contained in the boundary of Ξ_A . It is known that the game is η -mixable if and only if $\Xi_A \subseteq \Sigma_A$ (which explains the name) if and only if $\Xi' \subseteq A$.

For $\pi \in \mathcal{P}(\Omega)$ and $g \in [0, \infty]^\Omega$, denote

$$E_\pi g = \sum_{\omega \in \Omega} \pi(\omega) g(\omega).$$

Assumption 3. Let $\pi \in \mathcal{P}(\Omega)$ be such that $\pi(\omega_1) = 0$ and $\pi(\omega_2) = 0$ for some $\omega_1 \neq \omega_2$. Let $m = \min_{\gamma \in \Xi'} E_\pi \gamma$. If $E_\pi \gamma_1 = m$ and $E_\pi \gamma_2 = m$ for some $\gamma_1, \gamma_2 \in \Xi'$, then either γ_1 minorizes γ_2 or vice versa.

Assumption 3 is rather awkward. But it holds for a wide class of games. In particular, it holds for all binary games and for all proper scoring rules. (But it does not hold, e.g., for non-binary “absolute-loss” game, where $\lambda(\omega, \gamma) = \sum_{i=1}^n |\omega_i - \gamma_i|$.)

On the other hand, some technical requirement of this kind is unavoidable. It does not appear just from our proof: For almost all $\pi \in \mathcal{P}(\Omega)$ there is a unique α_π such that Q defined by (5) is a supermartingale. And it is easy to construct an example where this correspondence cannot be extended to a continuous mapping from $\mathcal{P}(\Omega)$ to Ξ' . (One possible way is to consider the image of Ξ_A under point-wise exponential mapping $g \mapsto e^{-\eta g}$. Every $\pi \in \mathcal{P}(\Omega)$ can be naturally identified with a family of parallel hyperplanes. The inequality (4) actually means that α_π is a point of tangency, where one of the hyperplanes touches the image of Ξ_A . To get an example with a point of discontinuity, one may consider a vertical cylinder in the three-dimensional space and a horizontal hyperplane.) It is not clear how to cope with such cases under the supermartingale approach.

Theorem 3. *Let the game (Ω, Λ) satisfy Assumptions 1–3. If the AA is realizable for certain c and η , then there is a mapping $\alpha : \mathcal{P}(\Omega) \rightarrow \Sigma_\Lambda$ such that for these α , c , and η , Q defined by (5) is a forecast-continuous supermartingale.*

Proof. We construct the mapping α in two steps. First, we map $\mathcal{P}(\Omega)$ to Ξ' , and then we map Ξ' to the boundary of Σ_Λ . Geometrically, these steps amount to taking a tangent plane to Ξ' and the central projection from Ξ' to the boundary of Σ_Λ .

The mapping μ from $\mathcal{P}(\Omega)$ to Ξ' is defined by the formula

$$\mu_\pi = \arg \min_{g \in \Xi'} E_\pi g.$$

First, let us prove that μ_π is well-defined. By Assumption 2, there is a finite point in Λ and thus in Ξ' , hence the min is finite. The minimum is attained since Ξ_Λ is compact and $E_\pi g_1 \leq E_\pi g_2$ if $g_1 \leq g_2$. Let us prove that the minimum is attained at one point only¹. Assume the converse: for some π , there are at least two different points f and g where the minimum is attained, and $E_\pi f = E_\pi g = m$. By definition of Ξ , the point $-\ln((e^{-\eta f(\omega)} + e^{-\eta g(\omega)})/2)/\eta$ also belongs to Ξ . If it does not belong to Ξ' , there is a point $h \in \Xi'$ that minorizes it. For any reals x, y , we have $(e^x + e^y)/2 \geq e^{(x+y)/2}$, and the inequality is strict if $x \neq y$. Therefore, $h(\omega) \leq (f(\omega) + g(\omega))/2$, and if there exists ω such that $\pi(\omega) \neq 0$ and $f(\omega) \neq g(\omega)$, then $E_\pi h < E_\pi(f + g)/2 = m$. This contradiction proves that if $f(\omega) \neq g(\omega)$, then $\pi(\omega) = 0$. If f and g differ at one ω only, then one of them minorizes the other, thus only one can belong to Ξ' . The remaining case is directly forbidden by Assumption 3.

Let us prove that μ is continuous. Consider any sequence of π_i converging to π . First of all, prove that $E_{\pi_i} \mu_{\pi_i}$ converges to $E_\pi \mu_\pi$. Indeed, for any $\pi, \pi' \in \mathcal{P}(\Omega)$, we have $E_{\pi'} \mu_{\pi'} \leq E_{\pi'} \mu_\pi \leq E_\pi \mu_\pi + \max_\omega \mu_\pi(\omega) \sum_\omega |\pi'(\omega) - \pi(\omega)|$. Hence $|E_{\pi_i} \mu_{\pi_i} - E_\pi \mu_\pi| \leq \sum_\omega |\pi'(\omega) - \pi(\omega)| \times \max\{\mu_\pi(\omega), \mu_{\pi_i}(\omega) \mid \omega \in \Omega\}$, and the last expression tends to zero as π_i tends to π . We omitted several subtle points: how to bound μ_{π_i} and how to cope with the case of $\mu_\pi(\omega) = \infty$ (e.g., consider an auxiliary sequence of finite points converging to μ_π).

Now assume a sequence π_i converges to π and μ_{π_i} converges to some γ . By Assumption 1, γ belongs to Ξ_Λ . Further, $E_{\pi_i} \gamma$ converges to $E_\pi \gamma$ since their difference is bounded by $\max_\omega \gamma(\omega) \sum_\omega |\pi_i(\omega) - \pi(\omega)|$, and $E_{\pi_i}(\mu_{\pi_i} - \gamma)$ converges to zero since it is bounded by $\max |\mu_{\pi_i} - \gamma|$. Thus, $E_{\pi_i} \mu_{\pi_i}$ converges to $E_\pi \gamma$ and to $E_\pi \mu_\pi$, and $\gamma = \mu_\pi$ due to the uniqueness of μ_π .

Now let us construct a continuous mapping from Ξ' to Σ_Λ . Let $g \in \Xi_\Lambda$. By definition, it is a positive combination of some $\gamma \in \Sigma_\Lambda$. For $g(\omega) = 0$ for some ω , then $\gamma(\omega) = 0$ too for any γ from the combination. For ω such that $g(\omega) \neq 0$, there is a constant $c > 0$ such that $cg(\omega) \geq \gamma(\omega)$ for all γ from the combination. Taking the maximal such c over all ω , we get that for any $g \in \Xi_\Lambda$ there is a constant $c > 0$ such that $cg \in \Sigma_\Lambda$. Now take the minimal $c > 0$ such that $cg \in \Sigma_\Lambda$, this c will be called $C(g)$ (the minimum is attained due to

¹ In Bayesian framework this uniqueness means that the loss function is a strictly proper scoring rule, cf. [4].

Assumption 1). Clearly, the mapping $g \mapsto C(g)g$ is a continuous mapping from Ξ_A (and thus from Ξ') to the boundary of Σ_A . If the AA is realizable for some c (and η —recall that Ξ' depends on η), then $C(g) \leq c$ for all $g \in \Xi'$.

Let $\alpha_\pi = C(\mu_\pi)\mu_\pi$. Clearly, α_π is continuous and so is Q .

It remains to check that Q is a supermartingale, that is, satisfy (4). Dividing both sides by the right-hand side, we get that it suffices to check the following:

$$\sum_{\omega \in \Omega} \pi(\omega) \exp \left(\eta \left(\frac{\alpha_\pi(\omega)}{c} - \gamma(\omega) \right) \right) \leq 1,$$

for any $\gamma \in \Sigma_A$ and any $\pi \in \mathcal{P}(\Omega)$. This inequality will follow from

$$\sum_{\omega \in \Omega} \pi(\omega) e^{\eta(\mu_\pi(\omega) - \gamma(\omega))} \leq 1 \quad (6)$$

since $\alpha_\pi \leq c\mu_\pi$ (here we use that the AA is realizable for c and η). Take $\epsilon > 0$. Consider $-\frac{1}{\eta} \ln((1 - \epsilon)e^{-\eta\mu_\pi} + \epsilon e^{-\eta\gamma})$. By definition, this mixture belongs to Ξ .

When $\epsilon \rightarrow 0$, we have

$$\begin{aligned} -\frac{1}{\eta} \ln((1 - \epsilon)e^{-\eta\mu_\pi} + \epsilon e^{-\eta\gamma}) &= \mu_\pi - \frac{1}{\eta} \ln \left(1 + \epsilon \left(e^{\eta(\mu_\pi - \gamma)} - 1 \right) \right) \\ &= \mu_\pi - \frac{\epsilon}{\eta} \left(e^{\eta(\mu_\pi - \gamma)} - 1 \right) + o(\epsilon^2). \end{aligned}$$

Take the expectation E_π of the last expression. If (6) does not hold, this expectation is less than $E_\pi \mu_\pi$ for sufficiently small ϵ , which contradicts the definition of μ_π . $\square$

Remark 2. For any $g \in \Xi'$, we have $C(g) \geq 1$ since $\Xi_A \supseteq \Sigma_A$ and g is minimal in Ξ_A . Thus for η -mixable games ($c = 1$) we have $\alpha_\pi = \mu_\pi$ for all π , and the image of the mapping α is included in Λ . For arbitrary games, the image is included in the boundary of Σ_A . Note also that α is continuous.

3.3 Defensive Forecasting

Now we describe the *Defensive Forecasting* algorithm (DF) for the game of prediction with expert advice. It has five parameters: reals $c \geq 1$, $\eta > 0$, a function $\alpha : \mathcal{P}(\Omega) \rightarrow \Sigma_A$, a distribution P_0 on Θ , and a substitution function $\sigma : \Sigma_A \rightarrow \Lambda$ such that $\sigma(g) \leq g$ for all $g \in \Sigma_A$.

The parameters c , η , and α are such that the function Q is a forecast-continuous supermartingale. Let

$$Q^{P_0}(\{\gamma_1^\theta\}_{\theta \in \Theta}, \pi_1, \omega_1, \dots) = \sum_{\theta \in \Theta} P_0(\theta) Q(\gamma_1^\theta, \pi_1, \omega_1, \dots).$$

Clearly, Q^{P_0} is also a forecast-continuous supermartingale, therefore Lemma 3 applies.

At step N , the DF takes any π_N that satisfy the conclusion of Lemma 3, and then announces $\gamma_N = \sigma(\alpha_{\pi_N})$ as the next prediction of Learner.

Theorem 4. *If Q is a forecast-continuous supermartingale for c, η , and α , then the DF with parameters c, η, α, P_0 , and σ guarantees that at each step n for all experts θ*

$$L_n \leq cL_n^\theta + \frac{c}{\eta} \ln \frac{1}{P_0(\theta)}.$$

Proof. Lemma 3 guarantees that at each step Q^{P_0} is not greater than its initial value, 1. Thus,

$$\exp \left(\eta \sum_{n=1}^N \left(\frac{\alpha_{\pi_n}(\omega_n)}{c} - \gamma_n^\theta(\omega_n) \right) \right) \leq \frac{1}{P_0(\theta)},$$

and therefore

$$\sum_{n=1}^N \alpha_{\pi_n}(\omega_n) \leq cL_n^\theta + \frac{c}{\eta} \ln \frac{1}{P_0(\theta)}.$$

It remains to note that $\gamma_n = \sigma(\alpha_{\pi_n}) \leq \alpha_{\pi_n}$, and hence $L_n \leq \sum_{n=1}^N \alpha_{\pi_n}(\omega_n)$. $\square$

As we have seen, the AA and the DF are very close in the loss bound and also in their procedure. We can say even more: with the same parameters and inputs, they give the same predictions. More precisely, two sets coincide: the set of γ_N satisfying (1) and the set of γ_N such that they minorize α_{π_N} for π_N satisfying the conclusion of Lemma 3. Thus, within the standard setting of the prediction with expert advice, the DF is just another way of looking at the AA. In the next section, we consider a setting where these two algorithms differ.

4 Second-Guessing Experts

Let us consider an extension of the protocol of prediction with expert advice. The game is specified by the same elements (Ω, A) as above.

PREDICTION WITH SECOND-GUESSING EXPERT ADVICE

$L_0 := 0$.

$L_0^\theta := 0, \theta \in \Theta$.

FOR $n = 1, 2, \dots$:

Experts $\theta \in \Theta$ announce $\gamma_n^\theta : A \rightarrow A$.

Learner announces $\gamma_n \in A$.

Reality announces $\omega_n \in \Omega$.

$L_n := L_{n-1} + \gamma_n(\omega_n)$.

$L_n^\theta := L_{n-1}^\theta + \gamma_n^\theta(\gamma_n)(\omega_n)$.

END FOR.

The new protocol contains only one substantial change. Informally speaking, now the experts announce not their actual predictions, but conditional statements that specify their predictions depending on Learner's next step. Therefore, the loss of each expert is determined by the prediction of Learner as well as by

the outcome chosen by Reality. We will call the experts in this protocol *second-guessing experts*. Second-guessing experts are a generalization of experts in the standard protocol: a standard expert can be interpreted in the new protocol as a constant function.

The phenomenon of “second-guessing experts” occurs in real-world finance. For example, commercial banks serve as “second-guessing experts” for a central bank when they use variable interest rates (that is, the interest rate for the next period is announced as an explicit function of the central bank base rate).

In game theory, there is a notion of internal regret [6], which is related to the idea of second-guessing experts. The internal regret appears in the framework where for each prediction, which is called action in this context, there is an expert that consistently recommends this action, and Learner follows one of the experts at each step. The internal regret for a pair of experts (i, j) shows by how much Learner could decrease its loss by having followed expert j each time it followed expert i . This can be modelled by a second-guessing expert that agrees with Learner if Learner does not follow i , and recommends following j when the Learner follows i .

The internal regret is studied in randomized prediction protocols. In our case of deterministic Learner’s predictions, one cannot hope to get any interesting loss bound without additional assumptions. Indeed, Experts can always suggest exactly the “opposite” to the Learner’s prediction (for example, in the log loss game, predict 0 if Learner predicts $\gamma_n \leq 0.5$ and 1 otherwise), and Reality can “agree” with them; then the Experts’ losses remain zero, but the Learner’s loss grows linearly in the number of steps. In this paper we consider second-guessing experts that depend on the prediction of Learner *continuously*.

4.1 The DF for Second-Guessing Experts

First consider the case when γ_n^θ are continuous mappings from Λ to Λ . Surprisingly, the DF requires virtually no modifications.

Theorem 5. *Suppose Q defined by (5) is a forecast-continuous supermartingale for some c, η , and a continuous $\alpha : \mathcal{P}(\Omega) \rightarrow \Lambda$. Then a DF algorithm can be applied in the protocol of prediction with second-guessing expert advice for continuous experts; it guarantees the same loss bound as in the prediction with expert advice protocol.*

Proof. Let Q be a forecast-continuous supermartingale as a function on $(\Sigma_\Lambda \times \mathcal{P}(\Omega) \times \Omega)^*$. Let γ_n^θ be continuous functions $\Lambda \rightarrow \Lambda$. Since α is continuous, $\gamma_n^\theta(\alpha_\pi)$ is a continuous function $\mathcal{P}(\Omega) \rightarrow \Lambda$. Then Q with $\gamma_n^\theta(\alpha_\pi)$ substituted for e_n is a forecast-continuous supermartingale as a function on $(C(\Lambda \rightarrow \Lambda) \times \mathcal{P}(\Omega) \times \Omega)^*$, where $C(\Lambda \rightarrow \Lambda)$ is the set of continuous functions on Λ . We take the identity for a substitution function since α takes values already in Λ , and for the DF with the supermartingale above the proof of Theorem 4 applies. $\square$

Recall that for mixable games the mapping α constructed in the proof of Theorem 3 satisfies the conditions of the last theorem.

For games that are not mixable, it may happen that such α does not exist. Even more, for games where Λ is not connected (e.g., the simple game of prediction), the continuity of experts does not rule out the example with “opposite” predictions.

To cope with such cases, we modify the protocol of the game. Namely, we allow Experts and Learner to announce predictions from the boundary Σ'_Λ of Σ_Λ . That is, Experts are $\gamma_n^\theta : \Sigma'_\Lambda \rightarrow \Sigma'_\Lambda$, and Learner is $\gamma_n \in \Sigma'_\Lambda$. Theorem 5 requires minimal changes: now α is a function from $\mathcal{P}(\Omega)$ to Σ'_Λ , and the proof is modified accordingly. Theorem 3 supplies us with the α required.

4.2 The AA for Second-Guessing Experts

The AA cannot be applied to the second-guessing protocol directly. Recall the AA is based on the inequality (1), which is already resolved for γ_N . In the second-guessing protocol, the inequality contains γ_N on both sides:

$$\gamma_N(\omega_N) \leq -\frac{c}{\eta} \ln \sum_{\theta \in \Theta} \frac{P_{N-1}(\theta)}{\sum_{\theta \in \Theta} P_{N-1}(\theta)} \exp(-\eta \gamma_N^\theta(\gamma_N)(\omega_N)). \quad (7)$$

The DF implicitly solves this inequality within (the proof of) Lemma 3, which is a kind of fixed point theorem. We present a modification of the AA which uses a fixed point theorem explicitly. We use the following theorem (see e.g. [1]).

Theorem 6 (Brouwer). *If X is homeomorphic to a closed simplex, and $F : X \rightarrow X$ is a continuous function, then F has a fixed point.*

Our goal is to find a subset X of possible Learner’s predictions such that X is homeomorphic to a closed simplex, and a continuous function $G : X \rightarrow X$ such that for any $\gamma \in X$, the point $G(\gamma)$ is not greater than the right-hand side of (7) with γ substituted for γ_N . Then the modified AA works as follows: at each step, the AA constructs G and X (they may depend on the step and history), finds a fixed point $\gamma = G(\gamma)$ via the Brouwer theorem, and announces γ as the next prediction of Learner. Since any fixed point of G is a solution of (7), the standard analysis of the AA applies providing the same loss bound.

We construct the domain X and the function G in a different manner for mixable games with the original second-guessing protocol and for non-mixable games and the modified protocol.

Obviously, Ξ' is homeomorphic to a closed simplex (note that the “exponential image” $e^{-\eta \Xi_\Lambda}$ of Ξ_Λ is a bounded convex subset of $\mathbb{R}^{|\Omega|}$). For mixable games, $\Xi' \subseteq \Lambda$, and we let $X = \Xi'$ in this case. For non-mixable games, we consider the mapping $g \mapsto C(g)g$ used in the proof of Theorem 3, which is a homeomorphism of Ξ' to a part of Σ'_Λ . This part we take as X .

Now let us construct the function G . The beginning is common for both versions. A point $\gamma \in X$ is mapped to the point g such that

$$g(\omega) = -\frac{1}{\eta} \ln \sum_{\theta \in \Theta} \frac{P_{N-1}(\theta)}{\sum_{\theta \in \Theta} P_{N-1}(\theta)} \exp(-\eta \gamma_N^\theta(\gamma)(\omega))$$

for all ω . The point g belongs to Ξ_Λ by definition.

Lemma 4. *There is a continuous mapping $F : \Xi_A \rightarrow \Xi'$ such that for any $g \in \Xi_A$, it holds that $F(g) \leq g$.*

We postpone the proof and continue the construction of G . This F maps g to $F(g) \in \Xi'$. For mixable games, we are done. For non-mixable games, we need one more step: apply to $F(g)$ the homeomorphism from Ξ' to Σ'_A . The function G has the correct range by construction and is continuous as a composition of continuous mappings. The point $G(\gamma)$ is not greater than the right-hand side of (7) since $C(F(g))F(g) \leq cF(g) \leq cg$. Thus, we obtained the following

Theorem 7. *If the AA is realizable for the game (Ω, Λ) in the prediction with expert advice for some c and η , then the AA with fixed point is realizable for the same game for the prediction with second-guessing expert advice protocol (modified—for non-mixable games) for the same c and η , and guarantees the same loss bounds.*

Proof of Lemma 4. We construct a continuous mapping $F : \Xi_A \rightarrow \Xi'$ as a composition of mappings F_ω for all $\omega \in \Omega$. Each F_ω when applied to $g \in \Xi_A$ preserves the values of $g(o)$ for $o \neq \omega$ and decreases as far as possible the value $g(\omega)$ so that the result is still in Ξ_A . Formally, $F_\omega(g) = g'$ such that $g'(o) = g(o)$ for $o \neq \omega$ and $g'(\omega) = \min\{f(\omega) \mid f \in \Xi_A, \forall o \neq \omega f(o) = g(o)\}$.

Let us show that F_ω is continuous. It suffices to show that $F_\omega(g)(\omega)$ depends continuously on g , since the other coordinates do not change. We will show that $F_\omega(g)(\omega)$ is concave in g , continuity follows easily (see, e.g. [11]). Indeed, take any $t \in [0, 1]$, and $f, g \in \Xi_A$. Since Ξ_A is convex (and even exp-convex), then $tf + (1-t)g \in \Xi_A$ and $tF_\omega(f) + (1-t)F_\omega(g) \in \Xi_A$. The latter point has all the coordinates $o \neq \omega$ the same as the former. Thus, by definition of F_ω , we get $F_\omega(tf + (1-t)g)(\omega) \leq (tF_\omega(f) + (1-t)F_\omega(g))(\omega) = tF_\omega(f)(\omega) + (1-t)F_\omega(g)(\omega)$, which was to be shown.

All F_ω do not increase the coordinates. Since the set Ξ_A contains any point g with all its majorants, $F_\omega(f) = f$ implies that $F_\omega(g) = g$ for any g obtained from f by applying any $F_{\omega'}$. Therefore, the image of a composition of F_ω over all $\omega \in \Omega$ is included in Ξ' . $\square$

Remark 3. Actually, Lemma 4 constructs a continuous substitution function. In many natural games, the standard substitution functions appear to be continuous. In particular, for the log loss function,

$$(g_0, g_1) \mapsto ((-g_0 + g_1) \ln(e^{-g_0 + g_1} + 1), (-g_0 + g_1) \ln(e^{g_0 - g_1} + 1)).$$

For the quadratic loss function,

$$(g_0, g_1) \mapsto \left(\left(\frac{1 + g_0 - g_1}{2} \right)^2, \left(\frac{1 - g_0 + g_1}{2} \right)^2 \right).$$

Acknowledgements. This work was partly supported by EPSRC grant EP/F002998/1. Comments by Alex Gammernan, Glenn Shafer, Alexander Shen, and the anonymous referees have helped us improve the presentation.

References

- [1] Agarwal, R., Meehan, M., O'Regan, D.: Fixed Point Theory and Applications. Cambridge Tracts in Mathematics, vol. 141. Cambridge University Press, Cambridge (2001)
- [2] Blum, A., Mansour, Y.: From External to Internal Regre. *J. Mach. Learn. Res.* 8, 1307–1324 (2007)
- [3] Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, Cambridge (2006)
- [4] Dawid, A.P.: The geometry of proper scoring rules. *Annals of the Institute of Statistical Mathematics* 59, 77–93 (2007)
- [5] Foster, D., Vohra, R.: Asymptotic calibration. *Biometrika* 85, 379–390 (1998)
- [6] Foster, D., Vohra, R.: Regret in the online decision problem. *Games Econ. Behav.* 29, 104–130 (1999)
- [7] Gács, P.: Uniform test of algorithmic randomness over a general space. *Theoretical Computer Science* 341, 91–137 (2005)
- [8] Hoeffding, W.: Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association* 58, 13–30 (1963)
- [9] Levin, L.: Uniform tests of randomness. *Soviet Mathematics Doklady* 17, 337–340 (1976)
- [10] Li, M., Vitányi, P.: An Introduction to Kolmogorov Complexity and Its Applications, 2nd edn. Springer, New York (1997)
- [11] Rockafellar, R.: Convex Analysis. Princeton University Press, Princeton (1996)
- [12] Shafer, G., Vovk, V.: Probability and Finance: It's Only a Game. Wiley, New York (2001)
- [13] Stoltz, G., Lugosi, G.: Internal Regret in On-Line Portfolio Selection. *Machine Learning* 59, 125–159 (2005)
- [14] Stoltz, G., Lugosi, G.: Learning correlated equilibria in games with compact sets of strategies. *Games and Economic Behavior* 59, 187–209 (2007)
- [15] Vovk, V.: Aggregating Strategies. In: Fulk, M., Case, J. (eds.) *Proceedings of the Third Annual Workshop on Computational Learning Theory*, San Mateo, CA, pp. 371–383 (1990)
- [16] Vovk, V.: A game of prediction with expert advice. *Journal of Computer and System Sciences* 56, 153–173 (1998)
- [17] Vovk, V.: Competitive on-line learning with a convex loss function. Technical Report [arXiv:cs/0506041v3](#) [cs.LG], [arXiv.org](#) e-Print archive (September 2005)
- [18] Vovk, V.: On-line regression competitive with reproducing kernel Hilbert spaces. Technical Report [arXiv:cs/0511058v2](#) [cs.LG], [arXiv.org](#) e-Print archive (January 2006)
- [19] Vovk, V.: Metric entropy in competitive on-line prediction. Technical Report [arXiv:cs/0609045v1](#) [cs.LG], [arXiv.org](#) e-Print archive (September 2006)
- [20] Vovk, V.: Continuous and randomized defensive forecasting: unified view. Technical Report [arXiv:0708.2353v2](#) [cs.LG], [arXiv.org](#) e-Print archive (August 2007)

Aggregating Algorithm for a Space of Analytic Functions

Mikhail Dashevskiy

Computer Learning Research Centre,
Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK
`mikhail@cs.rhul.ac.uk`

Abstract. In this paper the problem of Prediction with Expert Advice is considered. We apply an existing algorithm, the Aggregating Algorithm, to a specific class of experts. This class of experts approximates (with respect to its parameter) the class of continuous functions and in this way it is close to a natural way of describing a possible dependence between two variables (continuous). We develop an explicit algorithm and prove an upper bound on the difference between the loss of our algorithm and the loss of the best expert, which has the order of the squared logarithm of the number of steps of the algorithm. This bound lies between existing bounds which have the form of the logarithm of the number of steps and the square root of the number of steps. Having more sets (algorithm, class of experts, upper bound) helps in choosing an appropriate way of solving the problem of Prediction with Expert Advice for a particular application.

1 Introduction

Let us consider the problem of Prediction with Expert Advice ([1, 3, 4, 9]), i. e. given a class of experts our algorithm outputs predictions which are not much worse (in a sense) than the best expert. Ideally, we would like to develop an algorithm which could compete with as wide a class of experts as possible being as close to the best expert as possible. The problem of Prediction with Expert Advice can be divided into two parts: choosing an appropriate class of experts and using the experts to construct an algorithm with high performance. To develop an algorithm with high overall performance it is important to successfully solve both parts of the problem.

To build a successful algorithm which solves the problem of Prediction with Expert Advice it is essential to choose a wide class of experts. In this paper we use a class of analytical functions, which approximates the space of continuous functions, as the space of experts, i. e. in the limit our space of experts (which depends on two parameters) is dense in the space of continuous functions. The problem of competing with classes of experts dense in the space of continuous functions is also studied in [1, 3, 8].

There are many ideas regarding how to use experts so that we receive a prediction that is not much worse than those given by the experts. In this paper we use

the idea of the Aggregating Algorithm (AA, [9]) and find an explicit algorithm that uses the functions from our space of experts (which is infinite dimensional). Currently there exist only explicit algorithms for the AA aggregating experts from finite-dimensional classes of experts. We hope that the ideas used in this paper can give some directions for creating an algorithm, which competes with all continuous functions, and has high performance.

The Aggregating Algorithm was applied to the space of linear functions (the case of Linear Regression) in [9]. We extend the result for the case of a wider class of experts. The performance of the algorithms solving the problem of Prediction with Expert Advice is measured by the regret term, the absolute difference between the total loss and the loss of the best expert. The AA for Regression has the regret term in the form of $O(\log T)$ (where T is the number of steps of the algorithm). The fact that our class of experts is wider worsens the regret term and the latter has the form of $O(\log^2 T)$.

The current algorithms can be divided into two groups according to the form of the regret term: those with the regret form of $O(\log T)$ ([9, 2, 5]) and those with the regret form of $O(\sqrt{T})$ ([4, 11]). Of course, the form of the regret term depends on the class of experts: the same algorithm can have different upper bounds on the regret term for different classes of experts. Besides the Aggregating Algorithm for Regression which has the regret term in the form of $O(\log T)$, there are the Forwarding Algorithm and the Incremental Off-line Algorithm from [2] and the Exponential Weights Algorithm (see [5]) with the same main term in the upper bound on the regret term.

There is another group of algorithms, those having $O(\sqrt{T})$ as an upper bound on their regret term. The Defensive Forecasting algorithm, which employs the idea of using Probability Theory results in the Game-Theoretical form, is one of them ([11]). Another example of such algorithms is the Aggregating Algorithm. These algorithm when competing with some infinite-dimensional benchmark classes (Reproducing Kernel Hilbert Spaces) have an upper bound (see [8]) on the regret term in the form of $O(\sqrt{T})$.

Our algorithm provides an upper bound on the regret term different from those which have traditional algorithms. Our result elaborates the result in [10] (where a bound in the form of $O(\log^2 T)$ is found) and an explicit algorithm is introduced and an explicit bound is found (i. e. a bound in the form of a polynomial of $\log T$). This is the main result of this paper: we developed an algorithm with a new upper bound on the regret term, which lies between existing results.

A natural way of representing experts is associating each expert with a continuous function. Saying that the dependence of two variables is a continuous function is usually not a strong constraint in applications. It would be of great benefit if we could guarantee that our learning algorithm performs not much worse than any expert in the form of continuous functions. In this paper we describe an algorithm that competes (effectively) with a class of experts, which approximates (in the limit) continuous bounded functions.

The other problem that we are trying to solve (or approximate to solving it) in this paper is the problem of the correspondence between the class of experts with which an algorithm competes and the form of the regret term of the algorithm for this class of experts. For example, it is known that the Aggregating Algorithm for Regression and the Exponential Weights Algorithm have the regret term in the form of $\log T$. Here we try to contribute to solving the problem of finding a correspondence between the two properties of an algorithm (we could use this information to choose the parameters of the algorithm as there is a trade-off between the width of the space of experts and the performance of the algorithm).

The rest of the paper is organized as follows. The next section describes the Aggregating Algorithm which we apply to a functional space and states an upper bound on the total loss. Section 3 is devoted to the application of the Aggregating Algorithm to the functional space where we use some particular properties of the selected space and introduce our algorithm. Section 4 gives an estimate on the upper bound of the algorithm and its proof. Section 5 concludes the paper and gives some directions for further research.

2 Aggregating Algorithm

In this part of the paper we consider one of the existing algorithms for solving the problem of Prediction with Expert Advice. The problem can be describe in the form of a game. In each round of the game: experts give their predictions, we look at the experts' predictions in order to give our prediction, the event happens (we see what happened in real life), and the losses of each expert and of our algorithm are calculated. Our aim is to create an algorithm that has total loss (the cumulative loss of all rounds of the game) is not much higher than the total loss of the best expert.

Now we will give a more formal definition of the Prediction Game. We consider here a game between three players, Nature, Experts, and Learner. Let Σ be a set called *signal space*, Ω be a set called *sample space*, Γ be a set called the *decision space*, and Θ be a measurable space called the *parameter space*. A *signal* is an element of the signal space, an *outcome* is an element of the sample space and a *decision* is an element of the decision space. The *loss function* $\lambda : \Omega \times \Gamma \rightarrow [0, \infty]$, the function which measures the performance of Experts and Learner is also part of the game. The following perfect-information game is considered: on each step $t = 1, 2, \dots$

- Nature outputs a signal $x_t \in \Sigma$.
- Experts make measurable predictions $\xi_t : \Theta \rightarrow \Gamma$; $\xi_t(\theta)$ is the prediction corresponding to the parameter $\theta \in \Theta$.
- Learner makes his own prediction $\gamma_t \in \Gamma$.
- Nature chooses an outcome $\omega_t \in \Omega$.
- Experts' loss calculation $L_T(\theta) = \sum_{t=1}^T \lambda(\omega_t, \xi_t(\theta))$.
- Learner's loss calculation $L_T = \sum_{t=1}^T \lambda(\omega_t, \gamma_t)$.

Learner wants to minimize the regret term, i. e. the difference between the total loss of the algorithm and the total loss of the best expert.

In this paper we consider the case $\Omega \subset \mathbb{R}, \Gamma \subset \mathbb{R}$. The Aggregating Algorithm (AA) works as follows: on each step it re-computes weights of the experts (their probability distribution) and mixes all experts according to this distribution. Thus AA gets a mixture of the experts' predictions (a *generalized prediction*), after that it finds the best (in some sense) prediction from Θ . The AA uses the parameters $\eta > 0$ learning rate, β is defined to be $e^{-\eta}$, P_0 is probability distribution on the set Θ of the experts. Intuitively P_0 is the prior distribution, which specifies the initial weights assigned to the experts. At each step the AA performs the following operations:

1. Weights re-computation according to the previous losses:

$$P_{t-1}(d\theta) = \beta^{\lambda(\omega_{t-1}, \xi_{t-1}(\theta))} P_{t-2}(d\theta), \quad \theta \in \Theta. \quad (1)$$

2. Predictions' mixing according to their weights:

$$g_t(\omega) = \log_{\beta} \int_{\Theta} \beta^{\lambda(\omega, \xi_t(\theta))} P_{t-1}^*(d\theta),$$

where $P_{t-1}^*(d\theta)$ is a normalized probability distribution, i. e.

$$P_{t-1}^*(d\theta) = \frac{P_{t-1}(d\theta)}{P_{t-1}(\Theta)}$$

and if $P_{t-1}(\Theta) = 0$, AA is allowed to choose any prediction. On this step we get $g_t(\omega)$ which is a function $\Omega \rightarrow \mathbb{R}$.

3. Looking for an appropriate prediction from Γ :

$$\gamma_t(g_t).$$

In [9] (Lemma 2) it is proved that in the case of the *square-loss function*, i. e. $\lambda(\omega, \gamma) = (\omega - \gamma)^2$, $\eta \leq \frac{1}{2Y}$ (where Y is a bound for $|y_t|$)

$$\gamma_t = \frac{g_t(-Y) - g_t(Y)}{4Y} \quad (2)$$

is a prediction from Θ .

The formula (2) gives us the final prediction on step t . Let Nature's signal be $x_t \in \mathbb{R}^M$, Nature's outcome be $y_t \in [-Y, Y]$ and the prediction be $\gamma_t \in \mathbb{R}$. Let $L_T(\theta)$ be the total loss corresponding to the expert with parameter $\theta \in \Theta$ over T steps and $L_T(AA(N, a))$ be the total loss of the Aggregating Algorithm with parameters N (the dimension of the Parameter Space) and a (a real number). The following theorem from [9] shows an estimate of the performance of the AA:

Proposition 1 (Vovk, 2001). *In the square-loss game there exists an algorithm which satisfies: for any class of experts Θ ,*

$$L_T(AA(n, a)) \leq \inf_{\theta} (L_T(\theta) + a \|\theta\|^2) + Y^2 \ln \det \left(I + \frac{1}{a} \sum_{t=1}^T x_t x_t' \right).$$

Let

$$X = \max_{1 \leq t \leq T} (\|x_t\|_{\infty}),$$

then $\forall T$

$$L_T(AA(n, a)) \leq \inf_{\theta} (L_T(\theta) + a \|\theta\|^2) + NY^2 \ln \left(\frac{TX^2}{a} + 1 \right),$$

where I is the identity matrix, $a > 0$ and N is the dimension of Θ .

Intuitively, a is a constant reflecting our prior expectations about the complexity $\|\theta\| = \sqrt{\sum_{i=j}^N \theta_j^2}$ of successful experts.

3 Application to $A_h(C)$

In [9] the AA was applied to the class of linear functions. Here we apply the algorithm to a wider class of functions.

Definition 1 ([7]). Let $A_h(C)$ be the class of all functions $\mathbb{C} \rightarrow \mathbb{R}$ defined on the stripe with width $h : |\operatorname{Im} z| \leq h$, periodic with the period 2π , analytical and bounded there by a constant C .

In this paper we apply AA to the functional space defined above. Using some particular properties of $A_h(C)$, we get an explicit form of the AA for $A_h(C)$ and prove an upper bound on its performance.

Any function $f \in A_h(C)$ can be written down in the Fourier series form:

$$f(x) = \sum_{k=-\infty}^{+\infty} c_k e^{ikx}.$$

We call c_k Fourier coefficients.

Proposition 2. For any $f \in A_h(C)$, $f(x) = \sum_{k=-\infty}^{+\infty} c_k e^{ikx}$ its Fourier coefficients satisfy $|c_k| \leq Ce^{-|k|h}$.

Proposition 3. The following upper bound on the remainder of the Fourier series holds:

$$|R_n| = \left| f(x) - \sum_{k=-n}^n c_k e^{ikx} \right| \leq 2Ce^{-nh} \frac{e^{-h}}{1 - e^{-h}}, \quad (3)$$

where $\sum_{k=-\infty}^{+\infty} c_k e^{ikx}$ is the Fourier series form of $f(x) \in A_h(C)$.

These statements can be easily proved using Definition 1 and their proofs can be found in [7]. If $C \rightarrow \infty, h \rightarrow 0$ then the restriction $A_h(C)$ on $[0, 2\pi]$ is dense in the space of continuous functions defined on $[0, 2\pi]$.

Definition 2. Let us call $A_h^n(C)$ the functional class consisting of all partial sums of $2n + 1$ elements of Fourier series of functions lying in $A_h(C)$, i. e. :

$$f(x) = A_h^n(C) = \left\{ \sum_{k=-n}^n c_k e^{ikx} : \exists c_{-\infty}, \dots, c_{-n-1}, c_{n+1}, \dots, c_{\infty} \in \mathbb{C}, \sum_{k=-\infty}^{\infty} c_k e^{ikx} \in A_h(C) \right\}.$$

In this paper we set the a priori distribution on the experts ($P_0(d\theta)$) as follows: $c_k = Ce^{-|k|h}(\theta_{2k} + i\theta_{2k+1})$, where θ_k are Gaussian independently distributed random variables. We will use predictions generated by this distribution, but some of them may be outside $A_h^n(C)$ because their Fourier coefficients can exceed $Ce^{-|k|h}$ as a random variable distributed as $N(0, 1)$ has its values in $(-\infty, \infty)$. It will not interfere with our result as we prove will prove it now for a wider class of experts and thus our algorithm is not much worse than any expert from $A_h^n(C)$ (or $A_h(C)$ as it will be proved later) as it is a subset of the set for which the result also holds.

Lemma 1. Let the set of experts Θ be $A_h^n(C)$, the loss function be quadratic, and $P_0(d\theta)$ be such that $\frac{c_k}{Ce^{-|k|h}}$ are distributed as $\theta_{2k} + i\theta_{2k+1}$ for $k = -n, \dots, n$, where θ_k are Gaussian independently distributed random variables. Then we have:

$$g_T(\omega) = \log_{\beta} B \int_{\Theta} \beta(\sum_{t=1}^{T-1} (\omega_t - \vec{V}'_t \vec{\theta})^2 + (\omega - \vec{V}'_T \vec{\theta})^2) - \ln \beta \sum_{k=-2n}^{2n+1} \frac{\theta_k^2}{2} d\vec{\theta},$$

where

$$\mathbb{C}^{4n+2} \ni \vec{V}_j = (v_{-2n,j}, v_{-2n+1,j}, \dots, v_{2n,j}, v_{2n+1,j})',$$

$$v_{k,j} = i^{\frac{1-(-1)^k}{2}} C e^{-|\lfloor \frac{k}{2} \rfloor| h + i \lfloor \frac{k}{2} \rfloor x_j},$$

$[x]$ is the integer part of x , $\vec{\gamma} \in \mathbb{R}^{4n+2}$ is the vector containing γ_k , $-2n \leq k \leq 2n + 1$, and $\mathbb{R} \ni B > 0$ is independent of ω .

Proof.

$$g_T(\omega) = \log_{\beta} \int_{\Theta} \beta^{\lambda(\omega, \xi_T(\theta))} P_{T-1}^*(d\theta) =$$

$$\log_{\beta} \int_{\Theta} \beta^{(\omega - \sum_{k=-n}^n c_k(\theta) e^{ikx_T})^2} P_{T-1}^*(d\theta),$$

where $c_k(\theta)$, $-n \leq k \leq n$, are the coefficients of the Fourier series of the prediction, corresponding to θ . Let us calculate $P_{T-1}(d\theta)$ using formula (1):

$$\begin{aligned} P_{T-1}(d\theta) &= \beta^{\lambda(\omega_{T-1}, \xi_{T-1}(\theta))} P_{T-2}(d\theta) = \beta^{(\omega_{T-1} - \sum_{k=-n}^n c_k e^{ikx_{T-1}})^2} P_{T-2}(d\theta) \\ &= \beta^{(\omega_{T-1} - \sum_{k=-n}^n c_k e^{ikx_{T-1}})^2} (\beta^{(\omega_{T-2} - \sum_{k=-n}^n c_k e^{ikx_{T-2}})^2} P_{T-3}(d\theta)) \\ &= \beta^{(\omega_{T-1} - \sum_{k=-n}^n c_k e^{ikx_{T-1}})^2 + (\omega_{T-2} - \sum_{k=-n}^n c_k e^{ikx_{T-2}})^2} P_{T-3}(d\theta) \\ &= \dots = \beta^{\sum_{j=1}^{T-1} (\omega_j - \sum_{k=-n}^n c_k e^{ikx_j})^2} P_0(d\theta). \end{aligned}$$

Note that $P_{T-1}^*(d\theta)$ differs from $P_{T-1}(d\theta)$ only on a multiplicative constant, which is the same for all $\omega \in \Omega$; thus we can include it in constant B from the statement of the theorem. From the last formula we have:

$$\begin{aligned} g_T(\omega) &= \log_\beta \int_{\Theta} \beta^{(\omega - \sum_{k=-n}^n c_k e^{ikx_T})^2} P_{T-1}^*(d\theta) = \\ &= \log_\beta (B \int_{\Theta} \beta^{(\omega - \sum_{k=-n}^n c_k e^{ikx_T})^2} \beta^{\sum_{j=1}^{T-1} (\omega_j - \sum_{k=-n}^n c_k e^{ikx_j})^2} P_0(d\theta)). \end{aligned}$$

Recall that the definition of $P_0(d\theta)$ means that $c_k = C e^{-|k|h} (\theta_{2k} + i\theta_{2k+1})$, where θ_k are Gaussian independently distributed random variables and $\sum_{k=-n}^n c_k e^{ikx_j} = \vec{V}_j' \vec{\theta}$, where

$$\begin{aligned} \mathbb{C}^{4n+2} \ni \vec{V}_j &= (C e^{-nh - inx_j}, iC e^{-nh - inx_j}, C e^{-(n-1)h - i(n-1)x_j}, \\ &= iC e^{-(n-1)h - i(n-1)x_j}, \dots, C e^{-(n-1)h + i(n-1)x_j}, iC e^{-(n-1)h + i(n-1)x_j}, \\ &= C e^{-nh + inx_j}, iC e^{-nh + inx_j})' \\ \text{and } \vec{\theta} &= (\theta_{-2n}, \theta_{-(2n-1)}, \dots, \theta_{2n}, \theta_{2n+1})'. \end{aligned}$$

$$\begin{aligned} g_T(\omega) &= \log_\beta \int_{\Theta} \beta^{(\omega - \sum_{k=-n}^n c_k e^{ikx_T})^2 + \sum_{j=1}^{T-1} (\omega_j - \sum_{k=-n}^n c_k e^{ikx_j})^2} P_0(d\theta) + \\ &= \log_\beta B = \log_\beta \int_{\Theta} \beta^{\sum_{j=1}^{T-1} (\omega_j - \vec{V}_j' \vec{\theta})^2 + (\omega - \vec{V}_T' \vec{\theta})^2} e^{\sum_{k=-2n}^{2n+1} \frac{\theta_k^2}{2}} d\vec{\theta} + \log_\beta B. \quad \square \end{aligned}$$

Proposition 4. When we apply AA to $A_h^n(C)$, γ_T from (2) can be written down in the following form:

$$\gamma_T = \frac{1}{2} \left(\sum_{j=1}^{T-1} 2\omega_j \vec{V}_j' \right)' \left(\sum_{j=1}^T \vec{V}_j \vec{V}_j' + \frac{1}{2} \right)^{-1} (\vec{V}_T'),$$

where $\vec{V}_j$ are the same as in previous lemma.

Proof. Using (2) we have:

$$\begin{aligned}
 \gamma_T &= \frac{1}{4Y} \log_{\beta} \frac{\beta_{gT}(-Y)}{\beta_{gT}(Y)} = \\
 &= \frac{1}{4Y} \log_{\beta} \frac{\int e^{-\eta(\sum_{j=1}^{T-1} (\omega_j - \bar{V}'_j \bar{\gamma})^2 + (Y + \bar{V}'_T \bar{\gamma})^2) - \sum_{k=-2n}^{2n+1} \frac{\gamma_k^2}{2}} d\bar{\gamma}}{\int e^{-\eta(\sum_{j=1}^{T-1} (\omega_j - \bar{V}'_j \bar{\gamma})^2 + (Y - \bar{V}'_T \bar{\gamma})^2) - \sum_{k=-2n}^{2n+1} \frac{\gamma_k^2}{2}} d\bar{\gamma}} = \\
 &= \frac{1}{4Y} \log_{\beta} \frac{\int e^{-\eta(\sum_{j=1}^{T-1} (\omega_j^2 - 2\omega_j \bar{V}'_j \bar{\gamma} + \bar{\gamma}' \bar{V}_j \bar{V}'_j \bar{\gamma}) + (Y^2 + 2Y \bar{V}'_T \bar{\gamma} + \bar{\gamma}' \bar{V}_T \bar{V}'_T \bar{\gamma})) - \frac{1}{2} \bar{\gamma}' \bar{\gamma}} d\bar{\gamma}}{\int e^{-\eta(\sum_{j=1}^{T-1} (\omega_j^2 - 2\omega_j \bar{V}'_j \bar{\gamma} + \bar{\gamma}' \bar{V}_j \bar{V}'_j \bar{\gamma}) + (Y^2 - 2Y \bar{V}'_T \bar{\gamma} + \bar{\gamma}' \bar{V}_T \bar{V}'_T \bar{\gamma})) - \frac{1}{2} \bar{\gamma}' \bar{\gamma}} d\bar{\gamma}} = \\
 &= \frac{1}{4Y} \log_{\beta} \frac{\int e^{-(\bar{\gamma}'(\eta(\sum_{j=1}^{T-1} \bar{V}_j \bar{V}'_j + \bar{V}_T \bar{V}'_T) + \frac{1}{2}) \bar{\gamma}) - \eta(\sum_{j=1}^{T-1} -2\omega_j \bar{V}'_j + 2Y \bar{V}'_T) \bar{\gamma}} d\bar{\gamma}}{\int e^{-(\bar{\gamma}'(\eta(\sum_{j=1}^{T-1} \bar{V}_j \bar{V}'_j + \bar{V}_T \bar{V}'_T) + \frac{1}{2}) \bar{\gamma}) - \eta(\sum_{j=1}^{T-1} -2\omega_j \bar{V}'_j - 2Y \bar{V}'_T) \bar{\gamma}} d\bar{\gamma}} = \\
 &= \frac{1}{4Y} F \left(\left(\sum_{j=1}^{T-1} \bar{V}_j \bar{V}'_j + \frac{1}{2} \right), - \sum_{j=1}^{T-1} 2\omega_j \bar{V}'_j, 2Y \bar{V}'_T \right),
 \end{aligned}$$

where

$$\begin{aligned}
 F(A, \vec{b}, \vec{c}) &= \\
 &= \inf_{\bar{\gamma} \in \mathbb{R}^{4n+2}} \left(\bar{\gamma}' A \bar{\gamma} + \vec{b}' \bar{\gamma} + \vec{c}' \bar{\gamma} \right) - \inf_{\bar{\gamma} \in \mathbb{R}^{4n+2}} \left(\bar{\gamma}' A \bar{\gamma} + \vec{b}' \bar{\gamma} - \vec{c}' \bar{\gamma} \right).
 \end{aligned}$$

It was found in [9] that the function F can be transformed into

$$\begin{aligned}
 F(A, \vec{b}, \vec{c}) &= \inf_{\bar{\gamma} \in \mathbb{R}^{4n+2}} \left(\bar{\gamma}' A \bar{\gamma} + \vec{b}' \bar{\gamma} + \vec{c}' \bar{\gamma} \right) - \\
 &= \inf_{\bar{\gamma} \in \mathbb{R}^{4n+2}} \left(\bar{\gamma}' A \bar{\gamma} + \vec{b}' \bar{\gamma} - \vec{c}' \bar{\gamma} \right) = -b' A^{-1} c.
 \end{aligned}$$

Thus, finally we have

$$\gamma_T = -\frac{1}{4Y} \left(- \sum_{j=1}^{T-1} 2\omega_j \bar{V}'_j \right)' \left(\sum_{j=1}^{T-1} \bar{V}_j \bar{V}'_j + \frac{1}{2} \right)^{-1} (2Y \bar{V}'_T). \quad \square$$

From now on we would like to show explicitly that $\vec{A}_j$ and γ_t depend on n , so we will write $\vec{A}_j^n$ and γ_t^n .

4 Upper Bound

In this section we prove an upper bound on the regret term of the algorithm described above. Let us denote AAA as the Aggregating Algorithm applied to $A_h(C)$.

Algorithm 1. AA for $A_h(C)$ **Require:** Parameters C, h **for** $t = 2, 3, \dots$ **do** get Nature's signal x_t **if** $t < 6$ OR $\lceil \log_2 \log_2(t-1) \rceil \neq \lceil \log_2 \log_2 t \rceil$ **then**

$$n := \left\lceil \frac{(\lceil \log_2 t \rceil + 1) \ln 2}{h} \right\rceil + 1$$

for $j = 1, \dots, t$ **do**

$$\begin{aligned} \vec{V}_j^n &:= (v_{-2n,j}, v_{-2n+1,j}, \dots, v_{2n,j}, v_{2n+1,j})' \in \mathbb{C}^{4n+2}, v_{l,j} \\ &= i^{\frac{1-(-1)^l}{2}} C e^{-\lceil \frac{l}{2} \rceil h + i \lceil \frac{l}{2} \rceil x_j} \end{aligned}$$

end for **else**

$$\begin{aligned} \vec{V}_t^n &:= (v_{-2n,t}, v_{-2n+1,t}, \dots, v_{2n,t}, v_{2n+1,t})' \in \mathbb{C}^{4n+2}, v_{l,t} \\ &= i^{\frac{1-(-1)^l}{2}} C e^{-\lceil \frac{l}{2} \rceil h + i \lceil \frac{l}{2} \rceil x_t} \end{aligned}$$

end if

$$\gamma_t^n := \frac{1}{2} \left(\sum_{j=1}^{t-1} 2\omega_j (\vec{V}_j^n)' \right)' \cdot \left(\sum_{j=1}^t \vec{V}_j^n (\vec{V}_j^n)' + \frac{1}{2} \right)^{-1} \left((\vec{V}_t^n)' \right)$$

 give γ_t^n as the final prediction get Nature's output ω_t **end for**

Theorem 1. Let $L_T(\text{AAA})$ be the total loss of Algorithm 1 over T steps. For $T > 3$ the following inequality holds:

$$L_T(\text{AAA}) - \inf_{f \in A_h(C)} (L_T(f)) \leq$$

$$\frac{1024R^2}{3h} \log_2^2 T + \log_2 T \left(\frac{34R^2}{h} + 73R^2 \right) + 50R^2 + \frac{4R}{h},$$

where $R := \max\{1, C\}$.

Proof. Let $L_T(\text{AAA}(T))$ be the total loss of AA algorithm which “knows” the number of steps T over all of them. From Proposition 1 we have:

$$L_T(\text{AAA}(T)) \leq \inf_{\theta \in A_h^n(C)} (L_T(\theta) + a\|\theta\|^2) + NY^2 \ln \left(\frac{TX^2}{a} + 1 \right),$$

where N is the dimension of Θ . On each step of the prediction protocol the difference between the optimal prediction in $A_h(C)$ and some prediction in $A_h^n(C)$ is no more than $2Ce^{-nh} \frac{e^{-h}}{1-e^{-h}}$, according to (3). Hence, the total loss of this algorithm can be estimated as

$$\begin{aligned} L_T(\text{AAA}(T)) &\leq \inf_{\theta \in A_h^n(C)} (L_T(\theta) + a\|\theta\|^2) + \\ &\quad (4n+2)Y^2 \ln \left(\frac{TX^2}{a} + 1 \right) + 2TCe^{-nh} \frac{e^{-h}}{1-e^{-h}} \quad (4) \end{aligned}$$

and this inequality holds for any $n > 0$. Now we can adjust this formula to our particular case: $a = 1$, because $\|\theta\|^2$ is the norm of coefficients of a function in its Fourier series, $X = \pi$, because the functions from our class are π -periodic and, finally, $Y = C$ because they are bounded by constant C everywhere.

Now we will try to get rid of n in this formula, looking for the best value of n . Thus, we can consider formula (4) as a minimization problem and we are looking for the minimum of

$$(4n + 2)C^2 \ln(T\pi^2 + 1) + 2TCe^{-nh} \frac{e^{-h}}{1 - e^{-h}}, \text{ varying } n:$$

$$\left((4n + 2)C^2 \ln(T\pi^2 + 1) + 2TCe^{-nh} \frac{e^{-h}}{1 - e^{-h}} \right)'_n =$$

$$4C^2 \ln(T\pi^2 + 1) - 2hTCe^{-nh} \frac{e^{-h}}{1 - e^{-h}} \Rightarrow e^{-nh} = \frac{4C \ln(T\pi^2 + 1)}{2hT} \frac{1 - e^{-h}}{e^{-h}}$$

and

$$n = \frac{1}{h} \left(\ln T - \ln \ln(T\pi^2 + 1) - \ln 2 + \ln h - h - \ln C - \ln(1 - e^{-h}) \right).$$

However, looking the best value of n is not always the best way of looking for a solution. For instance, if we take $n = \left\lceil \frac{\ln T}{h} \right\rceil + 1$ the formula of the upper bound will be more understandable. Thus we have

$$L_T(\text{AAA}(T)) \leq \inf_{\theta} (L_T(\theta)) + (4n + 2)C^2 \ln(T\pi^2 + 1) + 2TCe^{-nh} \frac{e^{-h}}{1 - e^{-h}} \leq$$

$$\inf_{\theta} (L_T(\theta)) + (4n + 2)C^2 \ln(T\pi^2 + 1) + D \leq$$

$$\inf_{\theta} (L_T(\theta)) + \frac{4 \ln T + 6h}{h} C^2 \ln(T\pi^2 + 1) + D,$$

where D is an additive constant equal to $C + 2C \frac{e^{-h}}{1 - e^{-h}}$.

Now we will consider the step of the algorithm where we change the parameter n . The algorithm changes the parameter when the number of the current step is 2, 3, 4, 5, 16, 256, 65536, ... Let us calculate the following sum:

$$\sum_{i=0}^{\lceil \log_2 \log_2 T \rceil + 1} \left| L_{\min\{2^{2^i}, T\}}(\text{AAA}(2^{2^i})) - \inf_{f \in A_h(C)} (L_{\min\{2^{2^i}, T\}}(f)) \right| \leq$$

$$\sum_{i=0}^{\lceil \log_2 \log_2 T \rceil + 1} \left(\frac{2^i 4 \ln 2 + 6h}{h} C^2 \ln(2^{2^i} \pi^2 + 1) + D \right).$$

In order to make our inequality simpler, we will use some rough estimations. This will make our result less precise but will give an easier understanding of the upper bound. Now we need some inequalities to make the result simpler:

$$\ln(T\pi^2 + 1) < \ln(T\pi^2) + 1, \quad C < R^2, \quad \frac{e^{-h}}{1 - e^{-h}} < \frac{1}{h}, \quad 1 < \ln T.$$

Substituting the corresponding parts of the inequality, we get:

$$\begin{aligned}
& \sum_{i=0}^{\lceil \log_2 \log_2 T \rceil + 1} \left| L_{\min\{2^{2^i}, T\}} \left(\text{AAA} \left(2^{2^i} \right) \right) - \inf_{f \in A_h(C)} \left(L_{\min\{2^{2^i}, T\}} (f) \right) \right| \leq \\
& \sum_{i=0}^{\lceil \log_2 \log_2 T \rceil + 1} \left(\left(\frac{4C^2 2^i}{h} + 6C^2 \right) (4 + 2^i) + C + \frac{2C}{h} \right) \leq \\
& \sum_{i=0}^{\lceil \log_2 \log_2 T \rceil + 1} 2^{2^i} \frac{4C^2}{h} + \sum_{i=0}^{\lceil \log_2 \log_2 T \rceil + 1} 2^i \left(\frac{4C^2}{h} + 6C^2 \right) + \\
& (\log_2 \log_2 T + 2) \left(24C^2 + C + \frac{2C}{h} \right) \leq \\
& \frac{1024C^2}{3h} \log_2^2 T - 1 + \log_2 T \left(\frac{32C^2}{h} + 48C^2 \right) - 1 + \\
& (\log_2 T + 2) \left(24C^2 + C + \frac{2C}{h} \right) \leq \\
& \frac{1024R^2}{3h} \log_2^2 T + \log_2 T \left(\frac{34R^2}{h} + 73R^2 \right) + 50R^2 + \frac{4R}{h}
\end{aligned}$$

since

$$\sum_{i=0}^T 2^i = 2^{T+1} - 1.$$

Now, to end the proof of the theorem, it is enough to show that

$$\begin{aligned}
& L_T(\text{AAA}) - \inf_{f \in A_h(C)} (L_T(f)) \leq \\
& \sum_{i=0}^{\lceil \log_2 \log_2 T \rceil + 1} \left| L_{\min\{2^{2^i}, T\}} \left(\text{AAA} \left(2^{2^i} \right) \right) - \inf_{f \in A_h(C)} \left(L_{\min\{2^{2^i}, T\}} (f) \right) \right|. \square
\end{aligned}$$

This bound is an intermediate estimate between $O(\log T)$ and $O(\sqrt{T})$. The result shows that there are algorithms with other (not found yet) estimates of the upper bound on the regret term.

5 Conclusion

In this paper we developed an algorithm which competes with a class of experts from a specific functional space. This functional space is a wide class of experts and the algorithm gives an idea on how to approach the problem of competing with experts in the form of continuous functions. An upper bound on the regret term of the total loss of the algorithm is proven. The regret term is of an order, different from the orders of the regret terms of the existing algorithms. Thus this

result approaches the problem of finding the correspondence between a class of experts and the upper bound on the regret term of an algorithm competing with these experts.

This paper suggests a number of directions for further research. One of them is related to the algorithm itself. If we could find an efficient way of aggregating experts from $A_h(C)$, then we could mix predictions from $A_h(C)$, $C \rightarrow \infty$, $h \rightarrow 0$ and thus compete with continuous functions. We suggest that in the trade-off richness of a class of experts—low upper bound on the regret term of the algorithm the class of continuous functions is a fairly large class of experts for most applications. Unfortunately, we have not yet found a way to use the described techniques to compete with continuous function.

Another direction for further research concerns the correspondence between a class of experts and the upper bound of the AA competing with this class of experts. Finding more classes of experts with different upper bounds on the regret term leads to a better decision on choosing the class of experts as there exists the trade-off mentioned in the previous paragraph.

To solve the problem of Prediction with Expert Advice (PEA) in applications one usually chooses either the error rate they can tolerate or a class of experts which reflects what kind of dependence can be expected between the signals and the outcomes. To make a decision regarding choosing initial parameters of an algorithm in PEA it is useful to have groups of three (algorithm, set of experts, upper bound on the regret term) with different upper bounds on the regret term. Our result adds one more group to the set of existing results (where the only known group with the upper bound in the form of $O(\log^2 T)$ is the result described in this paper) and approaches solving the problem of competing with continuous functions.

Acknowledgements. This work was supported by EPSRC through grant EP/E000053/1, “Machine Learning for Resource Management in Next-Generation Optical Networks”. The author is grateful to Volodya Vovk, Alexey Chernov and Brian Burford for valuable discussions and to reviewers for comments which led to improvements in the paper.

References

- [1] Auer, P., Cesa-Bianchi, N., Gentile, C.: Adaptive and selfconfident on-line learning algorithms. *Journal of Computer and System Sciences* 64, 48–75 (2002)
- [2] Azoury, K.S., Warmuth, M.K.: Relative Loss Bounds for On-Line Density Estimation with the Exponential Family of Distributions. *Machine Learning*, 211–246 (2001)
- [3] Cesa-Bianchi, N., Long, P.M., Warmuth, M.K.: Worstcase quadratic loss bounds for on-line prediction of linear functions by gradient descent. *IEEE Transactions on Neural Networks* 7, 604–619 (1996)
- [4] Cesa-Bianchi, N., Lugosi, G.: Prediction, learning, and games. Cambridge University Press, Cambridge (2006)

- [5] Freund, Y.: Predicting a binary sequence almost as well as the optimal biased coin. In: Proc. 9th Annu. Conf. on Comput. Learning Theory, pp. 89–98 (1996)
- [6] Hazan, E., Kalai, A., Kale, S., Agarwal, A.: Logarithmic Regret Algorithms for Online Convex Optimization. *Machine Learning* 69(2-3), 169–192 (2007)
- [7] Kolmogorov, A.N., Tikhomirov, V.M.: ϵ -Entropy and ϵ -Capacity of Sets in a Functional Space. *Usp. Mat. Nauk* 14(2), 3–86 (1959); *Russian Mathematical Surveys* 17, 277 (1961) (Translation)
- [8] Vovk, V.: On-line regression competitive with reproducing kernel Hilbert spaces (2008), <http://arxiv.org/abs/cs.LG/0511058>
- [9] Vovk, V.: Competitive On-line Statistics. *International Statistical Review* 69, 213–248 (2001)
- [10] Vovk, V.: Metric entropy in competitive on-line prediction (2006), <http://arxiv.org/abs/cs.LG/0609045>
- [11] Vovk, V., Nouretdinov, I., Takemura, A., Shafer, G.: Defensive forecasting for linear protocols (2005), <http://arxiv.org/abs/cs.LG/0506007>

Smooth Boosting for Margin-Based Ranking

Jun-ichi Moribe, Kohei Hatano, Eiji Takimoto, and Masayuki Takeda

Department of Informatics, Kyushu University
{moribe,hatano,eiji,takeda}@i.kyushu-u.ac.jp

Abstract. We propose a new boosting algorithm for bipartite ranking problems. Our boosting algorithm, called SoftRankBoost, is a modification of RankBoost which maintains only smooth distributions over data. SoftRankBoost provably achieves approximately the maximum soft margin over all pairs of positive and negative examples, which implies high AUC score for future data.

1 Introduction

Learning of ranking has been extensively studied recently [5, 12, 9, 4, 15, 6, 14, 1, 13]. Ranking is quite useful in information retrieval, recommendation tasks, bioinformatics and so on. A basic ranking problem called the bipartite ranking problem is defined over two classes, where the algorithm takes as input a set of positive and negative instances, and outputs a ranking function, which maps any instance to a real number. The goal is to obtain such a ranking function that for most pairs of positive and negative instances, it gives a higher value to the positive instance than to the negative instance.

A standard measure for evaluating the quality of a ranking function is its ROC curve [2, 8]. An ROC curve of a ranking function is obtained by plotting its true positive rate and false positive rate while changing its threshold. More precisely, the ROC curve of a ranking function h is defined as the line graph that passes through the points $(\alpha_\theta, \beta_\theta)$ with threshold θ ranging from $-\infty$ to ∞ , where α_θ is the fraction of the negative instances x such that $h(x) \geq \theta$ (false positives) and β_θ is the fraction of positive instances x such that $h(x) \geq \theta$ (true positives). Thus, any ROC curve is monotonically increasing and contains the points $(0, 0)$ and $(1, 1)$. Intuitively, a ranking function is better if its ROC curve goes through points with larger β -coordinates. So, the goodness of ROC curve is summarized as the area under the curve, which is called the AUC (Area Under the ROC Curve). There are many researches to learn functions achieving higher AUC values (See e.g., [12, 4]).

A major approach to learn good ranking functions is to reduce ranking problems into classification problems over pair of positive and negative instances [12, 9, 4, 15, 14]. More precisely, for the reduction, we consider a new instance space consisting of all pairs $(\mathbf{x}_i, \mathbf{x}'_j)$ of positive instances $\mathbf{x}_i$ and negative instances $\mathbf{x}'_j$, and consider a hypothesis class consisting of functions defined over pairs $(\mathbf{x}_i, \mathbf{x}'_j)$ of the form of $h(\mathbf{x}_i) - h(\mathbf{x}'_j)$ for some ranking functions h . Thus, we have a usual classification problem by considering all pairs $(\mathbf{x}_i, \mathbf{x}'_j)$ to be labeled positive.

Some generalization bounds are known as well [15, 1]. Among such results, Rudin et al. showed that large margin over pairs of positive and negative examples implies high AUC score for future examples under the standard assumption that data is drawn i.i.d. under the underlying distribution. Several researchers apply large margin classifiers such as SVMs over pairs of instance [12, 9, 4]. However, a disadvantage of this pairwise approach is that the sample size might become quadratically larger than the original data.

In this paper, we propose a new boosting algorithm which achieves large margin over pairs of instances while avoiding quadratically large data of pairs.

First of all, we begin with showing a simple observation that RankBoost, a boosting algorithm proposed by Freund et al. which is explicitly designed for ranking, provably maximizes the AUC, given that “weak rankers” are available. This result simplifies the result of Long and Servedio [13] by removing internal randomization in their algorithm.

Second, we further investigate general situations where given pairs of positive and negative instances are not separable by any linear combinations of base functions. In this case, based on a soft margin formulation over pairs, we propose a “smooth” version of RankBoost which we call SoftRankBoost. SoftRankBoost is a successor of “smooth” boosting algorithms, such as MadaBoost [7], SmoothBoost [18], AdaFlat [10], GiniBoost [11], and FilterBoost [3] which are designed to maintain only smooth distributions over data, i.e., distributions that are close to the original distribution (typically the uniform distribution over data).

Under the soft margin boosting framework by Warmuth et al. [19] we show that SoftRankBoost approximately achieves the maximum soft margin over large amount of pairs. More precisely, given p positive and n negative instances, precision parameters δ ($0 < \delta < 1$), $\nu \in \{1, 2, \dots, \min\{p, n\}\}$, SoftRankBoost outputs a linear combination of base functions which has margin $(1 - \delta)\gamma^*$ over at least $1 - \frac{(p+n)\nu}{pn}$, in fractions of all pn pairs, where γ^* is the maximum soft margin defined under the parameter ν .

We note that if we apply the standard soft margin optimization over pairs of instances, the same margin $(1 - \delta)\gamma^*$ is guaranteed for at least $1 - \frac{\nu}{pn}$ fraction of pairs, which is better than our result. However, our algorithm does not expand data quadratically unlike standard methods. Also, our bound is still reasonable unless data is biased, i.e., $p \ll n$ or $p \gg n$.

2 Preliminaries

Let $\mathcal{X}^+$ and $\mathcal{X}^-$ be the sets of positive instances and negative instances, respectively and let $\mathcal{X} = \mathcal{X}^+ \cup \mathcal{X}^-$. Let d be a distribution over $\mathcal{X}$. We say that a distribution d over $\mathcal{X}$ is nontrivial if d has non-zero probability over both positive and negative instances. Given a non-trivial distribution d , we denote d^+ and d^- as the marginal distribution of d over positive and negative instances, respectively. A ranking function h is any function from $\mathcal{X}$ to $[-1, +1]$. The AUC of hypothesis h with respect to a non-trivial distribution d over $\mathcal{X}$ is given as

$$AUC_d(h) = \Pr_{\mathbf{x}_i \sim d, \mathbf{x}'_j \sim d} \{h(\mathbf{x}_i) > h(\mathbf{x}'_j) \mid \mathbf{x}_i \in \mathcal{X}^+, \mathbf{x}'_j \in \mathcal{X}^-\},$$

where each $\mathbf{x}_i$ and $\mathbf{x}'_j$ are drawn independently from d .

Let S be a set of $m(=p+n)$ instances drawn i.i.d. from d , which includes p positive instances and n negative instances, respectively. We denote the subsets of positive and negative instances as $S^+ = \{\mathbf{x}_1, \dots, \mathbf{x}_p\}$ and $S^- = \{\mathbf{x}'_1, \dots, \mathbf{x}'_n\}$, respectively. Given $\rho > 0$, we define

$$\Pr_{S^+, S^-} \{h(\mathbf{x}_i) - h(\mathbf{x}'_j) < \rho\} = \frac{\sum_{i=1}^p \sum_{j=1}^n I(h(\mathbf{x}_i) - h(\mathbf{x}'_j) < \rho)}{pn},$$

where $I(\cdot)$ is the indicator function. Given S , we define

$$AUC_S(h) = \frac{\sum_{i=1}^p \sum_{j=1}^n I(h(\mathbf{x}_i) - h(\mathbf{x}'_j) > 0)}{pn}.$$

The following theorem was shown by Rudin et al.

Theorem 1 (Rudin et al. [15]). Let $\mathcal{F}$ be a set of functions from $\mathcal{X}$ to $\mathbb{R}$. Then, for any $\varepsilon > 0$, $\rho > 0$, for any $h \in \mathcal{F}$, it holds that

$$1 - AUC_d(h) \leq \Pr_{S^+, S^-} \{h(\mathbf{x}_i) - h(\mathbf{x}_j) \leq \rho\} + \varepsilon \quad (1)$$

with probability at least $1 - 2\mathcal{N}(\mathcal{F}, \frac{\varepsilon\rho}{8}) \exp\left\{-\frac{m\varepsilon^2 E^2}{8}\right\}$, where E is the expectation of $I(\mathbf{x}_i \in \mathcal{X}^+, \mathbf{x}'_j \in \mathcal{X}^-)$ when $\mathbf{x}_i$ and $\mathbf{x}'_j$ are drawn independently from d , and $\mathcal{N}(\mathcal{F}, \varepsilon)$ is the covering number of $\mathcal{F}$, which is defined as the minimum number of balls of radius ε needed to cover $\mathcal{F}$ using L_∞ norm.

Here, note that the covering number is smaller if ρ is larger. So, a robust approach to learn a hypothesis with high AUC is to enlarge the margin ρ over the pairs of positive and negative instances.

In this paper, we assume a finite set $\mathcal{H} = \{h_1, h_2, \dots, h_N\}$ of functions from $\mathcal{X}$ to $[-1, +1]$. For convenience, we further assume that if $\mathcal{H}$ contains h , it also contains $g = -h$. Our hypothesis class $\mathcal{F}$ is the set of convex combination of functions in $\mathcal{H}$, that is,

$$\mathcal{F} = \left\{ f \mid f(\mathbf{x}) = \sum_{k=1}^N \alpha_k h_k(\mathbf{x}), \sum_{k=1}^N \alpha_k = 1, \alpha_k \geq 0 \right\}.$$

Now, our goal is to find a function $f \in \mathcal{F}$ which has large margin ρ over pairs of instances in S^+ and S^- .

3 Boosting the AUC

In this section, before investigating large margin classification over linearly inseparable pairs of instances, we first review RankBoost [9]¹ and we show that

¹ More precisely, we review RankBoost.B [9] which is an efficient implementation of RankBoost for bipartite ranking problems.

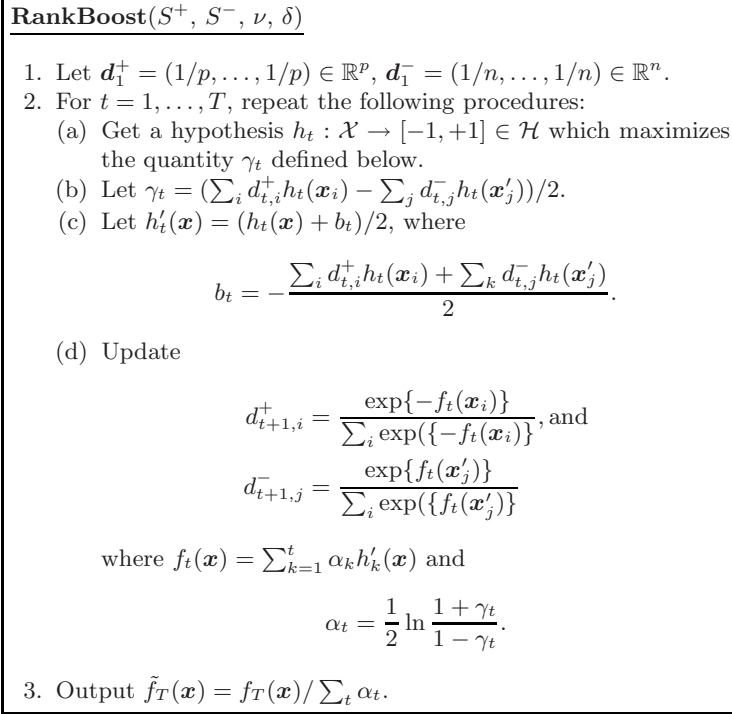


Fig. 1. RankBoost. Note that the bias term b_t is redundant. We add the bias term only to prove the convergence property of RankBoost.

RankBoost provably maximizes the AUC over given training data, provided that a set $\mathcal{H}$ always contains a “weak ranker” with respect to any distribution. More precisely, we employ the following assumption ².

Definition 1 (Weak AUC ranker assumption, Long and Servedio [13]). We assume that for any non-trivial distribution d over S , a set of functions $\mathcal{H}$ contains a function $h : \mathcal{X} \rightarrow [-1, +1]$ such that $AUC_d(h) \geq 1/2 + \gamma$ ($0 < \gamma < 1/2$).

Then we review RankBoost, whose full description is given in Figure 1. Note that, at 2(c) in Figure 1, we introduce the bias term b_t which does not appear in the original RankBoost. Although this modification does not change the behavior of RankBoost, but it gives a simple proof and leads us to our new boosting algorithm which we will explain later.

The following lemma shows a weak AUC ranker can be transformed into a weak learner for distributions in which positive and negative instances are equally likely.

² In the result of Long and Servedio[13], they use a slightly different definition of AUC: $AUC_d(h) = \Pr_{\mathbf{x}_i \sim d^+, \mathbf{x}_j \sim d^-} \{h(\mathbf{x}_i) > h(\mathbf{x}_j)\} + \frac{1}{2} \Pr_{\mathbf{x}_i \sim d^+, \mathbf{x}_j \sim d^-} \{h(\mathbf{x}_i) = h(\mathbf{x}_j)\}$. Our following proof also holds under their definition.

Lemma 1. If there is a function $\tilde{h} : \mathcal{X} \rightarrow R$ such that $AUC_d(h) \geq 1/2 + \gamma$ for a d_t over S defined as

$$d_t(\mathbf{x}) = \begin{cases} \frac{d_t^+(\mathbf{x})}{2} & \mathbf{x} \in S^+ \\ \frac{d_t^-(\mathbf{x})}{2} & \mathbf{x} \in S^-, \end{cases} \quad (2)$$

then, there exists a function $h : \mathcal{X} \rightarrow \{-1, +1\}$ such that

$$\frac{\sum_i d_{t,i}^+ h(\mathbf{x}_i) - \sum_i d_{t,j}^- h(\mathbf{x}_j)}{2} \geq \gamma.$$

Proof. The definition of a ROC curve of $\tilde{h}$ implies that, there exists a threshold θ such that

$$\Pr_{d_t^+}\{\tilde{h}(\mathbf{x}) \geq \theta\} \geq \Pr_{d_t^-}\{\tilde{h}(\mathbf{x}) \geq \theta\} + \gamma.$$

Now consider the binary function $h(\mathbf{x}) = \text{sign}(\tilde{h}(\mathbf{x}) - \theta)$. Let tp , fn , fp , tn be probabilities of true positive, false negative, false positive, and true negative instances with respect to h and d_t , respectively. Then, using this notation, the last inequality is written as

$$\frac{tp}{tp + fn} \geq \frac{fp}{fp + tn} + \gamma.$$

Note that $tp + fn = 1/2$ and $fp + tn = 1/2$. So, by rearranging, we have

$$2tp \geq 2fp + \gamma.$$

Since $tp + fn = fp + tn = 1/2$ and $tp - fp = tn - fn$, we get

$$\frac{\sum_i d_{t,i}^+ h(\mathbf{x}_i) - \sum_i d_{t,j}^- h(\mathbf{x}_j)}{2} \geq \gamma. \quad \square$$

Then we prove a simple lemma which is a core of our analysis.

Lemma 2. At each iteration t ,

$$\sum_i d_{t,i}^+ h'_t(\mathbf{x}_i) = - \sum_j d_{t,j}^- h'_t(\mathbf{x}'_j) = \frac{\gamma_t}{2}.$$

Proof. Since b_t varies in $[-1, +1]$, the range of h' is $[-1, +1]$ as well. Then, we have

$$\begin{aligned} \sum_i d_{t,i}^+ h'_t(\mathbf{x}_i) &= \frac{1}{2} \sum_i d_{t,i}^+ h_t(\mathbf{x}_i) + \frac{b_t}{2} \\ &= \frac{\sum_i d_{t,i}^+ h_t(\mathbf{x}_i) - \sum_j d_{t,j}^- h_t(\mathbf{x}'_j)}{4} = \frac{1}{2} \gamma_t, \end{aligned}$$

as claimed. Similarly, we obtain $\sum_j -d_{t,j}^- h'_t(\mathbf{x}'_j) = \gamma_t/2$. and we complete the proof. $\square$

Now we prove the AUC-maximizing property of RankBoost.

Theorem 2. Under the weak ranker assumption on $\mathcal{H}$, RankBoost output a function whose AUC is at least $1 - \varepsilon$ in $O(\frac{\ln(1/\varepsilon)}{\gamma^2})$ iterations.

Proof.

$$\begin{aligned} AUC_S(\text{sign}(f_T)) &= \Pr_{S^+, S^-} \{ \text{sign}(f_T(\mathbf{x}_i)) > \text{sign}(f_T(\mathbf{x}_j)) \} \\ &= \Pr_{S^+} \{ \text{sign}(f_T(\mathbf{x}_i)) = 1 \} \Pr_{S^-} \{ \text{sign}(f_T(\mathbf{x}_j)) = 0 \} \\ &\geq 1 - \Pr_{S^+} \{ f_T(\mathbf{x}_i) < 0 \} - \Pr_{S^-} \{ f_T(\mathbf{x}_j) > 0 \}. \end{aligned}$$

Now, according to Lemma 2 and using an analysis of AdaBoost [16], both terms $\Pr_{S^+} \{ f_T(\mathbf{x}_i) < 0 \}$ and $\Pr_{S^-} \{ f_T(\mathbf{x}_j) > 0 \}$ are bounded respectively by $e^{-2cT\gamma^2}$ for some constant $c > 0$. So, $AUC(\text{sign}(f_T))$ is at least $1 - \varepsilon$ after $T = O(\frac{\ln(1/\varepsilon)}{\gamma^2})$ iterations. $\square$

4 Boosting the Margin over Pairs of Instances

In this section, we propose a new boosting algorithm which achieve large margin over pair of instances.

More formally, based on the recent result of Warmuth et al. [19], we formulate our problem as optimizing the soft margin over pairs as follows: For positive and negative sets of instances S^+ and S^- , the set $\mathcal{H}$ of functions, and any fixed $\nu \in \{1, \dots, pn\}$, we consider the following problem:

$$\begin{aligned} \rho^* &= \max_{\boldsymbol{\alpha}, \rho, \boldsymbol{\xi}} \rho - \frac{1}{\nu} \sum_{i=1}^p \sum_{j=1}^n \xi_{ij} \\ \text{sub.to} \quad &\sum_k \alpha_k (h_k(\mathbf{x}_i) - h_k(\mathbf{x}'_j)) / 2 \geq \rho - \xi_{ij} \quad (i = 1, \dots, p, j = 1, \dots, n) \\ &\sum_{k=1}^N \alpha_k = 1, \quad \alpha_k \geq 0 \quad (k = 1, \dots, N) \\ &\xi_{ij} \geq 0 \quad (i = 1, \dots, p, j = 1, \dots, n) \end{aligned}$$

Then, using Lagrangian multipliers, the dual problem is given as

$$\begin{aligned} \gamma^* &= \min_{d, \gamma} \gamma \\ \text{sub.to} \quad &\sum_{i,j} d_{ij} (h_k(\mathbf{x}_i) - h_k(\mathbf{x}'_j)) / 2 \leq \gamma \quad (k = 1, \dots, N) \end{aligned}$$

$$0 \leq d_{ij} \leq \frac{1}{\nu} \quad (i = 1, \dots, p, j = 1, \dots, n)$$

$$\sum_{i=1}^p \sum_{j=1}^n d_{ij} = 1.$$

By duality, we have $\rho^* = \gamma^*$. Further, it can be shown that (see, e.g., [17, 19]), the optimal solution guarantees the number of pairs $(\mathbf{x}_i, \mathbf{x}'_j)$ for which $\sum_k \alpha_k (h_k(\mathbf{x}_i) - h_k(\mathbf{x}'_j))/2 \leq \rho^*$ is at most ν .

For a fixed $\nu \geq 1$, we assume that the optimal solution exists. From now on, based on this assumption, we construct a new boosting algorithm. We note that our new algorithm does not solve the soft margin optimization problem as described above. Instead, our algorithm is guaranteed to produce a final classifier having margin $(1 - \delta)\gamma^*$ over a large amount of pairs of positive and negative instances, where δ ($0 < \delta < 1$) is an input parameter.

4.1 Our New Boosting Algorithm

Our boosting algorithm SoftRankBoost uses the following “potential” function (which was first used in MadaBoost [7]).

$$L(x) = \begin{cases} e^x, & \text{if } x \leq 0 \\ x + 1 & \text{if } x > 0. \end{cases}$$

The derivative of $L(x)$, denoted as $\ell(x)$ is given as

$$\ell(x) = \begin{cases} e^x, & \text{if } x \leq 0 \\ 1 & \text{if } x > 0. \end{cases}$$

A benefit of using this potential function is that during the course of boosting, weights over instances are always bounded. This property plays an important role in smooth boosting algorithms [7, 18, 10, 11, 3].

The description of SoftRankBoost is given in Figure 2. For simplicity of the description, for any $\gamma > 0$, and a function $f : \mathcal{X} \rightarrow \mathbb{R}$ we denote

$$\text{err}_{\gamma}^{+}(f) = |\{\mathbf{x}_i \in S^{+} \mid f(\mathbf{x}_i) < \gamma\}|, \text{ and}$$

$$\text{err}_{\gamma}^{-}(f) = |\{\mathbf{x}'_j \in S^{-} \mid -f(\mathbf{x}'_j) < \gamma\}|,$$

respectively.

Then we prove the following lemma.

Lemma 3. If $\min\{\text{err}_{\tilde{\gamma}_t}^{+}(\tilde{f}_t), \text{err}_{\tilde{\gamma}_t}^{-}(\tilde{f}_t)\} > \nu$, then $\max_{i,j} d_{t,i}^{+} d_{t,j}^{-} \leq 1/\nu$.

Proof. Without loss of generality, we assume that $\text{err}_{\tilde{\gamma}_t}^{+}(\tilde{f}_t) < \text{err}_{\tilde{\gamma}_t}^{-}(\tilde{f}_t)$. Then we have

$$\text{err}_{\tilde{\gamma}_t}^{-}(\tilde{f}_t) \geq \nu.$$

SoftRankBoost(S^+, S^-, ν, δ)

1. Let $\mathbf{d}_1^+ = (1/p, \dots, 1/p) \in \mathbb{R}^p$, $\mathbf{d}_1^- = (1/n, \dots, 1/n) \in \mathbb{R}^n$.
2. For $t = 1, \dots$, repeat the following procedures:
 - (a) Get a hypothesis $h_t : \mathcal{X} \rightarrow [-1, +1] \in \mathcal{H}$ which maximizes the quantity γ_t defined below.
 - (b) Let $\gamma_t = (\sum_i d_{t,i}^+ h_t(\mathbf{x}_i) - \sum_j d_{t,j}^- h_t(\mathbf{x}'_j))/2$.
 - (c) Let $h'_t(\mathbf{x}) = (h_t(\mathbf{x}) + b_t)/2$, where

$$b_t = -\frac{\sum_i d_{t,i}^+ h_t(\mathbf{x}_i) + \sum_k d_{t,j}^- h_t(\mathbf{x}'_j)}{2}.$$

Let $\hat{\gamma}_t = (1 - \delta) \min_t \gamma_t$.

- (d) Update

$$d_{t+1,i}^+ = \frac{\ell(-f_t(\mathbf{x}_i) + \hat{\gamma}_t \sum_{k=1}^t \alpha_k)}{\sum_i \ell(-f_t(\mathbf{x}_i) + \hat{\gamma}_t \sum_{k=1}^t \alpha_k)}, \text{ and}$$

$$d_{t+1,j}^- = \frac{\ell(f_t(\mathbf{x}'_j) + \hat{\gamma}_t \sum_{k=1}^t \alpha_k)}{\sum_i \ell(f_t(\mathbf{x}'_j) + \hat{\gamma}_t \sum_{k=1}^t \alpha_k)},$$

where $f_t(\mathbf{x}) = \sum_{k=1}^t \alpha_k h'_k(\mathbf{x})$ and

$$\alpha_t = \frac{\gamma_t - \hat{\gamma}_t}{2(1 + \hat{\gamma}_t^2)}.$$

- (e) If $\max\{\text{err}_{\hat{\gamma}_t}^+(\tilde{f}_t), \text{err}_{\hat{\gamma}_t}^-(\tilde{f}_t)\} \leq \nu$, then $T = t$ and break.
3. Output $\tilde{f}_T(\mathbf{x}) = f_T(\mathbf{x}) / \sum_t \alpha_t$.

Fig. 2. SoftRankBoost

Therefore we obtain

$$\begin{aligned} d_{t,j}^- &= \frac{\ell(f_t(\mathbf{x}'_j) + \hat{\gamma}_t \sum_{k=1}^t \alpha_k)}{\sum_i \ell(f_t(\mathbf{x}_i) + (\hat{\gamma}_t \sum_{k=1}^t \alpha_k))} \\ &\leq \frac{\ell(f_t(\mathbf{x}'_j) + \hat{\gamma}_t \sum_{k=1}^t \alpha_k)}{\text{err}_{\hat{\gamma}_t}^-(\tilde{f}_t)} \\ &\leq \frac{\ell(+f_t(\mathbf{x}'_j) + \hat{\gamma}_t \sum_{k=1}^t \alpha_k)}{\nu} \leq \frac{1}{\nu}. \end{aligned}$$

Since $d_{t,i}^+ \leq 1$, we get $d_{t,i}^+ d_{t,j}^- \leq \frac{1}{\nu}$. □

Lemma 3 guarantees the smoothness of distributions d_t^+ and d_t^- during iterations of the algorithm. Then we use the following Lemma later.

Lemma 4 (Hatano[11]). For any $a \in \mathbb{R}$ and any $x \in [-1, +1]$, it holds that

$$L(x + a) \leq L(a) + \ell(a)x + \ell(a)x^2.$$

Now we analyze the decrease of the potential function L per iteration.

Lemma 5. For any iteration t in which $\text{err}_{\hat{\gamma}_t}^+(\tilde{f}_t) \geq \nu$,

$$\frac{1}{p} \sum_i L(-f_{t+1}(\mathbf{x}_i) + \sum_k^{t+1} \hat{\gamma}_k \alpha_k) - \frac{1}{p} \sum_i L(-f_t(\mathbf{x}_i) + \sum_k^t \hat{\gamma}_k \alpha_k) \leq -\frac{\nu \delta^2 \hat{\gamma}_t^2}{8p}.$$

Also, for any iteration t in which $\text{err}_{\hat{\gamma}_t}^-(\tilde{f}_t) \geq \nu$,

$$\frac{1}{n} \sum_j L(f_{t+1}(\mathbf{x}'_j) + \sum_k^{t+1} \hat{\gamma}_k \alpha_k) - \frac{1}{n} \sum_j L(f_t(\mathbf{x}'_j) + \sum_k^t \hat{\gamma}_k \alpha_k) \leq -\frac{\nu \delta^2 \hat{\gamma}_t^2}{8n}.$$

Proof. We only prove the first inequality. The second inequality can be proved symmetrically. Note that

$$|\alpha_t(-y_i h'_t(\mathbf{x}_i) + \hat{\gamma}_t)| \leq \frac{\gamma_t - \hat{\gamma}_t}{2(1 + \hat{\gamma}_t^2)}(1 + \hat{\gamma}_t) \leq \frac{1}{2} \cdot 2 \leq 1.$$

So, by applying Lemma 4,

$$\begin{aligned} & \frac{1}{p} \sum_i L(-f_{t+1}(\mathbf{x}_i) + \sum_k^{t+1} \hat{\gamma}_k \alpha_k) - \frac{1}{p} \sum_i L(-f_t(\mathbf{x}_i) + \sum_k^t \hat{\gamma}_k \alpha_k) \\ & \leq \frac{1}{p} \sum_i \ell(-f_t(\mathbf{x}_i) + \sum_k \hat{\gamma}_k \alpha_k) \cdot \{\alpha_t(-\gamma_t + \hat{\gamma}_t) + \alpha_t^2(1 - 2\gamma_t \hat{\gamma}_t + \hat{\gamma}_t^2)\} \\ & \leq \frac{1}{p} \sum_i \ell(-f_t(\mathbf{x}_i) + \sum_k \hat{\gamma}_k \alpha_k) \cdot \{\alpha_t(-\gamma_t + \hat{\gamma}_t) + \alpha_t^2(1 + \hat{\gamma}_t^2)\} \\ & \stackrel{\text{def}}{=} \Delta(\alpha_t) \end{aligned}$$

Letting $\alpha_t = \frac{(\gamma_t - \hat{\gamma}_t)}{2(1 + \hat{\gamma}_t^2)}$, we obtain

$$\begin{aligned} \Delta_t(\alpha_t) &= -\frac{1}{4} \frac{(\gamma_t - \gamma_t)^2}{1 + \hat{\gamma}_t^2} \frac{1}{p} \sum_i \ell(-f_t(\mathbf{x}_i) + \sum_k \hat{\gamma}_k \alpha_k) \\ &\leq -\frac{\nu}{4} \frac{(\gamma_t - \gamma_t)^2}{(1 + \hat{\gamma}_t^2)p} \leq -\frac{\nu \delta^2 \hat{\gamma}_t^2}{8p} \end{aligned} \quad \square$$

Theorem 3. After $T = O(\frac{p+n}{\nu \delta^2 \gamma^{*2}})$ iterations, SoftRankBoost terminates and it holds that

$$\Pr_{S^+, S^-} \left\{ \frac{\tilde{f}_T(\mathbf{x}_i) - \tilde{f}_T(\mathbf{x}'_j)}{2} \leq (1 - \delta)\gamma^* \right\} \leq \frac{(p+n)\nu}{pn}.$$

Proof. Since $\gamma_t \geq \gamma^*$, we have

$$\begin{aligned} & \Pr_{S^+, S^-} \left\{ \frac{\tilde{f}_T(\mathbf{x}_i) - \tilde{f}_T(\mathbf{x}'_j)}{2} \leq (1 - \delta)\gamma^* \right\} \\ & \leq \Pr_{S^+, S^-} \{ \tilde{f}_T(\mathbf{x}_i) - \tilde{f}_T(\mathbf{x}'_j) \leq 2\hat{\gamma}_T \} \end{aligned}$$

Note that $\tilde{f}_t(\mathbf{x}_i) - \tilde{f}_t(\mathbf{x}'_j) < 2\hat{\gamma}_t$ implies $\tilde{f}_t(\mathbf{x}_i) < \hat{\gamma}_t$ or $-\tilde{f}_t(\mathbf{x}'_j) < \hat{\gamma}_t$. So, by using the union bound, we have

$$\begin{aligned} & \Pr_{S^+, S^-} \{ \tilde{f}_T(\mathbf{x}_i) - \tilde{f}_T(\mathbf{x}'_j) \leq 2\hat{\gamma}_T \} \\ & \leq \frac{1}{pn} \sum_i \sum_j I(\tilde{f}_T(\mathbf{x}_i) < \hat{\gamma}_T) + \frac{1}{pn} \sum_i \sum_j I(-\tilde{f}_T(\mathbf{x}'_j) < \hat{\gamma}_T) \\ & = \frac{1}{p} \sum_i I(\tilde{f}_T(\mathbf{x}_i) < \hat{\gamma}_T) + \frac{1}{n} \sum_j I(-\tilde{f}_T(\mathbf{x}'_j) < \hat{\gamma}_T) \\ & \leq \frac{1}{p} \sum_i L(-f_T(\mathbf{x}_i) + \hat{\gamma}_T \sum_t \alpha_t) + \frac{1}{n} \sum_j L(f_T(\mathbf{x}'_j) + \hat{\gamma}_T \sum_t \alpha_t) \end{aligned}$$

By Lemma 5, after $T = \frac{8 \max\{p, n\}}{\nu \delta^2 \gamma^{*2}}$ iterations, the first term in the right hand side of the last inequality is bounded by

$$1 - T \frac{\nu \delta^2 \gamma^{*2}}{8p} \leq \frac{\nu}{p}.$$

The second term is also bounded by $1 - T \frac{\nu \delta^2 \gamma^{*2}}{8n} \leq \frac{\nu}{n}$. So, SoftRankBoost terminates in T iterations. Then, after T iterations,

$$\Pr_{S^+, S^-} \left\{ \frac{\tilde{f}_T(\mathbf{x}_i) - \tilde{f}_T(\mathbf{x}'_j)}{2} \leq (1 - \delta)\gamma^* \right\} \leq \frac{\nu}{p} + \frac{\nu}{n} = \frac{(p + n)\nu}{pn}. \quad \square$$

5 Experiments

We conduct preliminary experiments over artificial datasets generated by a sparse linear classifier with random noise over labels. Each artificial dataset consists of n -dimensional $\{-1, +1\}$ -valued vectors with $n = 100$. Each vector is labeled with a threshold function f , which is represented as $f(\mathbf{x}) = \text{sign}(x_{i_1} + \dots + x_{i_k} + 1)$ for some $i_1, \dots, i_k$ s.t. $1 \leq i_1 \leq i_2 \leq \dots \leq i_k \leq n$. For $k = 30$ we generate random $m = 1000$ examples labeled by the linear threshold function, so that positive and negative examples are equally likely. In addition, we add random noise over labels by flipping the labels randomly with probability 5%, 10%, and 15%, respectively.

For each dataset, we prepare decision stumps and the constant hypothesis $+1$ (i.e. the hypothesis that always answers $+1$) as weak hypotheses. For each ± 1 -valued attribute of each example, we prepare the decision stump which answers the value of the attribute.

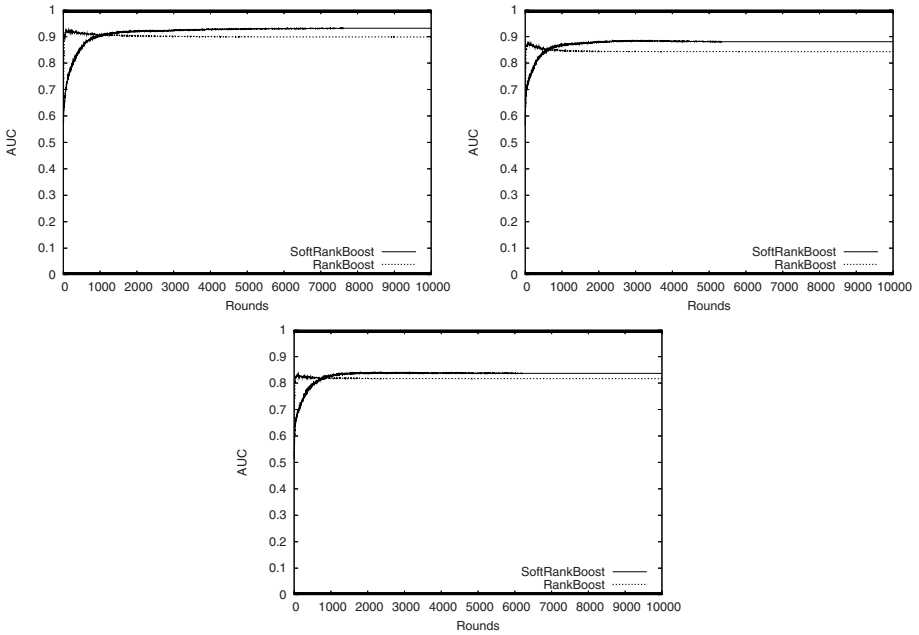


Fig. 3. Test AUCs of boosting algorithms for balanced artificial datasets with random noises 5% (upper left), 10% (upper right), and 15% (lower) over labels. Here, positive and negative examples are generated equally likely.

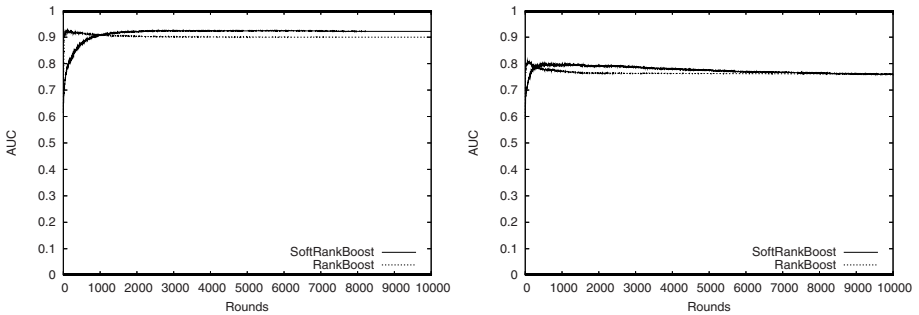


Fig. 4. Test AUCs of boosting algorithms for unbalanced artificial datasets with random noises 5%. Here, the ratios of positive and negative examples are roughly 7 : 3 (left) and 9 : 1 (right), respectively

We evaluate the boosting algorithms by cross validation. We split each data randomly 20 times, where each example is put into a training set with probability 70% and a test set with probability 30%. For each training set, we run RankBoost in 10000 steps, and run SoftRankBoost with parameters $\delta = 0.5$, and $\nu = (0.1)m$. Note that, 10000 steps are sufficiently long for RankBoost to

converge over the datasets. The AUC of the final hypothesis of each algorithm is evaluated over test data and we average the AUC over 20 trials. The results are summarized in Figure 3. As can be seen in Figure 3, SoftRankBoost achieves higher AUCs than RankBoost over balanced datasets with noises.

Then we change the ratio of positive and negative examples and examine how SoftRankBoost behaves for unbalanced datasets. More precisely, we change the probability of generating a negative instance to be 70% and 90%, respectively. Here we fix the random noise probability with 5%.

The result is summarized in Figure 4. Even though our theoretical guarantee is not good for unbalanced datasets, SoftRankBoost still outperforms RankBoost.

6 Conclusions and Future Work

In this paper, for bipartite ranking problems, we propose SoftRankBoost, a smooth version of RankBoost which achieves large margin over pairs of positive and negative instances. As future work, we would like to understand SoftRankBoost deeply. In particular, many boosting algorithms can be viewed as optimizers of some optimization problems (e.g., [19]). Clarifying the underlying optimization problem might give us a better algorithm. Also, we plan to evaluate the performance of SoftRankBoost over real datasets.

References

- [1] Balcan, N., Bansal, N., Beygelzimer, A., Coppersmith, D., Langford, J., Sorkin, G.B.: Robust reductions from ranking to classification. In: *Proceedings of the 20th Annual Conference on Learning Theory*, pp. 604–619 (2007)
- [2] Bradley, A.P.: The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition* 30, 1145–1159 (1997)
- [3] Bradley, J.K., Shapire, R.: Filterboost: Regression and classification on large datasets. In: *Advances in Neural Information Processing Systems* 20, pp. 185–192 (2008)
- [4] Brefeld, U., Scheffer, T.: Auc maximizing support vector learning. In: *Proceedings of the ICML Workshop on ROC Analysis in Machine Learning* (2005)
- [5] Cohen, W.W., Schapire, R.E., Singer, Y.: Learning to order things. *Journal of Artificial Intelligence Research* 10, 243–279 (1999)
- [6] Cortes, C., Mohri, M.: Auc optimization vs. error rate minimization. In: *Advances in Neural Information Processing Systems* 16 (2004)
- [7] Domingo, C., Watanabe, O.: MadaBoost: A modification of AdaBoost. In: *Proceedings of 13th Annual Conference on Computational Learning Theory*, pp. 180–189 (2000)
- [8] Fawcett, T.: Roc graphs: Notes and practical considerations for researchers. Technical report, HPL-2003-4, HP (2003)
- [9] Freund, Y., Iyer, R., Shapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research* 4, 933–969 (2003)
- [10] Gavinsky, D.: Optimally-smooth adaptive boosting and application to agnostic learning. *Journal of Machine Learning Research* (2003)

- [11] Hatano, K.: Smooth boosting using an information-based criterion. In: Proceedings of the 17 th International Conference on Algorithmic Learning Theory, pp. 304–319 (2006)
- [12] Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (2002)
- [13] Long, P.M., Servedio, R.A.: Boosting the area under the roc curve. In: Advances in Neural Information Processing Systems 20 (2008)
- [14] Rudin, C.: Ranking with a p-norm push. In: Proceedings of 19th Annual Conference on Learning Theory, pp. 589–604 (2006)
- [15] Rudin, C., Cortes, C., Mohri, M., Shapire, R.E.: Margin-based ranking meets boosting in the middle. In: Proceedings of the 18th Annual Conference on Learning Theory, pp. 63–78 (2005)
- [16] Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. *Machine Learning* 37(3), 297–336 (1999)
- [17] Schölkopf, B., Smola, A.J., Williamson, R.C., Bartlett, P.L.: New support vector algorithms. *Neural Computation* 12(5), 1207–1245 (2000)
- [18] Servedio, R.A.: Smooth boosting and learning with malicious noise. *Journal of Machine Learning Research* 4, 633–648 (2003)
- [19] Warmuth, M., Glocer, K., Rätsch, G.: Boosting algorithms for maximizing the soft margin. In: Advances in Neural Information Processing Systems 20, pp. 1585–1592 (2008)

Learning with Continuous Experts Using Drifting Games

Indraneel Mukherjee and Robert E. Schapire

Princeton University
Department of Computer Science
35 Olden Street
Princeton, NJ 08540
`{imukherj,schapire}@cs.princeton.edu`

Abstract. We consider the problem of learning to predict as well as the best in a group of experts making continuous predictions. We assume the learning algorithm has prior knowledge of the maximum number of mistakes of the best expert. We propose a new master strategy that achieves the best known performance for online learning with continuous experts in the mistake bounded model. Our ideas are based on drifting games, a generalization of boosting and online learning algorithms. We also prove new lower bounds based on the drifting games framework which, though not as tight as previous bounds, have simpler proofs and do not require an enormous number of experts.

1 Introduction

We consider the problem of learning to predict as well as the best in a group of experts. Our model consists of a series of rounds. In each round, experts make predictions in $[-1, +1]$. This can be interpreted as giving a binary prediction, for example, if it will rain or not, with a certain degree of *confidence*. In particular, this means that an expert can choose to *abstain* from giving any prediction at all, in which case it predicts 0. The problem is to design a master algorithm that combines the expert predictions in each round to give its own binary prediction in $\{-1, +1\}$. At the end of each round, nature (or an adversary) reveals the truth, which is a value in $\{-1, +1\}$. The experts and the master suffer loss that depends on the amount by which their predictions deviated from the truth. Our goal is to ensure that our master algorithm does not suffer much loss relative to the best expert.

An important feature of our model, which we define rigorously in Section 2, is that we assume the master algorithm has prior knowledge of a bound k on the total loss that the best expert will suffer. With binary experts, outputting predictions in $\{-1, +1\}$, this problem was essentially solved entirely by Cesa-Bianchi et al. [3] who proposed the Binomial Weights (BW) algorithm. However, their work cannot be applied to our setting since here the experts are continuous, with predictions in $[-1, +1]$. In such a setting, other methods, notably

exponential-weight algorithms [2, 6, 7], can be used instead. However, such algorithms do not enjoy the same level of tight optimality of the BW algorithm, and it has been an open problem since the introduction of BW as to whether this method can be generalized to continuous experts.

In this paper, we present just such a generalization. In Section 3, we propose a new master strategy which gives the best known performance for this problem. Our algorithm predicts using a weighted majority of the experts' predictions in each round, where the weights are carefully chosen to ensure that the master's loss is small relative to k . We also show that our algorithm runs in polynomial time.

Our algorithm is based on the *drifting games* framework introduced by Schapire [9]. This framework generalizes a number of online and boosting learning algorithms, including boost-by-majority [4], AdaBoost [6], the weighted majority algorithm [7] and Binomial Weights [3]. We apply the drifting games framework directly both to derive our algorithm, and to analyze its performance which, as seen in Section 4, relies heavily on properties of drifting games.

We also provide in Section 5 new lower bound constructions for master algorithms which employ weighted-majority predictions. These are slightly weaker than those provided by Cesa-Bianchi et al. [3] which already show that our algorithm is nearly the best possible when the number of experts is very large. However, their techniques are based on Spencer's [10] sophisticated results for Ulam's game, and require an enormous number of experts. In contrast, our lower bounds use simpler arguments based on the drifting games framework, and are meaningful for any number of experts.

A consequence of our results is that learning in our framework with continuous experts is apparently no harder than learning with abstaining experts, i.e., experts whose predictions are restricted to be in $\{-1, 0, +1\}$ (assuming that $2k$ is an integer), although there does appear to be a small gap between abstaining and binary experts.

Other related work. Abernethy et al. [?] extended the BW algorithm to the setting where the experts remain binary, but the master is allowed to predict continuously. Their results also apply to a setting where experts can split themselves to randomly predict -1 or $+1$. Note that such splitting experts, although superficially similar to the ones in the current paper, are in fact quite different, and as a consequence, their results cannot be applied immediately to our setting.

Continuous-time versions of drifting games, with potential applications to online learning, were studied by Freund and Oppen [5]. In their setting, learning rounds are no longer discrete, but are instead continuous.

2 Expert Learning Model

Our expert learning model can be viewed as the following game. The players of the game are a fixed set of m experts, a master algorithm, and an adversary. The game proceeds through T rounds. In each round t , the following happen:

- The master chooses real weights $w_1^t, \dots, w_m^t$ over the experts.
- Each expert i makes a prediction $x_i^t \in [-1, +1]$. The experts' predictions are controlled by the adversary; we distinguish between the experts and the adversary for clarity of exposition.
- The master predicts $\hat{y}^t \triangleq \text{sign}(\sum_i w_i^t x_i^t) \in \{-1, 0, +1\}$. The sign function maps positive reals to 1, negative reals to -1 and 0 to itself.
- The adversary then chooses a label $y^t \in \{-1, +1\}$, causing expert i to suffer loss $\frac{1}{2}|y^t - x_i^t|$, and the master to suffer loss $\mathbf{1}(y^t \neq \hat{y}^t)$, where $\mathbf{1}$ is the indicator function. Note that predicting 0 counts as a mistake.

The total loss of any player is the sum of the losses in each round. It is guaranteed that some expert will suffer less than k total loss, where k is known ahead of time to the master. The goal of the master is to come up with a strategy to choose distributions $\mathbf{w}^t$ in each round, so as to minimize his loss against the worst possible adversary. The performance of every fixed strategy will thus be a function of m and k .

We will only consider *conservative* master algorithms, i.e., algorithms that *ignore* rounds where it does not make a mistake, so that the weights it chooses in a certain round depend only on past rounds where it made a mistake. This will also allow us to assume that as long as the game can continue, the master makes a mistake in every round. Since one can easily convert any master algorithm in a mistake bounded model like ours to a conservative one without loss of performance, we do not lose generality with this assumption.

3 Choosing Weights

We describe a strategy of the master for choosing a distribution on the expert in each round. Computing this strategy requires playing a different type of game called a *drifting game* introduced by Schapire ([9]). We begin with an abstract definition of the game, and then go on to show how such games can be used to derive the BW algorithm [3], which is the optimal (in the broadest possible sense) master strategy in the case of binary experts. We then show how similar ideas can be used to derive a master algorithm for continuous experts.

3.1 Drifting Games

A drifting game is played by a *shepherd* and m *sheep* (also known as *chips* in [9]) floating in $\mathbb{R}^d$. In the rest of the paper, the dimension d of every drifting game considered is 1. The game proceeds through T rounds. In each round t , the following happen:

- The shepherd chooses a weight $w_i^t \in \mathbb{R}$ for each sheep i . The sign indicates the direction he intends the sheep to move, and the magnitude encodes the importance he places on that sheep.

- Sheep i responds by shifting by z_i^t , where the limited imagination of sheep-kind forces z_i^t to belong to a fixed set of directions B . Additionally, the prowess of the shepherd demands that the following drifting constraint be obeyed

$$\sum_{i=1}^m w_i^t z_i^t \geq \delta \sum_{i=1}^m |w_i|. \quad (1)$$

Here $\delta \geq 0$ and $B \subseteq \mathbb{R}$ are parameters of the game.

The shepherd suffers a loss $L(s)$ for every sheep that is at location $s \in \mathbb{R}$ at the end of the game; here $L : \mathbb{R} \rightarrow \mathbb{R}$ is a real function on the space. Initially, all the sheep are at the origin, so at the end of the game, sheep i is at $\sum_{t=1}^T z_i^t$. The goal of the shepherd is to choose weights in a way that would minimize its average loss $\frac{1}{m} \sum_i L(\sum_t z_i^t)$, assuming the worst behavior from the sheep.

Schapire [9] suggests a shepherd strategy, OS, based on a set of potential functions $\phi_t : \mathbb{R}^d \rightarrow \mathbb{R}$ defined recursively as follows:

- $\phi_T(x) = L(x)$
- $\phi_{t-1}(x) = \min_{w \in \mathbb{R}} \max_{z \in B} (\phi_t(x + z) + wz - \delta|w|)$

Denoting by s_i^t the position of sheep i at time t , the OS algorithm chooses w_i^t as follows

$$w_i^t \in \arg \min_{w \in \mathbb{R}} \max_{z \in B} (\phi_{t+1}(s_i^t + z) + w - \delta|w|).$$

(In this paper, we regard $\arg \min$ or $\arg \max$ as returning the set of all values realizing the minimum or maximum.) Schapire [9] provides an upper bound on the performance of the OS algorithm, and argues that (under some natural assumptions) it is optimal when the number of sheep m is very large. We record the results in the theorem below.

Theorem 1 (Drifting Games [9]). *Under some technical assumptions on B and L , the loss suffered by the OS algorithm is upper bounded by $\phi_0(\mathbf{0})$ where ϕ is defined as above. Further, given any $\epsilon > 0$, for sufficiently large m , the sheep can force any shepherd algorithm playing for T rounds to suffer a loss of $\phi_0(\mathbf{0}) - \epsilon$.*

3.2 Learning with Binary Experts Using Drifting Games

Consider our expert learning model with the change that experts make $\{-1, +1\}$ instead of continous predictions. The Binomial Weights algorithm [3] is the best possible master strategy for this problem, even among master algorithms not restricted to predicting a weighted majority of the experts' predictions at each stage. We show how a master can simulate a drifting game to derive a strategy for choosing weights on the experts so as to perform as well as the BW algorithm.

The drifting game takes place in $\mathbb{R}$, so that $d = 1$. Its parameters are $B = \{-1, +1\}$ and $\delta = 0$, and the loss function is $L(s) = \mathbf{1}(x \leq 2k - T)$. The number

of rounds is $T = T_0 + 1$ where T_0 will be specified later. For every expert, there is a sheep. At the beginning of a round, the master uses the shepherd's choice $w_1, \dots, w_m$ for that round to assign weights to experts. After seeing the expert predictions x_i and the label y produced by the adversary, the master causes sheep i to drift by $z_i = -yx_i$. The drifting constraint (1) holds since we are in the conservative setting and assume a mistake is made by the master in each round. Schapire [9] shows that the resulting algorithm is equivalent to BW, for a certain choice of T_0 .

We use the notation $\binom{q}{\leq k}$ to denote $\sum_{i=0}^k \binom{q}{i}$.

Theorem 2 (Learning with Binary Experts [3],[9]). *Consider the expert learning model described in Section 2, with the change that the expert predictions lie in $\{-1, +1\}$. For this problem, when T_0 is set to be*

$$\max \left\{ q \in \mathbb{N} : q \leq \lg m + \lg \binom{q}{\leq k} \right\}$$

the number of mistakes made by the master algorithm described in this section is upper bounded by T_0 . Further, the resulting algorithm can be computed efficiently.

Proof. At the heart of the proof, and the reason behind our choice of T_0 , lies the following result which was proved by Schapire [9]: If a drifting game with parameters $\delta = 0, B = \{-1, +1\}$ and loss function $L(x) = \mathbf{1}(x \leq 2k - T)$ is played for T rounds, then ϕ_t can be computed exactly, yielding

$$\phi_0(0) = 2^{-T} \binom{T}{\leq k}. \quad (2)$$

Note that the position of a sheep after t rounds is $2M - t$, where M is the loss suffered by the corresponding expert till then. Since we are guaranteed a mistake bound of at most k on some expert, we always have $\sum_i L(s_i^t) \geq 1$, where s_i^t is sheep i 's position after t rounds of play. If the game could continue for $T = 1 + T_0$ rounds, using Theorem 1 and (2) we would have the following contradiction:

$$\frac{1}{m} \leq \frac{1}{m} \sum_i L(s_i^T) \leq \phi_0(0) = 2^{-T} \binom{T}{\leq k} < \frac{1}{m}$$

The last inequality follows from the fact that T_0 was chosen to satisfy $T_0 = \max\{\# \text{ of rounds} : \phi_0(\mathbf{0}) \geq \frac{1}{m}\}$. This upper bounds the maximum number of rounds for which the game can continue, or equivalently, the maximum number of mistakes our master algorithm makes, by T_0 . $\square$

3.3 Drifting Games for Continuous Experts

The same approach from the previous section can be applied to our expert learning model, where experts make $[-1, +1]$ predictions. The drifting game parameter B changes to $B = [-1, +1]$, and a new expression for T_0 has to be chosen; everything else remains the same. We summarize the master strategy in Algorithm 1, where we choose $T_0 = \max\{q \in \mathbb{N} : q \leq \lg m + \lg \binom{T+1}{\leq k}\}$. We can now state our first main result.

Algorithm 1. Master algorithm for continuous experts**Require:** k - mistake bound, m - number of experts

```

 $T_0 \leftarrow \max\{q \in \mathbb{N} : q \leq \lg m + \lg(\binom{T+1}{\leq k})\}$ 
 $T \leftarrow 1 + T_0, B \leftarrow [-1, +1]$ 
 $\delta \leftarrow 0, L \leftarrow \mathbf{1}(x \leq 2k - T)$ 
Setup drifting game with  $T, B, \delta, L$ , and shepherd OS

{Note: Game cannot continue beyond  $T_0$  rounds}
for  $t = 1$  to  $T_0$  do
  Accept  $w_1^t, \dots, w_m^t$  from shepherd
  Accept predictions  $x_1^t, \dots, x_m^t$  from experts.
  Predict  $\hat{y}^t = \text{sign}(\sum_i w_i x_i^t)$ 
  Accept label  $y^t$  from adversary.
  For each  $i$ , make sheep  $i$  drift by  $z_i^t \triangleq -y^t x_i^t$ 
end for

```

Theorem 3 (Learning with Continuous Experts). *Consider the expert learning model described in Section 2. For that problem, the loss of the master algorithm described in Algorithm 1 is upper bounded by T_0 , which is equal to*

$$\max \left\{ q \in \mathbb{N} : q \leq \lg m + \lg \binom{q+1}{\leq k} \right\}. \quad (3)$$

As in learning with binary experts, the choice of T_0 in Theorem 3 is dictated by the analysis of the drifting game used for playing with continuous experts. This analysis also constitutes our main technical contribution, and is summarized in the next theorem, but we defer a proof till the next section.

Theorem 4 (Drifting Games for $[-1, +1]$ Experts). *Consider the drifting game with parameters $\delta = 0, B = [-1, +1]$, total number of rounds T and loss function $L(x) = \mathbf{1}(x \leq 2k - T)$. The value of the potential function for this game at any integer point s is given by*

$$\phi_{T-t}(s + 2k - T) = \begin{cases} 1 & \text{if } s \leq 0 \\ 1 - 2^{-t} \sum_{i=0}^{s-1} \binom{t+i}{\lfloor \frac{t+i}{2} \rfloor} & \text{else.} \end{cases} \quad (4)$$

In particular we have

$$\phi_0(0) = 2^{-T} \binom{T+1}{\leq k}.$$

Further, the OS strategy for this game can be computed efficiently.

Proof of Theorem 3. Observe that $T_0 = \max\{ \# \text{ of rounds} : \phi_0(0) \geq \frac{1}{m} \}$, where ϕ_0 is the potential associated with the drifting game in Theorem 4. The rest of the proof is the same as that for Theorem 2. $\square$

We can loosely upper bound the expression for the number of mistakes in Theorem 3 by

$$2k + \ln m \left(1 + \sqrt{1 + \frac{4k}{\ln m}} \right) - 1.$$

In Section 5 we will prove that, when the number of experts m is around 2^k , the mistake guarantee given in Theorem 3 is the best possible, up to an additive $O(\log k)$ term, when considering master algorithms that predict a weighted majority of the experts' predictions in each round.

4 Analysis of Drifting Games for Continuous Experts

Throughout we will be using the following two facts: ϕ_t is decreasing, and takes values in $[0, 1]$. These facts were proved more generally by Schapire [9].

We begin with a technical result necessary for proving Theorem 4.

Theorem 5 (Piecewise Convexity). *For every round t , ϕ_t is piecewise convex with pieces breaking at integers, i.e., for every integer n , ϕ_t is convex in $[n, n + 1]$.*

The proof of this theorem is complicated and we defer it to Section 6. The proof relies on Lemma 1, which will also be useful otherwise. This lemma can be proved using a more general result in [9], but here we give a direct proof for the case of interest.

Lemma 1. *If ϕ_t is piecewise convex with pieces breaking at integers, then for $s \notin \mathbb{Z}$,*

$$\phi_{t-1}(s) = \max \left\{ \frac{z\phi_t(s+z') - z'\phi_t(s+z)}{z-z'} : z, z' \in \mathbb{Z}, zz' < 0 \right\} \quad (5)$$

where

$$Z = \{z \in [-1, +1] : s+z \in \mathbb{Z}\} \cup \{-1, +1\}. \quad (6)$$

For s integral, $\phi_{t-1}(s)$ is the maximum of $\phi_t(s)$ and the above expression.

Proof. By definition

$$\phi_{t-1}(s) = \min_w \max_{z \in [-1, +1]} (\phi_t(s+z) + wz).$$

For fixed s and w , our assumptions imply that $\phi_t(s+z) + wz$ is piecewise convex in z . As z varies over the convex set $[-1, +1]$, the maximum will be realized either at an endpoint, -1 or 1 , or when $s+z$ lies at one of the endpoints of the convex pieces, which happens at the integers. This shows that we can restrict z to Z while evaluating $\phi_{t-1}(s)$.

Denote by Δ the simplex of distributions over Z . By the discussion above,

$$\begin{aligned} \phi_{t-1}(s) &= \min_w \max_{z \in Z} (\phi_t(s+z) + wz) \\ &= \min_w \max_{p \in \Delta} \mathbb{E}_{z \sim p} [\phi_t(s+z) + wz] \\ &= \max_{p \in \Delta} \min_w \mathbb{E}_{z \sim p} [\phi_t(s+z) + wz] \end{aligned}$$

where the last equality comes from Corollary 37.3.2 of [8]. Interpreting the right side as the Lagrangean dual we may compute $\phi_{t-1}(s)$ as the solution to the following optimization problem

$$\begin{aligned} \max_{p \in \Delta} \quad & \mathbb{E}_{z \sim p} [\phi_t(s + z)] \\ \text{s.t.} \quad & \mathbb{E}_{z \sim p}[z] = 0. \end{aligned}$$

The above is a linear program and is hence optimized at vertices of the polytope $\{p \in \Delta : \mathbb{E}_{z \sim p}[z] = 0\}$, which are mean-zero distributions supported on two points z, z' of opposite signs, or concentrated on 0 when feasible i.e. when $s \in Z$. Maximizing $\mathbb{E}_{z \sim p} [\phi_t(s + z)]$ over such vertices p yields the lemma. $\square$

It is now straightforward to prove Theorem 4.

Proof of Theorem 4: Theorem 5 and Lemma 1 imply that for integer points s

$$\phi_{t-1}(s) = \max \left\{ \phi_t(s), \frac{\phi_t(s-1) + \phi_t(s+1)}{2} \right\}. \quad (7)$$

One can finish the proof by directly substituting into (7) the expression for $\phi_t(s)$ given in (4), and verifying that the inequality holds. We omit calculations.

For efficiency, note that the value of any ϕ_t at any point s depends upon values at $\mathbb{Z}$ and $s + \mathbb{Z}$. One can easily check that, for all t , $\phi_t(s) = 1$ for $s \leq 2k - T$ and $\phi_t(s) = 0$ for $s \geq T$. Since the value at integers can be easily computed from the expression in the theorem, we can compute $\phi_t(s)$ by applying standard dynamic programming techniques in time polynomial in T . $\square$

Notice that (7) is the same as what we would get if the sheep were allowed to drift only by $-1, 0, +1$ at each time step. Correspondingly, in terms of provable upper bounds, our algorithm performs no worse with continuous experts than it does with abstaining experts, while with binary experts, the upper bound on performance is a tiny bit better.

5 Lower Bounds

In this section we provide lower bounds for online learning with continuous experts which almost match the upper bounds of Theorem 3, thus showing that the drifting game based Algorithm 1 is near optimal.

Theorem 6 (Lower bound for expert learning). *Consider the expert learning model defined in section 2. For every master algorithm, the adversary can choose labels and cause the experts to make predictions in each round in a manner so as to force the master algorithm to suffer a loss of*

$$\max \left\{ q \in \mathbb{N} : q < \lg \left(\frac{m}{\sqrt{k}} \right) + \lg \left(\frac{q+1}{\leq k} \right) + \Theta(1) \right\}, \quad (8)$$

where $\Theta(1)$ is a quantity bounded between some absolute constants c_1 and c_2 .

The loss bound given above, and the upper bound in 3 define the smallest integer T such that $2^{-T} \binom{T+1}{\leq k}$ is less than $\frac{1}{m}$ and $O(\frac{\sqrt{k}}{m})$ respectively. Since $O(\log m)$ rounds will always be necessary, and $2^{-T} \binom{T+1}{\leq k}$ decreases exponentially fast when $T > 3k$, we see that the gap between the upper and lower bounds is only $O(\log k)$ when the number of experts m is around 2^k . We believe that a more careful analysis will show that the real gap between the upper and lower bounds is much smaller.

The proof of Theorem 6 consists of showing how an adversary in the expert model can exploit adversarial sheep movement in the drifting game to force any master algorithm to suffer high loss. We then resort to the following result on drifting games, whose proof is deferred to the next section, to complete our argument. This is the converse of what we saw in Section 3.3, where a well performing shepherd algorithm gave rise to master algorithms suffering low loss.

Theorem 7. *Consider the drifting game with parameters $\delta = 0$, $B = [-1, +1]$, number of rounds T and loss function $\mathbf{1}(x \leq 2k - T)$. For any shepherd algorithm, there exists a strategy for the sheep that causes the shepherd to suffer a loss of*

$$\phi_0(0) - \frac{\Theta(\sqrt{k})}{m}$$

at the end of the game.

Proof of Theorem 6: The adversary in our expert model (defined in Section 2) simulates a drifting game in $\mathbb{R}$, with parameters as above. The drifting game is played for T rounds, where T is given by the expression (8). For every expert, there is a sheep. At the beginning of a round, if the master places weights $w_1, \dots, w_m$ on the experts, the adversary causes the shepherd to drive each sheep i in direction w_i . If the sheep drift in direction $z_1, \dots, z_m$, he causes expert i to predict $x_i = z_i$ (remember the adversary controls expert predictions). The drifting constraint $\sum_i w_i z_i \geq \delta = 0$ ensures that the weighted majority prediction of the master is 0 or 1. The adversary then outputs the label $y = -1$, causing the master to make a mistake in each round.

Note that the position of a sheep after t rounds is $2M - t$ where M is the loss suffered by the corresponding expert till then; thus an expert has suffered at most k loss if and only if the corresponding sheep lies at a point less than $2k - T$ at the end of the game. Hence, by our choice of loss function, the mistake bound on the experts is equivalent to ensuring the constraint that the loss suffered by the shepherd algorithm is strictly positive at the end of the game, so that at least one sheep has a final loss of 1.

Theorem 7 guarantees that the sheep can drift in a way so that the shepherd suffers at least $\phi_0(0) - \Theta(\sqrt{k})/m$ loss, where we know from Theorem 4 that $\phi_0(0) = \binom{T+1}{\leq k}$. Our choice of T satisfies $\phi_0(0) - \Theta(\sqrt{k})/m > 0$, completing the proof. $\square$

5.1 Lower Bound for Drifting Game

We prove Theorem 7. Schapire ([9]) provides a similar though slightly weaker lower bound ($\phi_0(0) - O(T/\sqrt{m})$ instead of $\phi_0(0) - (\sqrt{k}/m)$) which leads to considerably weaker expert learning lower bounds. The reason is that Schapire's arguments hold for much more general drifting games. By carefully tailoring his proof to our specific learning model, we achieve significant improvements.

Proof of Theorem 7: We will show that on round t , the sheep can choose to drift in directions z_i so that

$$\frac{1}{m} \sum_i \phi_{t+1}(s_i^{t+1}) \geq \frac{1}{m} \sum_i \phi_t(s_i^t) - \frac{U_t}{m}. \quad (9)$$

Here s_i^t is the position of sheep i in round t , and

$$U_t \triangleq \max_{s^t} \frac{\phi_{t+1}(s^t - 1) - \phi_{t+1}(s^t + 1)}{2} \quad (10)$$

where the maximum is taken over all possible *integral* positions s^t of any sheep in round t . Note that this is different from the set of all possible positions, since the movement of the sheep is restricted to change by at most $+1$ or -1 in each round. Among the possible positions, we take supremum over only those positions which happen to lie at an integer.

Repeatedly applying the above yields

$$\frac{1}{m} \sum_i L(s_i^T) \geq \phi_0(0) - \frac{1}{m} \sum_t U_t.$$

Appealing to Lemma 4 will then produce the desired bound.

For each i , $s_i^0 = 0$. Our sheep strategy will choose every drift to be in $\{-1, 0, 1\}$. Hence we may assume $s_i^t \in \mathbb{Z}$ for each i, t .

Fix a round t . From Lemma 1 we have

$$\phi_t(s) = \max \left\{ \phi_{t+1}(s), \frac{\phi_{t+1}(s-1) + \phi_{t+1}(s+1)}{2} \right\}.$$

Let $I = \{1, \dots, m\}$, $I_0 = \{i : \phi_t(s_i^t) = \phi_{t+1}(s_i^t)\}$, $I_1 = I \setminus I_0$. For each $i \in I_0$ we set $z_i^t = 0$. This ensures

$$\sum_{i \in I_0} \phi_{t+1}(s_i^{t+1}) = \sum_{i \in I_0} \phi_t(s_i^t).$$

For each $i \in I_1$ we must have

$$\phi_t(s_i^t) = \frac{\phi_{t+1}(s_i^t - 1) + \phi_{t+1}(s_i^t + 1)}{2}$$

For such i we will choose z_i^t in $\{-1, +1\}$. Define $a_i^t \triangleq \frac{\phi_{t+1}(s_i^t - 1) - \phi_{t+1}(s_i^t + 1)}{2}$. Then, for each $i \in I_1$,

$$\phi_{t+1}(s_i^{t+1}) = \phi_t(s_i^t) - z_i^t a_i^t$$

since $s_i^{t+1} = s_i^t + z_i^t$. Thus

$$\sum_{i \in I_1} \phi_{t+1}(s_i^{t+1}) = \sum_{i \in I_1} \phi_t(s_i^t) - \sum_{i \in I_1} z_i^t a_i^t$$

Note that $a_i^t \in [0, U_t]$ by definition. If the shepherd weights for this round are $w_1^t, \dots, w_m^t$, it suffices to ensure that $\sum_{i \in I_1} w_i^t z_i^t \geq 0$ while keeping $\sum_{i \in I_1} a_i^t z_i^t$ below U_t .

By Lemma 2, there exists a subset $P \subseteq I_1$ such that

$$\left| \sum_{i \in P} a_i^t - \sum_{j \in I_1 \setminus P} a_j^t \right| \leq U_t.$$

Assume without loss of generality that $\sum_{i \in P} w_i^t - \sum_{i \in I_1 \setminus P} w_i^t \geq 0$. Then assigning $z_i^t = +1$ for $i \in P$ and $z_i^t = -1$ for $i \in I_1 \setminus P$ would ensure the drifting constraints as well as (9), completing our proof. $\square$

Lemma 2. *For any sequence $a_1, \dots, a_n$ of numbers in $[0, U]$*

$$\min_{P \subseteq I} \left| \sum_{i \in P} a_i - \sum_{j \in I \setminus P} a_j \right| \leq U$$

where $I = \{1, \dots, n\}$.

Proof. Define discrepancy to be the argument of the min, and let P^* realize the minimum. If P^* 's discrepancy were greater than U , we could transfer any a_i from the heavier group to get a partition with lower discrepancy, a contradiction.

Notice that the U_t can be trivially bounded by 1, since the ϕ_t take values in $[0, 1]$. That would give us a lower bound of $\phi(0) - \frac{T}{m}$. By being more careful, we get the following improvement.

Lemma 3. *Define U_t as in (10). Then*

$$U_{T-t} = \begin{cases} 2^{-t} \binom{t}{k} & \text{if } t > 2k \\ 2^{-t} \binom{t}{\lceil \frac{t}{2} \rceil} & \text{if } t \leq 2k. \end{cases}$$

Proof. Using (4), we have, for $s \geq 0$

$$\begin{aligned} & \frac{1}{2} [\phi_{T-t}(s-1+2k-T) - \phi_{T-t}(s+1+2k-T)] \\ &= 2^{-t-1} \left(\binom{t}{\lceil \frac{t+s-1}{2} \rceil} + \binom{t}{\lceil \frac{t+s}{2} \rceil} \right). \end{aligned} \quad (11)$$

Let $s+2k-T$ be the position of a sheep at the end of $T-t$ rounds. Since it can drift by at most -1 in the negative direction in any round, we have

$s + 2k - T \geq t - T$ so that $s \geq t - 2k$. We take two cases, depending on the value of k .

Suppose $t > 2k$. Then $s \geq t - 2k \geq 1$. Since (11) is larger for smaller (and non-negative) s , we can plug in $s = t - 2k$ to compute U_{T-t} .

$$U_{T-t} = 2^{-t-1} \left(\binom{t}{\lceil \frac{2t-2k-1}{2} \rceil} + \binom{t}{\lceil \frac{2t-2k}{2} \rceil} \right) = 2^{-t} \binom{t}{t-k} = 2^{-t} \binom{t}{k}.$$

When $t \leq 2k$, s can be less than 1. If $s < 0$, the left hand side of (11) is zero. If $s = 0$, the right hand side of the same equation equals $2^{-t} \binom{t}{\lceil \frac{t}{2} \rceil}$. Hence for $t \leq 2k$, $U_{T-t} = 2^{-t} \binom{t}{\lceil \frac{t}{2} \rceil}$.

Lemma 4. Define U_t as in (10). Then, $\sum_t U_t \leq \Theta(\sqrt{k})$.

Proof. Lemma 3 yields

$$\sum_t U_t = \sum_{t > 2k} U_t + \sum_{t \leq 2k} U_t = \sum_{t > 2k} 2^{-t} \binom{t}{k} + \sum_{t \leq 2k} 2^{-t} \binom{t}{\lceil \frac{t}{2} \rceil}.$$

The terms in the first summation decrease by at least a factor of $3/4$ successively, so that we can upper bound it by $4 \binom{2k}{k}$. Stirling's approximation yields $\binom{t}{\lceil \frac{t}{2} \rceil} < \frac{O(1)}{\sqrt{t}}$ for all positive integers t . Hence we have

$$\sum_t U_t \leq \frac{4}{\sqrt{2k}} + \sum_{t \leq 2k} \frac{O(1)}{\sqrt{t}} = \Theta(\sqrt{k})$$

completing our proof.

6 Proof of Theorem 5

Proof of Theorem 5: Lemma 5 shows that ϕ_t is convex in $(n, n+1)$. Since ϕ_t is decreasing, it is right-convex at n . Theorem 10.1 in [8] shows ϕ_t is continuous in $(n, n+1)$; therefore to show convexity in $[n, n+1]$ we need ϕ_t to be left-continuous at $n+1$. Inductively, ϕ_{t+1} is decreasing and convex in $[n, n+1]$, and hence necessarily left-continuous at $n+1$. Along with (5), and a little arguing (omitted due to lack of space), ϕ_t is left-continuous at $n+1$. $\square$

Lemma 5. For every integer n and round t , ϕ_t is convex in $(n, n+1)$.

Proof. By backwards induction on t . The base case holds since ϕ_T is the loss function $\mathbf{1}(x \leq 2k - T)$. Assume ϕ_{t+1} is piecewise convex. Fix any integer $n \in \mathbb{Z}$. We have to show that ϕ_t is convex in $(n, n+1)$. Recall that, for non-integral points s , (5),(6) states that $\phi_t = \max\{H_{13}, H_{23}, H_{14}, H_{24}\}$ where $H_{ij}(s) = \frac{z_i \phi_{t+1}(s+z_j) - z_j \phi_{t+1}(s+z_i)}{z_i - z_j}$, and $(z_1, z_2, z_3, z_4) = (-1, n-s, n+1-s, +1)$. Checking that H_{14} and H_{23} are convex is straightforward. It turns out that H_{13} and H_{24} need not be convex. However, below we show that $\max\{H_{23}, H_{24}\}$ is

convex, and a very similar proof works for showing $\max\{H_{23}, H_{13}\}$ is convex. As the supremum of convex functions is convex (Theorem 5.5 [8]), and because ϕ_t can be written as $\phi_t = \max\{\max\{H_{23}, H_{13}\}, \max\{H_{23}, H_{24}\}, H_{14}\}$, we are done.

We begin by making our task of showing $\max\{H_{23}, H_{13}\}$ is convex a little easier. The next lemma shows that it suffices to show only local convexity, meaning every point in the domain has a neighborhood over which the function is convex. The proofs of this and other technical lemmas are given later.

Lemma 6. *A locally convex function on $(0, 1)$ is convex.*

We eliminate a degenerate case before proceeding. If $\phi_{t+1}(n) = 0$, then $\phi_{t+1}(s)$ is zero for $s \geq n$, and H_{24} ends up being the 0 function. In this case, $\max\{H_{23}, H_{24}\} = H_{23}$ is convex. So assume $\phi_{t+1}(n) > 0$.

If H_{24} were locally convex, then $\max\{H_{23}, H_{24}\}$ locally convex would follow immediately. Unfortunately, H_{24} may fail to be convex in some neighborhood. We instead show that in any neighborhood, either H_{24} is convex, or $H_{23} \geq H_{24}$, which suffices. The conditions for each fact to hold are given in the next two lemmas. With the degenerate case ruled out, we simplify the conditions by introducing functions $f, g : (0, 1) \rightarrow \mathbb{R}$

$$f(x) = \frac{\phi_{t+1}(n+1+x)}{\phi_{t+1}(n)}, \quad g(x) = \frac{f(x) - 1}{1+x}, \quad (12)$$

and we continuously extend them at 0.

Lemma 7. *Let f, g be as in (12). Then, $\max\{H_{23}, H_{24}\} = H_{23}$ at a point $(n+x)$ if $g(0) \geq g(x)$.*

Proof. Using ϕ_{t+1} is decreasing, $f(0) \leq \frac{\phi_{t+1}(n+1)}{\phi_{t+1}(n)}$. The rest is simple algebra. $\square$

Lemma 8. *Let f, g be as in (12). Then, the left derivative f^L of f exists, and H_{24} is convex in a neighborhood $(n+U)$ if $\forall x \in U, g(x) \leq f^L(x)$.*

The conditions in Lemma 7, 8 motivate the following definition.

Definition 1. *Let $f : [0, 1) \rightarrow \mathbb{R}$ have a left-derivative f^L at all points, and define $g : [0, 1) \rightarrow \mathbb{R}$ as in (12). Then f satisfies the dichotomy if around every point there is a neighborhood $U \subseteq [0, 1)$ for which at least one of the following holds:*

- $\forall x \in U : g(0) \geq g(x)$.
- $\forall x \in U : g(x) \leq f^L(x)$.

If f satisfies the dichotomy, then our proof is complete, since then either Lemma 7 or Lemma 8 will apply. In either case, $\max\{H_{23}, H_{24}\}$ is locally convex. A picture providing intuition for why this might happen is given in Figure 1.

Continuing with our proof, observe that (12) defines f to be a positive scaling of ϕ_{t+1} , which is convex by the inductive assumption. By construction f is continuous at 0 and hence convex in $[0, 1)$. By Theorem 10.1 of [8], convexity of f implies continuity in $(0, 1)$ as well. It turns out by the next lemma, that these properties are sufficient.

Lemma 9. *Every convex, continuous $f : [0, 1) \rightarrow \mathbb{R}$ satisfies the dichotomy.* $\square$

Using the procedure outlined above, we may conclude that F is convex over $[a, b]$. Since this holds for arbitrary $0 < a$ and $b < 1$, F is convex over $(0, 1)$. $\square$

Proof of Lemma 8. As noted in the proof of Lemma 5, f is convex, so that by Lemma 10, its left derivative exists. Next observe that function h , defined as¹ $h(x) \triangleq \frac{1+xf(x)}{1+x} = \frac{H_{24}(n+x)}{\phi_{t+1}(n)}$, is convex in a neighborhood U iff H_{24} is convex in $n + U$. For any points $0 < x < y < 1$, and convex combination $z = \lambda x + \mu y$, we get, after some algebra,

$$\lambda h(x) + \mu h(y) - h(z) = \frac{\lambda\mu(y-x)}{1+z}(g(y) - g(x)) + \frac{z(\lambda f(x) + \mu f(y) - f(z))}{1+z}.$$

The second term is non-negative since f is convex, and the first term is non-negative if g is non-decreasing. Hence h , and thus H_{24} , is convex in a region where g is not decreasing, which happens if $0 \leq g^L(x) = \frac{f^L(x)-g(x)}{1+x}$, i.e., $f^L(x) \geq g(x)$. $\square$

Proof of Lemma 9. We will need the following fact. For any points $x < y \in (0, 1)$

$$g(y) \text{ is a weighted average of } g(x) \text{ and } \frac{f(y) - f(x)}{y - x}. \quad (14)$$

We take cases to show that for every point x , there is a neighborhood U containing it where either $g(0) \geq g(y) \forall y \in U$, or $f^L(y) \geq g(y) \forall y \in U$.

case 1: $g(0) > g(x)$: By the continuity of f and hence g , we get $g(0) \geq g(y)$ for y in an interval containing x .

case 2: $g(0) < g(x)$: We have $g(x) < \frac{f(x)-f(0)}{x}$ (by (14)) $\leq f^L(x)$ (by (13)). By left continuity, $f^L(y) > g(y)$ in a left neighbourhood of x . For any $y > x$, $g(y) \leq \max\{g(x), \frac{f(y)-f(x)}{y-x}\}$ (by (14)) $\leq \max\{g(x), f^L(y)\}$ (by (13)) $= f^L(y)$, since f^L is increasing and $f^L(x) > g(x)$. Thus $f^L(y) \geq g(y)$ holds in a neighbourhood of x .

case 3: $g(0) = g(x)$: We have $g(x) = \frac{f(x)-f(0)}{x}$ (by (14)) $\leq f^L(x)$ (by (13)). If strict inequality holds, then we are done as in case 2. Otherwise we have $f^L(x) = \frac{f(x)-f(0)}{x}$. By Lemma 10, $f^L(x) = \sup_{y < x} \frac{f(y)-f(x)}{y-x}$, so that for any $y < x$, $\frac{f(x)-f(y)}{x-y} \leq \frac{f(x)-f(0)}{x}$. If strict inequality holds then, since $\frac{f(x)-f(0)}{x}$ is a weighted average of $\frac{f(x)-f(y)}{x-y}$ and $\frac{f(y)-f(0)}{y}$, we get $f^L(x) = \frac{f(x)-f(0)}{x} < \frac{f(y)-f(0)}{y} \leq f^L(y)$, a contradiction, since f convex implies f^L is non-decreasing. Hence $\frac{f(x)-f(y)}{x-y} = \frac{f(x)-f(0)}{x}$ for all $y < x$ and the segment of the curve f between $(0, x)$ is a straight line. It follows from (14) that $g(y) = \frac{f(y)-f(0)}{y}$, which in turn is equal to $f^L(y)$, for y in a left neighborhood around x ; since $f^L(x) \geq g(x)$ implies $f^L(y) \geq g(y)$ for y in a right neighborhood of x (as shown in case 2), we have $f^L(y) \geq g(y)$ in some neighborhood of x .

We have considered all cases. The proof follows. $\square$

¹ For geometric intuition about h , refer to Figure 1.

Acknowledgements. Thanks to Jake Abernethy and Yoav Freund for many helpful discussions. This research was supported by NSF grants IIS-0325500.

References

- [1] Abernethy, J., Langford, J., Warmuth, M.K.: Continuous experts and the binning algorithm. In: Lugosi, G., Simon, H.U. (eds.) COLT 2006. LNCS (LNAI), vol. 4005, pp. 544–558. Springer, Heidelberg (2006)
- [2] Cesa-Bianchi, N., Freund, Y., Haussler, D., Helmbold, D.P., Schapire, R.E., Warmuth, M.K.: How to use expert advice. *Journal of the Association for Computing Machinery* 44(3), 427–485 (1997)
- [3] Cesa-Bianchi, N., Freund, Y., Helmbold, D.P., Warmuth, M.K.: On-line prediction and conversion strategies. *Machine Learning* 25, 71–110 (1996)
- [4] Freund, Y.: Boosting a weak learning algorithm by majority. *Information and Computation* 121(2), 256–285 (1995)
- [5] Freund, Y., Oppen, M.: Continuous drifting games. *Journal of Computer and System Sciences*, 113–132 (2002)
- [6] 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)
- [7] Littlestone, N., Warmuth, M.K.: The weighted majority algorithm. *Information and Computation* 108, 212–261 (1994)
- [8] Rockafellar, R.T.: *Convex Analysis*. Princeton University Press, Princeton (1970)
- [9] Schapire, R.E.: Drifting games. *Machine Learning* 43(3), 265–291 (2001)
- [10] Spencer, J.: Ulam’s searching game with a fixed number of lies. *Theoret. Comput. Sci.* 95(2), 307–321 (1992)

Entropy Regularized LPBoost

Manfred K. Warmuth¹, Karen A. Glocer¹, and S.V.N. Vishwanathan²

¹ Computer Science Department
University of California, Santa Cruz
CA 95064, U.S.A

{manfred,kag}@cse.ucsc.edu
² NICTA, Locked Bag 8001
Canberra ACT 2601, Australia
SVN.Vishwanathan@nicta.com.au

Abstract. In this paper we discuss boosting algorithms that maximize the soft margin of the produced linear combination of base hypotheses. LPBoost is the most straightforward boosting algorithm for doing this. It maximizes the soft margin by solving a linear programming problem. While it performs well on natural data, there are cases where the number of iterations is linear in the number of examples instead of logarithmic.

By simply adding a relative entropy regularization to the linear objective of LPBoost, we arrive at the Entropy Regularized LPBoost algorithm for which we prove a logarithmic iteration bound. A previous algorithm, called SoftBoost, has the same iteration bound, but the generalization error of this algorithm often decreases slowly in early iterations. Entropy Regularized LPBoost does not suffer from this problem and has a simpler, more natural motivation.

1 Introduction

In the boosting by sampling setting, we are given a training set of $\pm$ labeled examples and an oracle for producing base hypotheses that are typically not very good. A boosting algorithm is able to find a linear combination of hypotheses that is a better classifier than the individual base hypotheses. To achieve this, the boosting algorithms maintain a distribution on the examples. At each iteration, this distribution is given to the oracle, which must return a base hypothesis that has a certain weak guarantee w.r.t. the current distribution on the examples. The algorithm then redistributes the weight on the examples so that more weight is put on the harder examples. In the next iteration, the oracle again provides a base hypothesis with the same weak guarantee for the new distribution and incorporates the obtained base hypothesis into the linear combination, and so forth. When the algorithm stops, it outputs a linear combination of the base hypotheses obtained in all iterations.

When the data is linearly separable, then maximizing the margin is a provably effective proxy for low generalization error [20] and in the inseparable case, maximizing the soft margin is a more robust choice. The soft margin can be maximized directly with a linear program, and this is precisely the approach taken

by a variant LPBoost [9, 16, 3]. Unless otherwise specified, the name LPBoost stands for this variant. While this algorithm has been shown to perform well in practice, no iteration bounds are known for it. As a matter of fact, the number of iterations required by LPBoost can be linear in the number of examples [24]. So what is the key to designing boosting algorithms with good iteration bounds? Clearly, linear programming is not enough.

To answer this question, let us take a step back and observe that in boosting there are two sets of dual weights/variables: the weights on the fixed set of examples and the weights on the set of currently selected hypotheses. In our case, both sets of weights are probability distributions. It has been shown that AdaBoost determines the weights on the hypotheses by minimizing a sum of exponentials [7]. Many boosting algorithms are motivated by modifying this type of objective function for determining the weights of the hypotheses [4]. Alternatively AdaBoost can be seen as minimizing a relative entropy to the current distribution subject to the linear constraint that the edge of the last hypothesis is nonnegative [11, 12]. All boosting algorithms whose iteration bounds are logarithmic in the number of examples are motivated by either optimizing a sum of exponentials in the hypothesis domain or a cost function that involves a relative entropy in the example domain. This line of research culminated in the boosting algorithm SoftBoost [24], which requires $O(\frac{1}{\epsilon^2} \ln \frac{N}{\nu})$ iterations to produce a linear combination of base hypotheses whose soft margin is within ϵ of the maximum minimum soft margin. Here $\nu \in [1, N]$ is the trade-off parameter for the soft margin.¹ More precisely, SoftBoost minimizes the relative entropy to the initial distribution subject to linear constraints on the edges of all the base hypotheses obtained so far. The upper bound on the edges is gradually decreased, and this leads to the main problem with SoftBoost: the generalization error decreases slowly in early iterations.

Once the relative entropy regularization was found to be crucial in the design of boosting algorithms, a natural algorithm emerged: simply add $\frac{1}{\eta}$ times the relative entropy to the initial distribution to the maximum soft edge objective of LPBoost and minimize the resulting sum. We call this algorithm *Entropy Regularized LPBoost*. A number of similar algorithms (such as ν -Arc [16]) were discussed in Gunnar Rätsch's Ph.D. thesis [14], but no iteration bounds were proven for them even though they were shown to have good experimental performance. Most recently, a similar boosting algorithm was considered by [18, 19] based on the Lagrangian dual of the optimization problem that motivates the Entropy Regularized LPBoost algorithm. However the $O(\frac{1}{\epsilon^2} \ln N)$ iteration bound proven for the recent algorithm is rather weak. In this paper, we prove an $O(\frac{1}{\epsilon^2} \ln \frac{N}{\nu})$ iteration bound for Entropy Regularized LPBoost. This bound is similar to the lower bound of $O(\frac{\ln N}{g^2})$ for boosting algorithms [5] for the hard margin case, where g is the minimum guaranteed edge of the weak learner. In related work, the same bound was proved for another algorithm that involves a tradeoff with the relative entropy [21]. This algorithm, however, belongs to

¹ An earlier version of this algorithm called TotalBoost dealt with the separable case and maximized the hard margin [23].

the “corrective” family of boosting algorithms (which includes AdaBoost) that only update the weights on the examples based on the last hypothesis. LPBoost, SoftBoost and the new Entropy Regularized LPBoost are “totally corrective” in the sense that they optimize their weight based on all past hypothesis. While the corrective algorithms are always simpler and faster on an iteration by iteration basis, the totally corrective versions require drastically fewer iterations and, in our experiments, beat the corrective variants based on total computation time. Also the simple heuristic of cycling over the past hypotheses with a corrective algorithm to make it totally corrective is useful for quick prototyping but in our experience this is typically less efficient than direct implementations of the totally corrective algorithms based on standard optimization techniques.

Outline: In Section 2 we discuss the boosting setting and motivate the Entropy Regularized LPBoost algorithm by adding a relative entropy regularizer to a linear program formulated in the weight domain on the examples. We give the algorithm in Section 3 and discuss its stopping criterion. The dual of Entropy Regularized LPBoost’s minimization problem is given in Section 4, and our main result, the iteration bound, is covered in Section 5. Finally, Section 6 contains our experimental evaluation of Entropy Regularized LPBoost and its competitors. The paper concludes with an outlook and discussion in Section 7.

2 The Boosting Setting and LPBoost

In the boosting setting, we are given a set of N labeled training examples (x_n, y_n) , $n = 1 \dots N$, where the instances x_n are in some domain $\mathcal{X}$ and the labels $y_n \in \pm 1$. Boosting algorithms maintain a distribution $\mathbf{d}$ on the N examples, so $\mathbf{d}$ lies in the N dimensional probability simplex $\mathcal{S}^N$. Intuitively, the examples that are hard to classify are given more weight. In each iteration $t = 1, 2, \dots$ the algorithm gives the current distribution $\mathbf{d}^{t-1}$ to an oracle (a.k.a. the weak learning algorithm), which returns a new base hypothesis $h^t : \mathcal{X} \rightarrow [-1, 1]$ from some base hypothesis class $\mathcal{H}$. The hypothesis returned by the oracle comes with a certain guarantee of performance. This guarantee will be discussed in Section 3.

One measure of the performance of a base hypothesis h^t w.r.t. the current distribution $\mathbf{d}^{t-1}$ is its *edge*, which is defined as $\sum_{n=1}^N d_n^{t-1} y_n h^t(x_n)$. When the range of h^t is ± 1 instead of the interval $[-1, 1]$, then the edge is just an affine transformation of the weighted error ϵ_{h^t} of hypothesis h^t . A hypothesis that predicts perfectly has edge 1 and a hypothesis that always predicts incorrectly has edge -1 , while a random hypothesis has edge approximately 0. The higher the edge, the more useful the hypothesis is for classifying the training examples. The edge of a set of hypotheses is defined as the maximum edge of the set.

It is convenient to define an N -dimensional vector $\mathbf{u}^t$ that combines the base hypothesis h^t with the labels y_n of the N examples: $u_n^t := y_n h^t(x_n)$. With this notation, the edge of the hypothesis h^t becomes $\mathbf{u}^t \cdot \mathbf{d}^{t-1}$.

After a hypothesis h^t is received, the algorithm must update its distribution $\mathbf{d}^{t-1}$ on the examples using $\mathbf{u}^t$. The final output of the boosting algorithm is always a convex combination of base hypotheses $f_{\mathbf{w}}(x_n) = \sum_{q=1}^T w_q h^q(x_n)$, where

Algorithm 1. Entropy Regularized LPBoost

-
1. **Input:** $S = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$, accuracy parameter $\epsilon > 0$, regularization parameter $\eta > 0$, and smoothness parameter $\nu \in [1, N]$.
 2. **Initialize:** $\mathbf{d}^0$ to the uniform distribution.
 3. **Do for** $t = 1, \dots$
 - (a) Send $\mathbf{d}^{t-1}$ to oracle and obtain hypothesis h^t .
Set $u_n^t = y_n h^t(\mathbf{x}_n)$
Assume $\mathbf{u}^t \cdot \mathbf{d}^{t-1} \geq g$, where g need not be known.
 - (b) Set $\delta^t = \min_{q=1 \dots t} P^q(\mathbf{d}^{q-1}) - P^{t-1}(\mathbf{d}^{t-1})$,
where $P^t(\mathbf{d}) = \max_{q=1,2,\dots,t} \mathbf{u}^q \cdot \mathbf{d} + \frac{1}{\eta} \Delta(\mathbf{d}, \mathbf{d}^0)$.
 - (c) **If** $\delta^t \leq \epsilon/2$ **then** set $T = t - 1$ and break.
 - (d) **Else** Update the distribution to $[\mathbf{d}^t, \gamma^t] = \underset{\mathbf{d}, \gamma}{\operatorname{argmin}} \gamma + \frac{1}{\eta} \Delta(\mathbf{d}, \mathbf{d}^0)$,
s.t. $\mathbf{u}^q \cdot \mathbf{d} \leq \gamma$, for $1 \leq q \leq t$, and $d_n \leq 1/\nu$, for $1 \leq n \leq N$, $\sum_n d_n = 1$.
This can be done using sequential quadratic programming.
 4. **Output:** $f_{\mathbf{w}}(\mathbf{x}) = \sum_{q=1}^T w_q h^q(\mathbf{x})$, where the coefficients w_q maximize the soft margin over the hypothesis set $\{h^1, \dots, h^T\}$ using the LP soft margin problem (2).
-

w_q is the coefficient of the hypothesis h^q added at iteration q . Ideally, all examples should be on the correct side of the hyperplane defined by the linear combination $f_{\mathbf{w}}$. In other words, for all examples (x_n, y_n) , we want $y_n f_{\mathbf{w}}(x_n) > 0$. The *hard margin* of an example (x_n, y_n) measures by how much an example is on the right side and is defined as $y_n f_{\mathbf{w}}(x_n)$. The hard margin of a set of examples is taken to be the minimum margin of the set. When the sign of the convex combination is consistent with the labels, the examples are separated by the hyperplane $f_{\mathbf{w}}$, and this margin is positive. Note that edges are linear in the distribution over the examples and margins are linear in the distribution over the current set of hypotheses.

There is a simple linear programming problem that defines a basic boosting algorithm: update to any distribution that minimizes the maximum edge of the t hypotheses seen so far. That is, $\mathbf{d}^t \in \underset{\mathbf{d} \in \mathcal{S}^N}{\operatorname{argmin}} \max_{q=1,2,\dots,t} \mathbf{u}^q \cdot \mathbf{d}$. The resulting boosting algorithm is the hard margin version of LPBoost ([9]). By linear programming duality, the minimum-maximum edge equals the maximum-minimum margin:

$$\min_{\mathbf{d} \in \mathcal{S}^N} \max_{q=1,2,\dots,t} \mathbf{u}^q \cdot \mathbf{d} = \max_{\mathbf{w} \in \mathcal{S}^t} \min_{n=1,2,\dots,N} \sum_{q=1}^t u_n^q w_q. \quad (1)$$

In the case when examples are not separable by a linear combination of the base hypotheses, then the hard margins are naturally replaced by the *soft margins*. The term “soft” here refers to a relaxation of the margin constraint. We now allow examples to lie below the margin but penalize them linearly via slack variables ψ_n . The resulting optimization problem (2) is again a linear program,

$$\max_{\mathbf{w} \in \mathcal{S}^t, \psi \geq 0} \min_{n=1,2,\dots,N} \left(\sum_{q=1}^t u_n^q w_q + \psi_n \right) - \frac{1}{\nu} \sum_{n=1}^N \psi_n, \quad (2)$$

where the trade-off parameter ν is chosen in the range $[1..N]$. Adding slack variables in the hypothesis domain (2) gives rise to the capping constraints $\mathbf{d} \leq \frac{1}{\nu} \mathbf{1}$ in the dual example domain (see e.g. [16, 3] for an early discussion of capping):

$$\min_{\mathbf{d} \in \mathcal{S}^N, \mathbf{d} \leq \frac{1}{\nu} \mathbf{1}} \max_{q=1,2,\dots,t} \mathbf{u}^q \cdot \mathbf{d}. \quad (3)$$

This suggests that boosting algorithms that cap the weight on the examples do well on inseparable data for the same reasons as algorithms that maximize the soft margin.² By linear programming duality, the value at iteration $t = 1, 2, \dots$ of (2) is equal to the value of its dual (3), and we will denote this value by P_{LP}^t . When $t = 0$, then the maximum is over an empty set of hypotheses and is defined as -1 (i.e. $P_{\text{LP}}^0 := -1$).

3 The Entropy Regularized LPBoost Algorithm

In the minimization problem that motivates the main algorithm of this paper, *Entropy Regularized LPBoost*, a relative entropy is added to the linear programming problem in the example domain (3). The relative entropy between distributions $\mathbf{d}$ and $\mathbf{d}^0$ is defined as $\Delta(\mathbf{d}, \mathbf{d}^0) = \sum_{n=1}^N d_n \ln \frac{d_n}{d_n^0}$. The factor $1/\eta$ is a trade-off parameter between the relative entropy and the maximum edge. The modified mini-max problem is defined as follows:

$$\min_{\substack{\mathbf{d} \cdot \mathbf{1} = 1 \\ \mathbf{d} \leq \frac{1}{\nu} \mathbf{1}}} \max_{q=1,2,\dots,t} \underbrace{\mathbf{u}^q \cdot \mathbf{d} + \frac{1}{\eta} \Delta(\mathbf{d}, \mathbf{d}^0)}_{:= P^t(\mathbf{d})}. \quad (4)$$

Note that the constraint $\mathbf{d} \geq \mathbf{0}$ was dropped because the relative entropy is not defined for negative d_n , and therefore the constraints $\mathbf{d} \geq \mathbf{0}$ are enforced implicitly. The relative entropy term makes the objective function $P^t(\mathbf{d})$ strictly convex and therefore the min has a unique solution, which we denote as $\mathbf{d}^t$. We also define $P^t(\mathbf{d})$ when $t = 0$. In this case the maximum is over an empty set of hypotheses and is defined as -1 as before. Thus $P^0(\mathbf{d})$ becomes $-1 + \frac{1}{\eta} \Delta(\mathbf{d}, \mathbf{d}^0)$, which is minimized at $\mathbf{d}^0$ and therefore $P^0(\mathbf{d}^0) := -1$.

The Entropy Regularized LPBoost algorithm, shown in Algorithm 1 predicts at trial t with this distribution $\mathbf{d}^t$. Note that in the statement of the algorithm we reformulated (4) into an equivalent convex optimization problem. If the regularization term $\frac{1}{\eta} \Delta(\mathbf{d}, \mathbf{d}^0)$ is dropped from the optimization problem then this algorithm becomes the original LPBoost, whose solution is not unique and depends on the LP solver that is used.

² When $\nu = 1$, then the capping constraints in (3) are vacuous and this problem is equivalent to the l.h.s. of (1). Similarly, when $\nu = 1$, then the derivatives of the objective of the maximization problem (2) w.r.t. each slack variable ψ_n are all at most zero and $\boldsymbol{\psi} = \mathbf{0}$ is an optimal solution for the maximization over $\boldsymbol{\psi}$. This makes the slack variables disappear and (2) becomes the r.h.s. of (1).

In each iteration t , the boosting algorithm sends a distribution $\mathbf{d}^{t-1}$ to the oracle and the oracle returns a hypothesis h^t from the base hypotheses set $\mathcal{H}$. The returned hypothesis satisfies a certain quality assumption. The strongest assumption is that the oracle returns a hypothesis with maximum edge, i.e. h^t lies in $\operatorname{argmax}_{h \in \mathcal{H}} \mathbf{u}^h \cdot \mathbf{d}^{t-1}$. Iteration bounds for this oracle essentially follow from the work of [22] who use bundle methods to optimize functions of the form

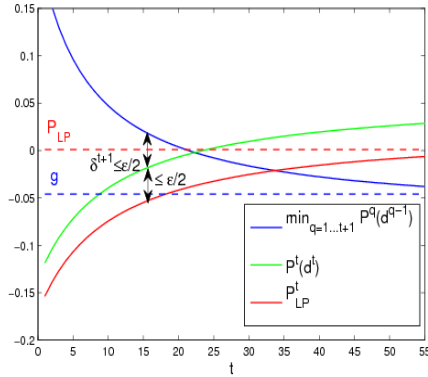
$$\min_{\substack{\mathbf{d} \leq \frac{1}{\nu} \mathbf{1} \\ \mathbf{d} \cdot \mathbf{1} = 1}} \frac{1}{\eta} \Delta(\mathbf{d}, \mathbf{d}_0) + \max_{h \in \mathcal{H}} \mathbf{u}^h \cdot \mathbf{d}.$$

They show that $O(\frac{\eta}{\epsilon})$ iterations suffice to get within $\epsilon/2$ of the optimum value of this optimization problem. However, the algorithm requires the computation of the gradient of $\max_{h \in \mathcal{H}} \mathbf{u}^h \cdot \mathbf{d}$ evaluated at the current distribution $\mathbf{d}^{t-1}$. The hypothesis returned by the above maximum edge oracle is such a gradient. Setting $\eta = \frac{\ln N}{\epsilon}$ and using a version of the below lemma gives the bound.

This strong oracle is also assumed for the recent corrective algorithm of [21]. In contrast, in this paper we follow [17, 23, 24] and only require a lower bound g on the edge of the hypothesis returned by the oracle. Iteration bounds for this weaker oracle are much harder to obtain.

Assumption on the weak learner:

We assume that given any distribution $\mathbf{d}^{t-1} \in \mathcal{S}^N$ on the examples, the oracle returns a hypothesis h^t with edge at least g . We call g the *guarantee* of the oracle. In other words, if $\mathbf{u}^t$ is the vector that combines the base hypothesis h^t with the labels of the examples, then $\mathbf{u}^t \cdot \mathbf{d}^{t-1} \geq g$. In a moment we will discuss the range of admissible values of g .



Stopping criterion: The algorithm monitors $\delta^t := \min_{q=1,2,\dots,t} P^q(\mathbf{d}^{q-1}) - P^{t-1}(\mathbf{d}^{t-1})$, and stops when $\delta^{T+1} \leq \epsilon/2$, for a predefined threshold $\epsilon > 0$.

Recall that the output of the algorithm is a linear combination $f_{\mathbf{w}}(\mathbf{x}) = \sum_{q=1}^T w_q h^q(\mathbf{x})$, where the coefficients w_q are an optimal solution of the LP soft margin problem (3) over the hypothesis set $\{h^1, \dots, h^T\}$. We first bound the number of iterations T needed before the value P_{LP}^T of this optimization problem gets within ϵ of the guarantee g . All relevant quantities used in the proof are depicted in Figure 1.

Lemma 1. *If $\eta \geq \frac{2}{\epsilon} \ln \frac{N}{\nu}$ in (4), then $\delta^{T+1} \leq \epsilon/2$ implies $g - P_{LP}^T \leq \epsilon$, where g is the guarantee of the oracle.*

Fig. 1. Depiction of the stopping criterion: $\delta^{T+1} \leq \epsilon/2$ implies $g - P_{LP}^T \leq \epsilon$

Proof. Since $\Delta(\mathbf{d}, \mathbf{d}^0) \leq \ln \frac{N}{\nu}$ and $\eta \geq \frac{2}{\epsilon} \ln \frac{N}{\nu}$, we have $\frac{1}{\eta} \Delta(\mathbf{d}, \mathbf{d}^0) \leq \epsilon/2$ and

$$P^T(\mathbf{d}^T) \leq P_{\text{LP}}^T + \epsilon/2. \quad (5)$$

On the other hand, from the fact that $\Delta(\mathbf{d}, \mathbf{d}^0) \geq 0$ and the assumption on the weak learner, we know that

$$g \leq \min_{q=1,2,\dots,T+1} \mathbf{u}^q \cdot \mathbf{d}^{q-1} \leq \min_{q=1,2,\dots,T+1} P^q(\mathbf{d}^{q-1}).$$

Subtracting $P^T(\mathbf{d}^T)$ and using the stopping criterion we have

$$g - P^T(\mathbf{d}^T) \leq \min_{q=1,2,\dots,T+1} P^q(\mathbf{d}^{q-1}) - P^T(\mathbf{d}^T) = \delta^{T+1} \leq \epsilon/2.$$

Adding (5) to the above yields $g \leq P_{\text{LP}}^T + \epsilon$. $\square$

Now we must consider the range of the guarantee g that an oracle can achieve. Because $\mathbf{u}^t \in [-1, +1]^N$, it is easy to achieve $g \geq -1$. We claim that the maximum achievable guarantee is $g = P_{\text{LP}}$, where P_{LP} is defined as the value of (3) w.r.t. the entire hypothesis set $\mathcal{H}$ from which oracle can choose. That is,

$$P_{\text{LP}} := \min_{\mathbf{d} \in \mathcal{S}^N} \sup_{\mathbf{d} \leq \frac{1}{\nu} \mathbf{1}} \sup_{h \in \mathcal{H}} \mathbf{u}^h \cdot \mathbf{d}$$

and therefore, for any distribution $\mathbf{d}$ such that $\mathbf{d} \leq \frac{1}{\nu} \mathbf{1}$, we have $\max_{h \in \mathcal{H}} \mathbf{u}^h \cdot \mathbf{d} \geq P_{\text{LP}}$. Thus, there always exists a hypothesis in $\mathcal{H}$ with edge at least P_{LP} . Also, for any optimal distribution that realizes the value P_{LP} , there is no hypothesis of edge strictly greater than P_{LP} .

For computational reasons, the guarantee g of an oracle may be less than P_{LP} , and therefore we formulate our algorithm and iteration bound for an oracle w.r.t. any guarantee $g \in [-1, P_{\text{LP}}]$. It should be emphasized that our algorithm does not need to know the guarantee g achieved by the oracle.

The line P_{LP} is depicted in Figure 1. The sequence $\langle P_{\text{LP}}^t \rangle$ approaches this line from below and the sequence $\langle \min_{q=1\dots t+1} P^q(\mathbf{d}^{q-1}) \rangle$ approaches the line g from above. When $g < P_{\text{LP}}$, as shown in Figure 1, then both sequences cross, and when $g = P_{\text{LP}}$, then both sequences get arbitrarily close.

Note that the strong oracle discussed earlier which always returns a hypothesis in $\mathcal{H}$ of *maximum edge* w.r.t. the provided distribution clearly has the maximum guarantee P_{LP} . Indeed, on many data sets, this more computationally expensive oracle converges faster than an oracle that only returns a hypothesis of edge P_{LP} (see [17] for an experimental comparison), even though there are no improved iteration bounds known for this stronger oracle.

Our algorithm produces its final linear combination based on the soft margin linear programming problem (3). Alternatively, it could produce a final weight vector $\mathbf{w}$ based on the dual of the regularized minimum-maximum edge problem (4) given in the next section. When $\eta \geq \frac{2}{\epsilon} \ln \frac{N}{\nu}$, then the regularization term $\frac{1}{\eta} \Delta(\mathbf{d}, \mathbf{d}^0)$ is at most $\epsilon/2$, implying that the values of the regularized problems in the example domain are always at most $\epsilon/2$ larger than the corresponding

unregularized problems. Therefore computing the final $\mathbf{w}$ in Entropy Regularized LPBoost via the dual of the regularized problem is also a reasonable choice. In our experiments (not shown) this did not affect the generalization error of the algorithm.

4 The Dual Optimization Problem of Entropy Regularized LPBoost

In this section we compute the Lagrangian dual of (4) and discuss the dual relationship between the the problem of optimizing the distribution on the examples versus optimizing the distribution on the current set of hypotheses. We state the following lemma without proof; the proof is standard and follows from standard arguments (see e.g. [2]).

Lemma 2. *The Lagrangian dual of (4) is*

$$\max_{\mathbf{w}} \hat{\Theta}^t(\mathbf{w}, \psi), \quad \text{s.t. } \mathbf{w} \geq \mathbf{0}, \mathbf{w} \cdot \mathbf{1} = 1, \psi \geq 0,$$

$$\text{where } \hat{\Theta}^t(\mathbf{w}, \psi) := -\frac{1}{\eta} \ln \sum_{n=1}^N d_n^0 \exp(-\eta(\sum_{q=1}^t u_n^q w_q + \psi_n)) - \frac{1}{\nu} \sum_{n=1}^N \psi_n.$$

The optimal solution $\mathbf{d}^t$ of (4) can be expressed in terms of the dual variables $\mathbf{w}^t$ and ψ^t as follows:

$$d_n^t := \frac{d_n^0 \exp(-\eta(\sum_{q=1}^t u_n^q w_q^t + \psi_n^t))}{\sum_{n'} d_{n'}^0 \exp(-\eta(\sum_{q=1}^t u_{n'}^q w_q^t + \psi_{n'}^t))}. \quad (6)$$

Furthermore, the value of the primal is equal to the value of the dual. Also, for the optimal primal solution $\mathbf{d}^t$ and optimal dual solution $\mathbf{w}^t$,

$$\sum_{q=1}^t w_q^t \mathbf{u}^q \cdot \mathbf{d}^t = \max_{q=1,2,\dots,t} \mathbf{u}^q \cdot \mathbf{d}^t.$$

Note that $\hat{\Theta}^t(\mathbf{w}, \psi)$ is a “smoothed” minimum soft margin of the examples for the linear combination $\mathbf{w}$ of the first t hypotheses. As $\eta \rightarrow \infty$, this margin becomes the minimum soft margin. Similarly, $P^t(\mathbf{d})$ is the smoothed maximum edge of the first t hypothesis when the distribution on the examples $\mathbf{d}$ lies in the capped probability simplex. Again as $\eta \rightarrow \infty$, this edge becomes the maximum edge over the capped simplex. Smoothing is done by different means in the primal versus the dual domain. In the primal it is done by adding one over η times a relative entropy to the max. In the dual, the log partition function smoothes the minimum soft margin.

A smoothing effect can also be seen in the distribution $\mathbf{d}^t$ over the examples. Whereas LPBoost puts its entire weight onto the examples with minimum soft margin w.r.t. the current hypothesis set $\{h^1, \dots, h^t\}$, Entropy Regularized LPBoost spreads the weight to examples with higher soft margins by taking the

soft min of these margins. The degree of the smoothing is controlled by η . As $\eta \rightarrow \infty$, Entropy Regularized LPBoost reverts to an instantiation of LPBoost (i.e. all weight is put on examples with minimum soft margin).

5 Iteration Bound for Entropy Regularized LPBoost

This section contains our main result. For clarity, the number of iterations corresponds to the number of hypotheses incorporated into the final linear combination, rather than calls to the oracle. The number of calls to the oracle is $T+1$ but the number of hypotheses in the final linear combination is T . In other words, the hypothesis h^{T+1} obtained in the last call to the oracle is discarded.

Our first technical lemma bounds the increase $P^t(\mathbf{d}^t) - P^{t-1}(\mathbf{d}^{t-1})$ in terms of a quadratic term.

Lemma 3. *If $\eta \geq \frac{1}{2}$, then $P^t(\mathbf{d}^t) - P^{t-1}(\mathbf{d}^{t-1}) \geq \frac{1}{8\eta}(P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}))^2$.*

Proof. First observe that $P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}) = \max_{q=1,2,\dots,t} \mathbf{u}^q \cdot \mathbf{d}^{t-1} - \max_{q=1,2,\dots,t-1} \mathbf{u}^q \cdot \mathbf{d}^{t-1}$. Clearly the first max is at least as large as the second. If both are the same, then the lemma trivially holds because $P^t(\mathbf{d}^{t-1}) = P^{t-1}(\mathbf{d}^{t-1})$. If $P^t(\mathbf{d}^{t-1}) > P^{t-1}(\mathbf{d}^{t-1})$, then the first max equals $\mathbf{u}^t \cdot \mathbf{d}^{t-1}$. We can also rewrite the second max by invoking Lemma 2 with $t-1$ instead of t , obtaining

$$P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}) = \mathbf{u}^t \cdot \mathbf{d}^{t-1} - \sum_{q=1}^{t-1} w_q^{t-1} \mathbf{u}^q \cdot \mathbf{d}^{t-1} := (\mathbf{u}^t - \underbrace{\sum_{q=1}^{t-1} w_q^{t-1} \mathbf{u}^q}_{:=\mathbf{x}}) \cdot \mathbf{d}^{t-1}.$$

We still need to show that when $\mathbf{x} \cdot \mathbf{d}^{t-1} \geq 0$, $P^t(\mathbf{d}^t) - P^{t-1}(\mathbf{d}^{t-1}) \geq \frac{1}{8\eta}(\mathbf{x} \cdot \mathbf{d}^{t-1})^2$.

By Lemma 2, the value of the optimization problem defining Entropy Regularized LPBoost, $P^t(\mathbf{d})$, equals the value of its dual problem. We begin by lower bounding the increase of this value between successive iterations. Let $\mathbf{w}^t$ and $\boldsymbol{\psi}^t$ denote optimal parameters for the dual problem at iteration t . Because the dual is a maximization problem, $\hat{\Theta}^t(\mathbf{w}^t, \boldsymbol{\psi}^t) \geq \hat{\Theta}^t(\mathbf{w}, \boldsymbol{\psi})$ for any suboptimal $\mathbf{w} \in \mathcal{P}^t$ and $\boldsymbol{\psi} \geq \mathbf{0}$. For our lower bound on the value we replace $\boldsymbol{\psi}^t$ by the suboptimal previous value $\boldsymbol{\psi}^{t-1}$ and $\mathbf{w}^t$ by $\mathbf{w}^t(\alpha) = (1-\alpha) \begin{bmatrix} \mathbf{w}^{t-1} \\ 0 \end{bmatrix} + \alpha \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}$, where $\alpha \in [0, 1]$:

$$\begin{aligned} P^t(\mathbf{d}^t) - P^{t-1}(\mathbf{d}^{t-1}) &= \hat{\Theta}^t(\mathbf{w}^t, \boldsymbol{\psi}^t) - \hat{\Theta}^{t-1}(\mathbf{w}^{t-1}, \boldsymbol{\psi}^{t-1}) \\ &\geq \hat{\Theta}^t(\mathbf{w}^t(\alpha), \boldsymbol{\psi}^{t-1}) - \hat{\Theta}^{t-1}(\mathbf{w}^{t-1}, \boldsymbol{\psi}^{t-1}) \\ &= -\frac{1}{\eta} \ln \sum_{n=1}^N d_n^0 \exp \left(-\eta \sum_{q=1}^{t-1} u_n^q w_q^{t-1} - \eta \alpha u_n^t + \eta \alpha \sum_{q=1}^{t-1} u_n^q w_q^{t-1} - \eta \psi_n^{t-1} \right) \\ &\quad + \frac{1}{\eta} \ln \sum_{n=1}^N d_n^0 \exp \left(-\eta \sum_{q=1}^{t-1} u_n^q w_q^{t-1} - \eta \psi_n^{t-1} \right) \end{aligned}$$

$$\stackrel{(6)}{=} -\frac{1}{\eta} \ln \sum_{n=1}^N d_n^{t-1} \exp \left(-\eta \alpha \underbrace{\left(u_n^t - \sum_{q=1}^{t-1} u_n^q w_q^{t-1} \right)}_{:=x_n} \right).$$

This holds for any $\alpha \in [0, 1]$. Since $x_n \in [-2, 2]$, then $\exp(x_n) \leq \frac{2+x_n}{4} \exp(-2\eta\alpha) + \frac{2-x_n}{4} \exp(2\eta\alpha)$ and this lets us lower bound the above as

$$\begin{aligned} & -\frac{1}{\eta} \ln \left(\frac{2 + \mathbf{x} \cdot \mathbf{d}^{t-1}}{4} \exp(-2\eta\alpha) + \frac{2 - \mathbf{x} \cdot \mathbf{d}^{t-1}}{4} \exp(2\eta\alpha) \right) \\ & = 2\alpha - \frac{1}{\eta} \ln \left(1 - \frac{2 - \mathbf{x} \cdot \mathbf{d}^{t-1}}{4} (1 - \exp(4\eta\alpha)) \right) \end{aligned}$$

By applying the following inequality from [10]

$$\forall c \in [0, 1] \text{ and } r \in \mathbb{R} : -\ln(1 - c(1 - e^r)) \geq -cr - \frac{r^2}{8},$$

the above can be lower bounded by $2\alpha - \frac{2 - \mathbf{x} \cdot \mathbf{d}^{t-1}}{4} 4\alpha - \frac{16\eta\alpha^2}{8}$. This is maximized at $\alpha = \frac{\mathbf{x} \cdot \mathbf{d}^{t-1}}{4\eta}$ which lies in $[0, 1]$ because $\mathbf{x} \cdot \mathbf{d}^{t-1} \in [0, 2]$ and $\eta \geq \frac{1}{2}$. Plugging this α into the above, we get $\frac{(\mathbf{x} \cdot \mathbf{d}^{t-1})^2}{8\eta}$ as desired. $\square$

Recall the definition of δ^t that we monitor in our proof:

$$\delta^t = \min_{q=1,2,\dots,t} P^q(\mathbf{d}^{q-1}) - P^{t-1}(\mathbf{d}^{t-1}).$$

As done in [22], we now prove a quadratic recurrence for δ^t .

Lemma 4. *If $\eta \geq 1/2$ and $\delta^t \geq 0$, then $\delta^t - \delta^{t+1} \geq \frac{(\delta^t)^2}{8\eta}$, for $1 \leq t \leq T$.*

Proof. We prove this recurrence by bounding the inequality of the previous lemma from above and below. We upper bound the l.h.s. via

$$P^t(\mathbf{d}^t) - P^{t-1}(\mathbf{d}^{t-1}) \leq \underbrace{\min_{1 \leq q \leq t} P^q(\mathbf{d}^{q-1}) - P^{t-1}(\mathbf{d}^{t-1})}_{\delta^t} - \underbrace{\min_{1 \leq q \leq t+1} P^q(\mathbf{d}^{q-1}) - P^t(\mathbf{d}^t)}_{\delta^{t+1}}.$$

To lower bound the r.h.s. of the same inequality, first observe that

$$P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}) \geq \min_{1 \leq q \leq t} P^q(\mathbf{d}^{q-1}) - P^{t-1}(\mathbf{d}^{t-1}) = \delta^t,$$

Since we assumed $\delta^t \geq 0$, we can lower bound the r.h.s. as

$$\frac{(P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}))^2}{8\eta} \geq \frac{(\delta^t)^2}{8\eta}.$$

$\square$

The lemma requires $\delta^t \geq 0$. The stopping criterion actually assures that $\delta^t > \frac{\epsilon}{2}$, for $1 \leq t \leq T$. Instead of using a recurrence relation, the standard approach would be to show that the value of the underlying optimization problem drops by at least a constant. See e.g. [23, 24] for examples for this type of proof. More precisely, in our case we have

$$P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}) \geq \frac{(\delta^t)^2}{8\eta} \geq \frac{1}{32\eta}\epsilon^2.$$

Unfortunately, for our solution to get ϵ -close to the guarantee g , we need that η is inversely proportional to ϵ and therefore this proof technique only yields the iteration bound of $O(\frac{1}{\epsilon^3} \ln \frac{N}{\nu})$. We shall now see that our recurrence method leads to an improved iterations bound of $O(\frac{1}{\epsilon^2} \ln \frac{N}{\nu})$, which is optimal in the hard margin case (when $\nu = 1$).

Lemma 5. *Let $\langle \delta^1, \delta^2, \dots \rangle$ be a sequence of non-negative numbers satisfying the following recurrence, for $t \geq 1$: $\delta^t - \delta^{t+1} \geq c(\delta^t)^2$, where $c > 0$ is a positive constant. Then for all integers $t \geq 1$,*

$$\frac{1}{c(t-1 + \frac{1}{\delta^1 c})} \geq \delta^t.$$

This is Sublemma 5.4 of [1] which is easily proven by induction.

Theorem 1. *If $\eta = \max(\frac{2}{\epsilon} \ln \frac{N}{\nu}, \frac{1}{2})$ in (4), then Entropy Regularized LPBoost terminates in*

$$T \leq \max(\frac{32}{\epsilon^2} \ln \frac{N}{\nu}, \frac{8}{\epsilon})$$

iterations with a final convex combination of hypotheses for which $g - P_{LP}^t \leq \epsilon$.

Proof. Since $\eta \geq \frac{1}{2}$, we can apply Lemma 4 with $c = \frac{1}{8\eta}$. This give us $\delta^t \leq \frac{8\eta}{t-1 + \frac{8\eta}{\delta^1}}$, which can be rearranged to $t \leq \frac{8\eta}{\delta^t} - \frac{8\eta}{\delta^1} + 1 < \frac{8\eta}{\delta^t}$, since $\eta \geq \frac{1}{2}$ and $0 \leq \delta^1 \leq 1$. By the stopping criterion, $\delta^T > \epsilon/2$ and $\delta^{T+1} \leq \epsilon/2$. Thus the above inequality implies that

$$T < \frac{8\eta}{\delta^T} \leq \frac{16\eta}{\epsilon}. \quad (7)$$

By Lemma 1, $\delta^{T+1} \leq \epsilon/2$ implies tolerance ϵ if $\eta \geq \frac{2}{\epsilon} \ln \frac{N}{\nu}$. Plugging $\eta = \max(\frac{2}{\epsilon} \ln \frac{N}{\nu}, \frac{1}{2})$ into the above inequality results in $T \leq \max(\frac{32}{\epsilon^2} \ln \frac{N}{\nu}, \frac{8}{\epsilon})$. Because the aborted iteration $T + 1$ is not counted, we arrive at the iteration bound of the theorem. $\square$

Note that $\frac{2}{\epsilon} \ln \frac{N}{\nu} \geq 1/2$ iff $\epsilon \leq 4 \ln \frac{N}{\nu}$. As pointed out before, when $\eta \rightarrow \infty$ then Entropy Regularized LPBoost morphs into a version of LPBoost. Notice that the iteration bound (7) is linear in η . Therefore as $\eta \rightarrow \infty$, the iteration bound becomes vacuous.

6 Experimental Evaluation

All of our experiments utilize data from ten benchmark data sets derived from the UCI benchmark repository as previously used in [15]. We compared the soft margin algorithms LPBoost, Entropy Regularized LPBoost, and SoftBoost – with AdaBoost, LogitBoost [8], and BrownBoost[6]. AdaBoost³ was chosen as a comparator because it is the most common boosting algorithm overall. LogitBoost and BrownBoost were chosen because they are well known and designed for inseparable data.

We used RBF networks as the base learning algorithm.⁴ The data comes in 100 predefined splits into training and test sets. For the soft margin algorithms we set $\epsilon = 0.01$. Then for each each of the splits we used 5-fold cross-validation to select the optimal free parameters for each of the algorithms. Finally, in Entropy Regularized LPBoost, η was set to $\frac{2}{\epsilon} \ln \frac{N}{\nu}$. This leads to 100 estimates of the generalization error for each method and data set.

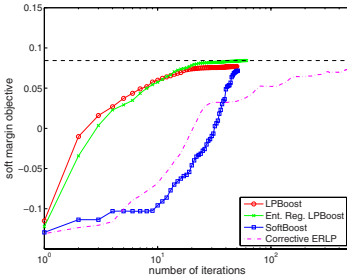


Fig. 2. Soft margin objective vs. the number of iterations on a single run for the Banana data set with $\epsilon = 0.01$ and $\nu/N = 0.1$. In Entropy Regularized LPBoost, $\eta = \frac{2}{\epsilon} \ln \frac{N}{\nu}$. Note that LPBoost is virtually indistinguishable from Entropy Regularized LPBoost, and SoftBoost’s margin begins increasing much later than the others. Also, the corrective algorithm converges more slowly than the totally corrective version.

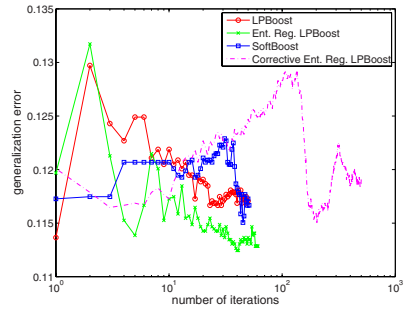


Fig. 3. Generalization error for the same set of algorithms. The corrective algorithm converges more slowly than the totally corrective algorithms. To depict them in the same figure, we use a logarithmic scale of the iteration number in both plots. Note that eventually the generalization error of the corrective algorithm ends up in the same ball-park.

The first step in our empirical analysis of Entropy Regularized LPBoost, is to verify that it actually maximizes the soft margin. Figure 2 shows the soft margin vs. number of iterations for LPBoost, Entropy Regularized LPBoost, and SoftBoost for a single run of the Banana data set from the UCI Benchmark

³ In AdaBoost, the parameter α_t was chosen to minimize $\sum_n d_n^t \exp(-\alpha u_n^t)$.

⁴ The data is from <http://theoval.cmp.uea.ac.uk/~gcc/matlab/index.shtml>. The RBF networks were obtained from the authors of [15], including the hyper-parameter settings for each data set.

Table 1. Generalization error estimates and standard deviations for ten UCI benchmark data sets. As expected, there is no statistically significant difference between LPBoost, SoftBoost, and Entropy Regularized LPBoost, but they outperform AdaBoost and BrownBoost on most data sets.

	AdaBoost	LogitBoost	BrownBoost	LPBoost	SoftBoost	ERLPBoost
Banana	13.3± 0.7	12.4±0.6	12.9±0.7	11.1± 0.6	11.1± 0.5	11.1±0.6
B.Cancer	32.1± 3.8	30.2±4.0	30.2±3.9	27.8± 4.3	28.0± 4.5	28.0±4.4
Diabetes	27.9± 1.5	26.4±1.7	27.2±1.6	24.4± 1.7	24.4± 1.7	24.4±1.7
German	26.9± 1.9	25.3± 1.6	24.8±1.9	24.6± 2.1	24.7± 2.1	24.8±2.2
Heart	20.1± 2.7	19.6±2.8	20.0±2.8	18.4± 3.0	18.2± 2.7	18.3±2.8
Ringnorm	1.9± 0.3*	1.9± 0.2	1.9± 0.2	1.9± 0.2	1.8± 0.2	1.7± 0.2
F.Solar	36.1± 1.5	35.5± 1.5	36.1±1.4	35.7± 1.6	35.5± 1.4	35.5±1.6
Thyroid	4.4± 1.9*	4.7±2.1	4.6± 2.1	4.9± 1.9	4.9± 1.9	4.7±1.9
Titanic	22.8± 1.0	22.8±0.9	22.8±0.8	22.8± 1.0	23.0± 0.8	22.8±1.0
Waveform	10.5± 0.4	10.1±0.4	10.4±0.4	10.1± 0.5	9.8± 0.5	10.1±0.5

repository, with parameters set according to the above description. The results confirm our theoretical intuition that Entropy Regularized LPBoost maximizes the soft margin and more generally, that it does not deviate much from LPBoost. Moreover, it does not start as slowly as SoftBoost. Also included in this plot is the corrective version of Entropy Regularized LPBoost proposed in [21].

Using data from the same run used to generate Figure 2, we also examine generalization error as a function of the number of iterations. The results, shown in Figure 3, confirm that all of the algorithms have similar generalization in the end. Although the corrective algorithm takes many more iterations to converge, its generalization error eventually approaches that of the totally corrective algorithm.

The means and standard deviations for the full benchmark comparison of AdaBoost, LogitBoost, BrownBoost, LPBoost, SoftBoost, and Entropy Regularized LPBoost and are given in Table 1.⁵ As we expected, the generalization performances of the soft margin algorithms – LPBoost, Entropy Regularized LPBoost, and SoftBoost – are very similar. In fact, because both SoftBoost and Entropy Regularized LPBoost are intended to approximate LPBoost, we would have been very surprised to see a significant difference in their generalization error. The soft margin algorithms outperform AdaBoost on most data sets, while the generalization error of BrownBoost and LogitBoost lie between that of AdaBoost and the soft margin algorithms. Comparing the soft margin algorithms with AdaBoost, LogitBoost and BrownBoost reinforces the idea that maximizing the soft margin results in good algorithms.

Finally, we compared the totally corrective Entropy Regularized LPBoost with the corresponding corrective algorithm of [21]. While a single iteration of the corrective algorithm is faster than a single iteration of the totally corrective algorithm (implemented with sequential quadratic programming), the overall

⁵ Note that [15] contains a similar benchmark comparison. It is based on a different model selection setup leading to underestimates of the generalization error. Presumably due to slight differences in the RBF hyper-parameters settings, our results for AdaBoost often deviate by 1-2%.

Table 2. Comparison of total training time in seconds until margin is within $\epsilon = 0.1$ of optimum for LPBoost, Entropy Regularized LPBoost, and the corrective version

	LPBoost	Ent. Reg. LPBoost	Corr. Ent. Reg. LPBoost
Ringnorm	822	2.83e3	2.08e5
Titanic	80	102	2.02e4

running time of the totally corrective algorithm was much faster in our experiments. To demonstrate this, we selected ringnorm and titanic, the largest and smallest datasets of Table 1 respectively, and compared the total running times of LPBoost, Entropy Regularized LPBoost, and Corrective Entropy Regularized LPBoost (Table 2). We used the CPLEX optimizer on a cluster with Intel Xeon X5355 2.66GHz processors. The corrective algorithm of [21] does not come with a stopping criterion, so for a fair comparison we first determined the optimum margin to high precision. We then timed each algorithm until its margin was within $\epsilon = 0.1$ of the optimum. Observe that the corrective algorithm is significantly slower than the totally corrective algorithms on both datasets.

7 Conclusion

We used duality methods instead of Bregman projections to obtain a simple proof for the iteration bound of Entropy Regularized LPBoost⁶. We were not able to prove the bound by making fixed progress in each iteration. Instead, following [22] we arrived at a quadratic recurrence equation for the remaining duality gap and, in solving this recurrence, achieved the optimal bound.

We show that the Entropy Regularized LPBoost algorithm has performance comparable to LPBoost and SoftBoost on natural data. It fixes the slow start problem of SoftBoost. We believe that the striking simplicity of the optimization problem for Entropy Regularized LPBoost in which a relative entropy is traded off against a linear objective will lead to further insights into the nature of Boosting and its striking difference to the main other family of learning algorithms (which includes Support Vector Machines (SVMs)) that are motivated by regularizing with the squared Euclidean distance.

Acknowledgments. Thanks to Gunnar Rätsch for valuable cluster time. He also helped us focus on the optimization problem underlying Entropy Regularized LPBoost and develop its dual. The first author was supported by NSF grant CCR 9821087, the second by Los Alamos National Labs and the third by NICTA, Canberra.

⁶ This is not too surprising because Bregman projection methods and duality methods are often interchangeable. See [13] for a number of cases where both methods are given for proving iteration bounds on boosting algorithms.

References

- [1] Abe, N., Takeuchi, J., Warmuth, M.K.: Polynomial learnability of stochastic rules with respect to the KL-divergence and quadratic distance. *IEICE Transactions on Information and Systems* E84-D(3), 299–316 (2001)
- [2] Boyd, S., Vandenberghe, L.: *Convex Optimization*. Cambridge University Press, Cambridge (2004)
- [3] Demiriz, A., Bennett, K.P., Shawe-Taylor, J.: Linear programming boosting via column generation. *Mach. Learn.* 46(1-3), 225–254 (2002)
- [4] Duffy, N., Helmbold, D.: Potential boosters? In: Solla, S., Leen, T., Müller, K.-R. (eds.) *Advances in Neural Information Processing Systems 12*, pp. 258–264. MIT Press, Cambridge (2000)
- [5] Freund, Y.: Boosting a weak learning algorithm by majority. *Inform. Comput.* 121(2), 256–285 (1995); In: *COLT 1990*
- [6] Freund, Y.: An adaptive version of the boost by majority algorithm. In: *Proceedings of the 12th annual conference on Computational learning theory*, pp. 102–113. ACM Press, New York (1999)
- [7] Freund, Y., Schapire, R.: 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] Friedman, J., Hastie, T., Tibshirani, R.: Additive Logistic Regression: a Statistical View of Boosting. *The Annals of Statistics* 38(2) (2000)
- [9] Grove, A.J., Schuurmans, D.: Boosting in the limit: maximizing the margin of learned ensembles. In: *AAAI 1998/IAAI 1998*, Menlo Park, CA, USA, pp. 692–699 (1998)
- [10] Helmbold, D., Schapire, R.E., Singer, Y., Warmuth, M.K.: A comparison of new and old algorithms for a mixture estimation problem. *Machine Learning* 27(1), 97–119 (1997)
- [11] Kivinen, J., Warmuth, M.K.: Boosting as entropy projection. In: *Proc. 12th Annu. Conf. on Comput. Learning Theory*, pp. 134–144. ACM Press, New York (1999)
- [12] Lafferty, J.: Additive models, boosting, and inference for generalized divergences. In: *Proceedings of the 12th Annual Conference on Computational Learning Theory*, pp. 125–133. ACM Press, New York (1999)
- [13] Liao, J.: *Totally Corrective Boosting Algorithms that Maximize the Margin*. PhD thesis, University of California at Santa Cruz (December 2006)
- [14] Rätsch, G.: *Robust Boosting via Convex Optimization: Theory and Applications*. PhD thesis, University of Potsdam (2001)
- [15] Rätsch, G., Onoda, T., Müller, K.-R.: Soft margins for adaboost. *Mach. Learn.* 42(3), 287–320 (2001)
- [16] Rätsch, G., Schölkopf, B., Smola, A., Mika, S., Onoda, T., Müller, K.-R.: Robust ensemble learning. In: Smola, A., Bartlett, P., Schölkopf, B., Schuurmans, D. (eds.) *Advances in Large Margin Classifiers*, pp. 207–219. MIT Press, Cambridge, MA (2000)
- [17] Rätsch, G., Warmuth, M.: Efficient margin maximizing with boosting. *Journal of Machine Learning Research* 6, 2131–2152 (2005)
- [18] Rudin, C., Schapire, R., Daubechies, I.: Boosting based on a smooth margin. In: *Proceedings of the 17th Annual Conference on Computational Learning Theory*, pp. 502–517 (2004)
- [19] Rudin, C., Schapire, R., Daubechies, I.: Analysis of boosting algorithms using the smooth margin function. *The Annals of Statistics* 6(35), 2723–2768 (2007)

- [20] Schapire, R., Freund, Y., Bartlett, P., Lee, W.: Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics* 26(5), 1651–1686 (1998)
- [21] Shalev-Shwartz, S., Singer, Y.: On the equivalence of weak learnability and linear separability: New relaxations and efficient boosting algorithms. In: *Proceedings of the 21st annual conference on Computational learning theory*, pp. 311–321. Omicron (2008)
- [22] Smola, A., Vishwanathan, S.V.N., Le, Q.: Bundle methods for machine learning. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) *Advances in Neural Information Processing Systems 20*, pp. 1377–1384. MIT Press, Cambridge (2008)
- [23] Warmuth, M., Liao, J., Rätsch, G.: Totally corrective boosting algorithms that maximize the margin. In: *ICML 2006*, pp. 1001–1008. ACM Press, New York (2006)
- [24] Warmuth, M.K., Glocer, K., Rätsch, G.: Boosting algorithms for maximizing the soft margin. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) *Advances in Neural Information Processing Systems 20*. MIT Press, Cambridge (2007)

Optimally Learning Social Networks with Activations and Suppressions

Dana Angluin, James Aspnes*, and Lev Reyzin**

Department of Computer Science, Yale University
{angluin,aspnes}@cs.yale.edu, lev.reyzin@yale.edu

Abstract. In this paper we consider the problem of learning hidden independent cascade social networks using exact value injection queries. These queries involve activating and suppressing agents in the target network. We develop an algorithm that optimally learns an arbitrary social network of size n using $O(n^2)$ queries, matching the information theoretic lower bound we prove for this problem. We also consider the case when the target social network forms a tree and show that the learning problem takes $\Theta(n \log(n))$ queries. We also give an approximation algorithm for finding an influential set of nodes in the network, without resorting to learning its structure. Finally, we discuss some limitations of our approach, and limitations of path-based methods, when non-exact value injection queries are used.

1 Introduction

Social networks are used to model interactions within populations of individuals. These interactions could include distributing information, spreading a disease, or passing trends among friends. Viral marketing is an often used example of a process well modeled by social networks. In viral marketing, a company may want to market a product to some population. The idea is to carefully choose some people to target. This can be done, for instance, by giving these people a free sample of the product. The targeted people have relationships in their population, and they will possibly virally spread interest in this product to their friends, and so on.

Social networks are often used to describe this type of phenomenon, and there are many different models of social networks. All of these models (imperfectly) approximate these complicated real world phenomena. One of the most basic and well-studied models is the **independent cascade model** [6, 10, 9], and it is the one we consider in this paper. Informally, in the independent cascade model each individual, or agent, has some probability of influencing each other agent. When an agent is targeted with a product, he becomes activated, and then attempts to influence each of his friends, and so on. This model is called independent cascade

* Supported in part by NSF grant CNS-0435201.

** This material is based upon work supported under a National Science Foundation Graduate Research Fellowship.

because each agent's success probability in attempting to influence a friend is independent of the history of previous activation attempts in the network.

Social networks belong to the wider class of probabilistic networks. Probabilistic networks are circuits whose gate functions specify, for each combination of inputs, a probability distribution on the output. In the case of social networks, these gates compute rather simple functions of their inputs.

A natural question to ask is: what can we learn about the structure of these networks by experimenting with their behavior? Given access to a pool of agents in our network, one intuitive way in which we could experiment on this network would be to artificially excite some set of agents, for example by sending them political brochures in support of some measure, and then observe the consequences of the experiment. Furthermore, we will allow for the possibility of suppressing agents; when an agent is suppressed, he cannot be excited by another agent. To make things more realistic, and theoretically more interesting, we will not assume that we can observe the entire network. We will instead have an output agent, whose state at the end of this process we can see, for example the probability the president supports the measure.

Thus, in this paper we consider the setting where we can **inject values** into the network; we fix the states (or values) of any subset of agents in the target network and only observe the state of some specified agent, whom we think of as the output of the network. This is the value injection query model.

The idea of value injection queries was inspired both from hardness results in learning circuits by only manipulating inputs [5,11,8] and by models of gene suppression and gene overexpression in the study of gene interaction networks [1,7] and was proposed by Angluin et al [4]. They show that acyclic deterministic boolean circuits with constant fan-in and $O(\log n)$ depth are learnable in polynomial time with value injection queries. Angluin et al. [3] extend these results to circuits with polynomial-size alphabets. They show that transitively reduced acyclic deterministic circuits that have polynomial-size alphabets, constant fan-in, and no depth bound are learnable in polynomial time with value injection queries. Then, Angluin et al. [2] extend this work to probabilistic circuits. They show that constant fan-in acyclic boolean probabilistic circuits of $O(\log(n))$ depth can be approximately learned in polynomial time, but that this no longer necessarily holds once the alphabet becomes larger than boolean.

However, unlike in previous work on the value injection model, we allow our target social networks to have cycles. In many classes of networks, allowing for cycles would make the problem ill-defined in the value injection model, as the values on the nodes of the network may not be stable. In the social networks case, the values of the nodes in the network converge. Also, unlike in previous work, our learnability results do not require a degree bound on the target network. This gives us a nice theoretical model whose properties are interesting to explore.

In Section 2 we formally define the model, value injection queries, and learning criteria. In Section 3 we develop an algorithm that learns any social network in $O(n^2)$ queries and prove a matching lower bound for this problem. In Section 4 we show that in the special case when the network comes from the class of

trees, learning the network takes $\Theta(n \log(n))$ queries. In Section 5 we show some limitations of using path-based methods for learning social networks when value injection queries do not return exact probability distributions of value of the output node, which is the case in real-world settings. In Section 6 we give an approximation algorithm for learning influential sets of nodes in a social network.

2 Model

2.1 Social Networks

We consider a class of circuits that represents social networks. We are specifically interested in a variant of the model of deterministic circuits defined in [3, 4]. The social networks have no distinguished inputs – instead, value-injection experiments may be used to override the values on any subset of the agents.

An **independent cascade social network** S consists of a finite nonempty set of independent excitation agents A , one of which is designated as the **output agent**. Agents take values from a boolean alphabet $\Sigma = \{0, 1\}$, corresponding to the states *waiting* and *activated*, respectively. The size of the social network is $n = |A|$.

An **independent excitation** agent function f on k inputs has k parameters: the probabilities $p_1, \dots, p_k$. If the inputs to the agent are $(b_1, \dots, b_k) \in \{0, 1\}^k$, then the probability that $f(b_1, \dots, b_k)$ is 0 is

$$\prod_{i=1}^k (1 - p_i)^{b_i}.$$

We define $0^0 = 1$.

If we are told, in an arbitrary order, which inputs to f are 1, then we may sample from the correct output distribution for f as follows. Initially the output is 0. Given that $b_i = 1$, then with probability p_i we set the output to 1 and with probability $(1 - p_i)$ we leave it unchanged. This corresponds to our intuitive notion of the behavior of social networks; when a neighbor of an agent is activated, the agent has some probability of becoming activated as well, and an agent will remain inactive if it was not activated by any of its neighbors.

2.2 Graphs of Social Networks

The weighted **network graph** of the social network has vertices A and a directed edge (u, v) if agent u is one of the inputs of agent v . If u is an input to v with activation probability $p_{(u,v)}$, then the edge has weight $p_{(u,v)}$. We say an edge exists if it has positive weight. The weighted network graph of a social network captures all relevant information about the social network. Therefore, we will often refer to a social network in terms of its graph. The **depth** of a node in the network is the number of edges in the shortest path from the node to the output. The depth of the network is the maximum over the depths of all the

nodes in the network. The network is **acyclic** if the network graph contains no directed cycles. Unlike in previous work on value injection queries, in this paper we consider networks that may have cycles.

2.3 Experiments

The behavior of a social network consists of its responses to all possible value-injection experiments. In an experiment, some agents are fixed to values from $[0, 1]$ and others are left free. Fixing an agent to a 1 corresponds to **activating** or **firing** the agent, fixing to a 0 corresponds to **suppressing** the agent, and leaving an agent **free** allows it to function as it normally would. Fixing an agent to a value c between 0 and 1 corresponds to firing the agent with probability c and suppressing it with probability $1 - c$.

Formally, a **value-injection experiment** (or just experiment) e is a mapping from A to $\{[0, 1] \cup \{*\}\}$. If $e(g)$ is $*$, then the experiment e leaves agent g **free**; otherwise g is **fixed** to the value $e(g) \in [0, 1]$. If e is any experiment and $a \in [0, 1] \cup \{*\}$, the experiment $e|_{w=a}$ is defined to be the experiment e' such that $e'(w) = a$ and $e'(u) = e(u)$ for all $u \in A$ such that $u \neq w$.

We can define the behavior of a social network S as a function of a value-injection experiment in two different ways. The first is a percolation model. For each edge (u, v) , we leave it “open” with probability $p_{(u,v)}$ and “closed” with probability $1 - p_{(u,v)}$. For each node w in S , such that $e(w) = c$ for some $c \in [0, 1]$, we make node w fired with probability c and suppressed with probability $1 - c$. We let the indicator variable $I = 1$ if there is direct path using open edges from some fired node to the output node via free nodes, and we let $I = 0$ otherwise. This determines a probability distribution on assignments of 0 and 1 to I . We define the output $S(e)$ to be $E(I)$.

The following process, equivalent to the percolation model, defines the behavior of social network as a function of a value-injection experiment e . It is also the process that will guide the intuition and proofs in this paper. Initially every node is tentatively assigned the value 0. There is a queue of nodes to be assigned values, which initially contains the nodes fixed to values > 0 by e . The assignments are complete when the queue becomes empty. While the queue is nonempty, its first node v is dequeued. If $e(v) = *$, v is assigned the value 1. If $e(v) \neq *$, v is assigned a 1 with probability $e(v)$, and 0 with probability $(1 - e(v))$. If v is assigned a 1, for every node u such that v is an input to u , do the following.

1. If u is fixed to any value, or already assigned 1 or present in the queue, do nothing.
2. Otherwise, with probability $p_{(v,u)}$ add u to the queue, and with probability $(1 - p_{(v,u)})$ do nothing.

This process determines a joint probability distribution on assignments of 0 and 1 to the nodes of the social network S . In this case, the output $S(e)$ is the expected value of the output node given by e .

2.4 Behavior and Equivalence

The **behavior** of a network is the function that maps experiments e to output excitation probabilities $S(e)$. Two social networks S and S' are **behaviorally equivalent** if they have the same set of agents, the same output agent, and the same behavior, that is, if for every value-injection experiment e , $S(e) = S'(e)$. We also define a concept of approximate equivalence. For $\epsilon \geq 0$, S is **ϵ -behaviorally equivalent** to S' if they contain the same agents, the same output agent and for every value-injection experiment e , $|S(e) - S'(e)| \leq \epsilon$.

2.5 Queries

The learning algorithm gets information about the target network by repeatedly specifying an experiment e and observing the value assigned to the output node. Such an action is termed a **value injection query**. A value-injection query does not return $S(e)$, but instead returns a $\{0, 1\}$ value selected according to the probability $S(e)$. This means that the learner must repeatedly sample to approximate $S(e)$. To separate the effects of this approximation from the inherent information requirements of this problem, we define an **exact value injection query** to return $S(e)$. The focus of this paper is on exact value injection queries.

2.6 The Learning Problem

The learning problem we consider is: by making exact value injection queries to a target network S drawn from a known class of social networks, find a network S' that is behaviorally equivalent to S . The inputs to the learning algorithm are the names of the agents in S and the name of the output agent.

To help with terminology, let S be a social network. Let S' be any social network that differs only in edge (u, v) . We say edge (u, v) is **discoverable** for S if there exists an experiment e such that $S(e) \neq S'(e)$. Otherwise we say that the edge is not discoverable. We could also view the learning problem in terms of finding the discoverable edges and their probabilities.

2.7 A Note on the Generality of This Model

The model introduced in this section allows for the observation of the network by looking at the output of one selected node. However, this model is surprisingly general. One may wish to consider, for example, the ability to observe the number of nodes to fire as a result of an experiment. Such a scenario could be simulated in our model – given any social network, one could make a new output node that is activated by each node with some fixed, chosen probability. Now the probability the output is activated corresponds to the number of network nodes that are activated in an experiment.

One could also imagine networks where some nodes spontaneously fire with some probability. We can again simulate this in the model we introduced. We add a node that is fired with probability 1 whenever any node in the network

fires (all other nodes have 1-edges to the new node), and the new node can have edges to each node in the network, with probabilities corresponding to the desired spontaneous firing probabilities of the network nodes.

3 General Social Networks

In this section we prove the following theorem.

Theorem 1. *Any social network with n agents can be learned up to behavioral equivalence with $O(n^2)$ exact value injection queries and time polynomial in the number of queries.*

First, we develop excitation paths, which are a variant of test paths, a concept central in previous work on learning deterministic circuits [3, 4]. An **excitation path** for an agent a is a value-injection experiment in which a subset of the free agents form a simple directed path¹ in the circuit graph from (but not including) a to the output agent. All agents not on the path with inputs into the path are fixed to 0. A **shortest excitation path** is an excitation path of length equal to the depth of a .

Let G be the network graph of A . In G , the **up edges** are edges from nodes of larger depth to nodes of smaller depth, **level edges** are edges among nodes of the same depth, and **down edges** are edges from nodes of smaller depth to nodes of larger depth. An edge (u, v) is a **shortcut edge** if there exists a directed path in G of length at least two from u to v .

Lemma 1. *Let e be a shortest excitation path for node a and π be the nodes on the path. Let $p_1 \cdots p_k$ be the weights of the up edges in $\pi \cup a$. Then for $0 \leq c \leq 1$*

$$S(e|_{a=c}) = c \prod_{i=1}^k p_i.$$

Proof. In a shortest excitation path, if some node on the path does not activate, no node at smaller depth will activate, because a shortest excitation path cannot have shortcuts to nodes further along the path. Hence, all up edges must fire to fire the output. This happens exactly with probability $\prod_{i=1}^k p_i$. We note that this lemma still holds when a takes probability c , not only when its set to c by e . $\square$

Lemma 2. *Let e be an excitation path experiment for node v and let $\pi = v_k, \dots, v_0$ be the nodes along π in order from v to the output (with v_0 being the output node), such that there are no shortcut edges (v_i, v_j) for $j < i$ along π . Let $u \notin \pi$ be a node such that all edges from u to nodes on π are known and have weights < 1 . Let $e' = e|_{v=*, u=1}$. Then, given $S(e')$ we can compute $p_{(u,v)}$.*

Proof. We observe that because there are no shortcuts along π , no node v_i will activate in e' unless either u activated it, or v_{i+1} . Hence, any edge (v_j, v_k) where

¹ A path with no repeated vertices is called simple.

$j < k$ does not affect $S(e')$. Therefore, we can compute $S(e')$ by summing over all the ways v_0 can activate. Either u activates it directly with probability $p_{(u,v_0)}$, or if not (with probability $1 - p_{(u,v_0)}$) we look at the probability u activates v_1 and the probability of v_1 firing the output, and so on. These quantities can be computed using the logic of Lemma 1. For the calculation below, we rename node v to v_{k+1} .

$$S(e') = \sum_{i=0}^{k+1} \left(p_{(u,v_i)} \prod_{j<i} (1 - p_{(u,v_j)}) (p_{(v_{j+1},v_j)}) \right)$$

This equation is linear in $p_{(u,v_{k+1})}$, which we can solve for because the other quantities are known. $\square$

We now present Algorithm 1 for learning social networks.

Algorithm 1. Learning Arbitrary Social Networks

Let S be the target social network.
Initialize G to have the agents as vertices and no edges.
Run **Find-Up-Edges** to learn the leveled graph of S .
Add learned weighted edges to G .
Let $C = \emptyset$ be the complete set.
for Each level i in the graph, from the deepest level to the output node **do**
 Run **Find-Remaining-Edges**(G, C, i) to learn all level and down edges.
 Add all nodes at the current level to C .
end for
Output G and halt.

The subroutine **Find-Up-Edges** builds a leveled graph of S . Let **level** i be the set of all nodes at depth i . Find-Up-Edges assigns each node to a level and finds all up edges in the graph. Starting at the top level and proceeding downward, for each pair of nodes u and v , such that u is one level deeper than v , Find-Up-Edges finds a shortest excitation path for u that goes through v to learn $p_{(u,v)}$. This experiment leaves the path free and suppresses all other nodes in the graph. We show correctness by induction on the level. For the base case, the edges from nodes at depth 1 form the paths. Considering nodes at depth i we assume we know all up edges on the induced subgraph at depths 0 to $i - 1$. Therefore, for each node at depth $i - 1$ we have a shortest excitation path to the root. Thus, for each node u not yet assigned a level, we can try experiments with excitation paths via each node v at depth $i - 1$. Let e be such an experiment with π as the excitation path. And let $p_1 \cdots p_{i-1}$ be the weights of the up edges in π . By Lemma 1 we can compute

$$p_{(u,v)} = \frac{S(e|_{u=1})}{\prod_{j=1}^{i-1} p_j}.$$

If $p_{(u,v)} > 0$ we assign node u to level i .

After Find-Up-Edges is run, the remaining edges that need to be found are down and level edges. The subroutine **Find-Remaining-Edges**, shown in Algorithm 2, accomplishes this task. The algorithm keeps a complete set C in which all discoverable edges are known. C starts at the largest level and grows toward smaller levels. Find-Remaining-Edges finds all discoverable edges from the level it is on to the complete set. It also finds all discoverable edges between nodes at the level it is on. Then, that level is added to C .

Algorithm 2. Find-Remaining-Edges(Current Graph G , Complete Set C , Level i)

```

Let  $L$  be the set of nodes at the current level  $i$ .
Let  $M = L \cup C$ .
Let  $\Pi$  be a collection of paths.
Keep an  $|L|$  by  $|M|$  table  $T$ .  $\forall w_i \in L, x_j \in M$  s.t.  $w_i \neq x_j, T(w_i, x_j) = 1$ .
loop
    Set  $\Pi = \emptyset$ .
    for Each node  $w_i \in L$  do
        Find  $A_{w_i}$ , the set of all nodes reachable from  $w_i$  by 1-edges (incl.  $w_i$ ) in  $G$ .
        for Each node  $x_j \in M$  where  $T(w_i, x_j) = 1$  do
            Find the shortest path  $\pi_{w_i, x_j}$  in  $G - A_{w_i}$  from  $x_j$  to the root.
             $\Pi = \Pi \cup \pi_{w_i, x_j}$ .
        end for
    end for
    if  $\Pi = \emptyset$  then return.
    Let  $w_i, x_j$  minimize the length of  $\pi_{w_i, x_j} \in \Pi$ .
    Let experiment  $e$  fire  $w_i$ , leave  $\pi_{w_i, x_j}$  free, and suppress the rest of the nodes.
    Query  $S(e)$  and compute  $p_{(w_i, x_j)}$  by Lemma 2.
    Set  $T(w_i, x_j) = 0$ .
    If  $p_{(w_i, x_j)} > 0$  then add  $(w_i, x_j)$  to  $G$ .
end loop

```

Let L be the set of nodes on level i . To find down and level edges from nodes in L , Find-Remaining-Edges keeps a table T , with an entry for each possible edge originating from a node in L . Each entry is initially set to 1. After determining whether an edge is present, its corresponding entry becomes marked 0. The potential edges whose corresponding entries are marked 1 we call “unprocessed.”

For each unprocessed edge (u, v) , we find the set of all nodes we know are guaranteed to be activated when u is fired. This is the set of nodes reachable by edges of weight 1 from u in G . We call this set A_u . Now, we find the shortest path $\pi_{u, v}$ (if one exists) in $G - \{A_u\}$ from v to the output. If no unprocessed edge has such a path, then Find-Remaining-Edges terminates and the algorithm proceeds to the next level.

Otherwise, we take an edge (u, v) that minimizes the distance from v to the output in $G - \{A_u\}$. Let e be the experiment where u is fired, all nodes along $\pi_{u, v}$ are left free, and the rest of the nodes are suppressed. We will show that $S(e)$ is enough to determine $p_{(u, v)}$. Then, the entry for this edge is marked to 0

in the table, and if it is present, is added to G . Then the algorithm continues, recomputing the sets A_u for the remaining unprocessed edges.

We now show that the value of $S(e)$, as defined above, is sufficient to learn edge (u, v) . All edges from u to π are either up edges or have already been processed by the time edge (u, v) is considered, otherwise there would be an unprocessed edge from u to a node on π with a shorter distance to the root in $G - A_u$. All edges on π in $G - C$ are known from Find-Up-Edges, and the rest of the edges are known because they are in C . Hence, by Lemma 2, we can compute the weight of edge (u, v) , and add it to G if its weight is positive.

Find-Remaining-Edges returns when all remaining unprocessed down and level pairs of nodes u, v do not have a path from v to the root in $G - A_u$. The algorithm does not attempt to learn these edges. We will argue that when an execution of Find-Remaining-Edges terminates, all of the unprocessed edges are not discoverable. Let u, v be such a pair. Let S be the graph of the complete social network and B_u be the set of nodes reachable by edges of weight 1 in S . If there is no path from v to the root in $S - B_u$, edge (u, v) is clearly not discoverable. We note that $A_u \subseteq B_u$.

By way of contradiction, we will assume there exist vertices u (on level i) and v (on level $\geq i$) such that there is a path of discoverable edges from v to the root in $S - B_u$ but not in $G - A_u$ at the time Find-Remaining-Edge exits. Once this path reaches level $i - 1$ in G , then the path can be continued by following up edges to the root. By assumption, G has all discoverable edges among the complete set C , which contains all nodes at levels $> i$. Hence, there must be some smallest set of edges U going from nodes at level i , that are in S but not in G , such that if they were added to G , then there would be a path from v to the root node in $G - A_u$. All of the edges in U must lie on a path π . Let edge $(x, y) \in U$ be the unprocessed edge closest to the root along the path. Because edge (x, y) was unprocessed, there was a path of 1 edges from x to a node in π above y ; otherwise, there would be a path from y to the root in $G - A_x$ and (x, y) would have been processed. But taking the path of 1 edges from x to a node in π gives a path from v to the root in $G - A_u$ using one fewer unprocessed edge. This contradicts that U was the smallest set of edges that, if added to G , would make a path from v to the root in $G - A_u$. This contradicts our assumption that a discoverable edge exists that Find-Remaining-Edges does not find.

To analyze number of queries used, we observe that every query either confirms the absence of an edge or discovers one. Hence, Algorithm 1 performs at most $O(n^2)$ queries.

3.1 A Matching $\Omega(n^2)$ Lower Bound

We show an information theoretic lower bound for learning social networks that matches the bound of the algorithm.

Theorem 2. $\Omega(n^2)$ queries are required to learn a social network.

Proof. We give an information theoretic lower bound. We consider the following class of graphs on vertices $\{v_1, \dots, v_{2n+1}\}$. We let v_{2n+1} be the output. The

edges $(v_{n+1}, v_{2n+1}), (v_{n+2}, v_{2n+1}), \dots, (v_{2n}, v_{2n+1})$ all have weight 1. The edges $(v_1, v_{n+1}), (v_2, v_{n+2}), \dots, (v_n, v_{2n})$ also all have weight 1. For $1 \leq i \leq n$, $n+1 \leq j \leq 2n$, and $j \neq i+n$, each edge (v_i, v_j) is either present with weight 1 or absent. The rest of the edges are absent. There are $2^{\Omega(n^2)}$ such graphs and the answer to every exact value injection query is 1 bit because all present edges have weight 1. Algorithm 1 differentiates all graphs in this class because all edges in this class of graphs are up edges and are therefore discoverable. Hence, by an information theoretic lower bound, at least $\log 2^{\Omega(n^2)} = \Omega(n^2)$ queries are needed. $\square$

4 Trees

In this section, we will consider the special case in which the target social networks come from the class of trees. A **tree social network** is a social network whose edges are up edges that form a tree.

Theorem 3. *Learning a social network tree takes $\Theta(n \log n)$ exact value injection queries.*

Proof. We first show the lower bound. Consider a directed path of nodes, with the output node at an endpoint. All edges along the path have probability 1. The only unknown is the ordering of the nodes along the path. Let u and v be two nodes. We can test which of the two nodes has a smaller distance to the root by the experiment that fires u and suppresses v . If this fires the output, then u is closer to the root; otherwise, v is closer. Hence, all orderings can be distinguished. Because all edges have probability one, the result of any experiment is deterministically a 1 or 0, a 1-bit answer. There are $n!$ orderings of nodes. This gives an $\Omega(\log(n!)) = \Omega(n \log(n))$ information-theoretic lower bound.

We now develop an algorithm that meets this bound for trees. Let T be the target tree social network. In a tree, an **ancestor** of node u is any node on the path from u to the output. We can test whether node v is an ancestor of node u by firing u and suppressing v . If the result is > 0 , then v is not an ancestor of u . In general, to test whether there exists some node in V that is an ancestor of u , we can fire u and suppress all nodes in V . This allows us to find all k ancestors of a given node u by binary search in $O(k \log(n))$ queries. Because the ancestors of u form a path, we can sort them by their depth using $O(k \log(k))$ queries (an ancestor test involving two nodes provides a comparator) to get a directed path from u to the output.

Now, we will use the observation above to make an algorithm for reconstructing trees. We keep a graph T' that is a connected subgraph of T that we build up by adding new nodes until T' contains all the vertices in T . In attaching a new node u to T' , we first determine v , u 's deepest ancestor in T' . We can do this by recursively by splitting the nodes in T' into roughly equal halves H_1 and H_2 such that no node in H_2 is an ancestor of a node in H_1 . In one query we can test whether v is in H_1 by suppressing all nodes in H_2 and firing u ; thus, we can find v in $\log(n)$ queries. We then find, by binary search, the set of all ancestors of u in T that are not in T' , and we sort them by their distance to the root in T .

This gives a path of vertices from u to v that we can append to T' and continue this process until all the vertices are added to T' .

In adding a new node u to T' we spend $(\log(n))$ queries to find its deepest ancestor in T' , and $O(k \log(n))$ queries to add u 's $k \geq 0$ newly found ancestors to T' . This costs us an amortized $O(\log(n))$ queries per node, giving an $O(n \log(n))$ algorithm for learning the structure of the tree. We note that the structure is learned using just zero/non-zero information from the queries.

Finally, to learn the weights of the edges in the tree, because we have a shortest excitation path for each node, the edge weights can be discovered in n queries by Lemma 2. $\square$

5 Limitations of Excitation Paths

In this section, we construct a family of social networks in which there exists a node, that when fired, activates the output node with high probability, but any excitation path experiment for that node has an exponentially small probability of activating the output. Namely, we will prove the following theorem.

Theorem 4. *There exists a family of social networks S for which there exists a node $v \in S$ and an experiment e where only v is fired, such that for any excitation path experiment e_π for v ,*

$$S(e) = 2^{\Omega(\sqrt{n})} S(e_\pi)$$

Proof. Let $\{v_1, \dots, v_n\}$ be nodes in this network, with v_1 the output node. For all $1 < i < n-1$, let $p_{(i,i+1)} = 1$ - we call these **back edges**. For all $i, j > 0$ such that $i + j \leq n$, create a new node w_{ij} and let $p_{(w_{ij}, v_i)} = 1$ and $p_{(v_{i+j}, w_{ij})} = 2^{-j/\sqrt{n}}$.

Let e_1 be an excitation path experiment for v_n , where v_n is fired. $S(e_1)$ is the probability all edges along the path fire, which we can bound

$$\begin{aligned} S(e_1) &= \prod_{f_i: \sum f_i \geq n-1} 2^{-\frac{f_i}{\sqrt{n}}} \\ &= 2^{-\frac{\sum f_i}{\sqrt{n}}} \\ &= 2^{-\Omega(\sqrt{n})} \end{aligned}$$

The f_i 's represent the number of nodes forward their corresponding edges jump.

Let e_2 be the experiment where v_n is fired and the remaining agents are set free. We will show there exists a constant $c > 0$ such that $S(e_2) \geq c$.

We consider e_2 . The probability that v_n does not fire any other nodes is $\prod_{i=1}^n (1 - 2^{-i/\sqrt{n}})$. Now, we can bound the probability of the root firing. Let $T(i)$ be the probability the root becomes activated given v_i has fired. We set up a recurrence

$$\begin{aligned} T(1) &= 1 \\ T(n) &\geq \left(1 - \prod_{i=1}^{n-1} \left(1 - \frac{1}{2^{i/\sqrt{n}}}\right)\right) T(n-1) \end{aligned}$$

Where we have an inequality above because if v_n activates any other node, then v_{n-1} becomes activated due to the back edges.

Thus, $T(n) \geq \left(1 - \frac{1}{2}\sqrt{n}\right) T(n-1)$ because the first $\sqrt{n}$ terms of the product above are $\leq 1/2$. Unraveling the recurrence, we get

$$T(n) \geq \prod_{i=1}^{n-1} \left(1 - \frac{1}{2}\sqrt{i}\right),$$

We know $\lim_{n \rightarrow \infty} T(n) > 0$ if $\sum_{i=1}^{\infty} \frac{1}{2}\sqrt{i}$ converges. By the Cauchy Condensation Test, $\sum_{i=1}^{\infty} \frac{1}{2}\sqrt{i}$ converges if and only if $\sum_{i=1}^{\infty} 2^n \frac{1}{2}\sqrt{2^n}$ converges [14]. The ratio test easily tells us that $\sum_{i=1}^{\infty} 2^n \frac{1}{2}\sqrt{2^n}$ converges. Therefore, there exists a constant $c > 0$ such that $\forall n \ T(n) \geq c$. $\square$

This example shows that many paths, each of which has an exponentially small effect on the output, can add up to have a detectable effect on the output. When using non-exact value injection queries, the goal is to learn a circuit to approximate behavioral equivalence. Yet this example shows us that if the learner has access to only non-exact value injection queries, then to learn this circuit by only path based methods like our algorithms do, one would need an exponential number of experiments to detect the effect on the output. This implies that for non-exact value injection queries, either the circuits would need a depth limitation, or non path-based algorithms would need to be developed.

6 Finding Small Influential Sets of Nodes

We now examine a seemingly easier problem. Instead of learning the entire social network, we consider the task of finding a small set of influential nodes. More formally, let $I \subset V$ such that $v_n \notin I$, and let e_I be the experiment where all nodes in I are fired and the rest are left free. I has **influence** p if $S(e_I) \geq p$; we call such a set **influential**. We first show that it is NP-Hard to find the smallest set of certain influence, even if the structure of the network is known.

Theorem 5. *Given a social network S of size n and a threshold probability p , it is NP-Hard to approximate the size of the smallest set of nodes having influence p within $o(\log(n))$.*

Proof. We reduce from Set Cover. Take an instance of Set Cover with points $\{x_1, \dots, x_k\}$ and sets $\{X_1, \dots, X_l\}$. In the social network S , we create a nodes $\{v_1, \dots, v_k\}$ for the points and $\{w_1, \dots, w_l\}$ for the sets in the original Set Cover instance. If point x_i belongs to set X_j , we make an edge from w_j to v_i with associated probability of 1. We set the influence threshold parameter p to $\frac{1}{2}$. We run edges from all nodes v_i to the output, all with associated probability = $1 - \frac{1}{2}^{1/k}$. Activating a node w_i corresponds to choosing the set X_i and activating a node v_i corresponds to choosing an arbitrary set X_j that contains x_i . The

output will fire with probability $\geq \frac{1}{2}$ only if all of the v_i 's fire. This completes the reduction. Because Set Cover is NP-Hard to approximate to within $o(\log n)$ [13], so is approximating the size of the smallest influential set. $\square$

Theorem 6. *Let S be a social network of size n and let I be the smallest set of nodes having influence p , where $m = |I|$. We can find a set of nodes of size $m \log(p/\epsilon)$ of influence $(p-\epsilon)$ using $O(nm \log(p/\epsilon))$ exact value injection queries.*

Proof. Consider Algorithm 3.

Algorithm 3. An Algorithm for Finding a Set of Influential Nodes

Let S be the target social network.

Let p be the threshold probability.

Let ϵ be the error tolerance.

Let $I = \emptyset$.

Let e_I be the experiment where all nodes in I are fired, and the rest are left free.

while $S(e_I) < p - \epsilon$ **do**

 Let $v = \arg \max_{v_j \in V} S(e_I|_{v_j=1})$

$I = I \cup \{v\}$

end while

Return I

Assume the optimal solution X , where $S(e_X) \geq p$, has size m . We claim that at any stage of the algorithm, if $S(e_I) < p - \epsilon$, greedily adding one more node w to I makes

$$S(e_{I \cup \{w\}}) \geq S(e_I) + \frac{p - S(e_I)}{m}.$$

We can see this by noting that there exists a set of at most m nodes, namely X , that will get the probability all the way to p . By Lemma 3, some node will get us at least $\frac{1}{m}$ th of the way there.

Let k be the number of rounds this algorithm is run. We look at the difference between p and $S(e_I)$ after k rounds. By the observation above, we can compute the number of rounds to get the difference to within ϵ . For

$$p \left(1 - \frac{1}{m}\right)^k < \epsilon$$

it suffices that $k > m \log\left(\frac{p}{\epsilon}\right)$. Therefore, after $m \log\left(\frac{p}{\epsilon}\right)$ rounds, $S(e_I)$ is within ϵ of p . We check $O(n)$ nodes each round, making for $O(nm \log\left(\frac{p}{\epsilon}\right))$ queries. $\square$

We now reconcile the algorithm and the hardness of approximation result. Given a social network created by the Set Cover reduction from Theorem 5, we can try to learn the influential nodes using Algorithm 3. If we set

$$\epsilon = \frac{1}{2}^{\frac{1}{n}} - \frac{1}{2}^{\frac{1}{n-1}} = \Theta\left(\frac{1}{n^2}\right),$$

this makes ϵ small enough to force the algorithm to cover all of the v_i 's. It would find a set of $(m \log(pn^2)) = O(m \log(n))$ nodes having influence p , which gives a

$O(\log(n))$ approximation and matches the lower bound. It is worth noting that the greedy algorithm for Set Cover also matches its hardness of approximation lower bound [15].

We will use Lemma 3, a version of which is derived in [10]. A function f is **submodular** if $f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B)$ whenever $A \subseteq B$.

Lemma 3. *$S(e_I)$ is a positive monotone, submodular function of I . [10]*

Corollary 1. *If p is the maximum influence of any k node set in the network, then Algorithm 3, with a threshold of 1, terminated after k steps, produces a set with influence $\geq (1 - \frac{1}{e})p$.*

Proof. Nemhauser et al. [12] show that greedily maximizing a non-negative, monotone, submodular function on sets gives a $(1 - \frac{1}{e})$ approximation to the function on k -element sets. Hence, this follows from Lemma 3. $\square$

7 Open Problems

We leave open a number of interesting and challenging problems. Our results rely on exact value injection queries. While these queries are theoretically elegant, in real-world applications learners would normally only have access to non-exact value injection queries, and for such queries our algorithms would need to be modified, mainly because we look for shortest paths, not necessarily the paths least diluted by the multiplication of probabilities. The Angluin et al. [2] results on probabilistic networks adapt some exact value injection query algorithms to work in non-exact settings, yet we see no clear way of similarly modifying our algorithms. Furthermore, in moving to the non-exact setting, because of the results from Section 5, either target network depth would need to be limited, or algorithms would have to be invented that do not rely on excitation paths.

Another interesting topic to explore is what other classes of cyclic networks can be learned using similar algorithms? Our algorithms rely on the independence assumption in the independent cascade social network model. However, there are other more general models of social networks, like the **decreasing cascade model** [10]. It would be worthwhile exploring their learnability as well.

Finally, it is often the case that graph algorithms run faster on sparse graphs. It would be interesting to design an algorithm for learning social networks whose query complexity was a function of the size of the edge set in the target graph.

References

- [1] Akutsu, T., Kuhara, S., Maruyama, O., Miyano, S.: Identification of gene regulatory networks by strategic gene disruptions and gene overexpressions. In: SODA 1998: Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, Philadelphia, PA, USA, pp. 695–702. Society for Industrial and Applied Mathematics (1998)
- [2] Angluin, D., Aspnes, J., Chen, J., Eisenstat, D., Reyzin, L.: Learning acyclic probabilistic circuits using test paths. In: COLT 2008 (to appear, 2008)

- [3] Angluin, D., Aspnes, J., Chen, J., Reyzin, L.: Learning large-alphabet and analog circuits with value injection queries. In: The 20th Annual Conference on Learning Theory, pp. 51–65 (2007)
- [4] Angluin, D., Aspnes, J., Chen, J., Wu, Y.: Learning a circuit by injecting values. In: Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing, pp. 584–593. ACM Press, New York (2006)
- [5] Angluin, D., Kharitonov, M.: When won't membership queries help? *J. Comput. Syst. Sci.* 50(2), 336–355 (1995)
- [6] Goldenberg, J., Libai, B., Muller, E.: Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review* (2001)
- [7] Ideker, T., Thorsson, V., Karp, R.: Discovery of regulatory interactions through perturbation: Inference and experimental design. In: Pacific Symposium on Bio-computing 5, pp. 302–313 (2000)
- [8] Kearns, M., Valiant, L.: Cryptographic limitations on learning boolean formulae and finite automata. *J. ACM* 41(1), 67–95 (1994)
- [9] Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. In: KDD 2003: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 137–146. ACM, New York (2003)
- [10] Kempe, D., Kleinberg, J.M., Tardos, É.: Influential nodes in a diffusion model for social networks. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 1127–1138. Springer, Heidelberg (2005)
- [11] Kharitonov, M.: Cryptographic hardness of distribution-specific learning. In: STOC 1993: Proceedings of the twenty-fifth annual ACM symposium on Theory of computing, pp. 372–381. ACM Press, New York (1993)
- [12] Nemhauser, G., Wolsey, L., F.M.: An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming* 14, 265–294 (1978)
- [13] Raz, R., Safra, S.: A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In: STOC 1997: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, pp. 475–484. ACM, New York (1997)
- [14] Rudin, W.: Principles of Mathematical Analysis. International Series in Pure and Applied Mathematics. McGraw-Hill, New York (1976)
- [15] Slavík, P.: A tight analysis of the greedy algorithm for set cover. In: STOC 1996: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, pp. 435–441. ACM, New York (1996)

Active Learning in Multi-armed Bandits

András Antos¹, Varun Grover², and Csaba Szepesvári^{1,2}

¹ Computer and Automation Research Institute
of the Hungarian Academy of Sciences
Kende u. 13-17, Budapest 1111, Hungary
`antos@szit.bme.hu`

² Department of Computing Science
University of Alberta, Edmonton T6G 2E8, Canada
`{varun,szepesva}@cs.ualberta.ca`

Abstract. In this paper we consider the problem of actively learning the mean values of distributions associated with a finite number of options (arms). The algorithms can select which option to generate the next sample from in order to produce estimates with equally good precision for all the distributions. When an algorithm uses sample means to estimate the unknown values then the optimal solution, assuming full knowledge of the distributions, is to sample each option proportional to its variance. In this paper we propose an incremental algorithm that asymptotically achieves the same loss as an optimal rule. We prove that the excess loss suffered by this algorithm, apart from logarithmic factors, scales as $n^{-3/2}$, which we conjecture to be the optimal rate. The performance of the algorithm is illustrated in a simple problem.

1 Introduction

Consider the problem of production quality assurance in a factory equipped with a number of machines that produce products of different quality. The quality can be monitored by inspecting the products produced: An inspection of a product is modeled as a random number say between zero and one, one meaning the best, zero the poorest quality. The outcome will depend on random effects influencing the production and how the inspection was done, but the main assumption is that the mean of this random variable characterizes the maintenance state of the machine. Due to the randomness of the inspection results, multiple measurements are necessary to control the precision of the quality estimates. We are interested in keeping the precision of the estimates equal across the machines. If the inspection of a product is expensive (as is the case when inspection requires the destruction of the product) then to keep the cost low, it is logical to inspect machines that produce products of highly varying inspection results more frequently. The problem is then to decide about exactly how frequently the quality of each machine should be checked by inspecting a product produced on it. The loss is measured by taking the largest of the mean-squared errors of the estimates produced for the machines.

The basic problem is to estimate unknown quantities corresponding to a finite number of options by sequentially drawing random variables from distributions associated with the options so as to keep the estimation error across all the options the same. Active learning problems involve estimating unknown parameters by selectively and adaptively sampling from the input space. Hence, this problem can be seen as an instance of *active learning*. The problem is also similar to *multi-armed bandit problems* [7, 2] in that only one option (arm) can be probed at any time. However, the performance criterion is different from that used in bandits where the observed values are treated as rewards and performance during learning is what matters. Nevertheless, we will see that the exploration-exploitation dilemma which characterizes classical bandit problems will still play a role here. Because of this connection we call this problem the *max-loss value-estimation problem in multi-armed bandits*.

The formal description of this problem is as follows: We are interested in estimating the expected values (μ_k) of some distributions ($\mathcal{D}_k$), each associated with an option (or arm). If K is the number of options then $1 \leq k \leq K$. For any k , the decision maker can draw independent samples $\{X_{kt}\}_t$ from $\mathcal{D}_k$. The sample X_{kt} is observed when a sample is requested from option k the t^{th} time. (These samples correspond to the outcomes of inspections in the previous example). The samples are drawn sequentially: Given the information collected up to trial n the decision maker can decide which option to choose next. At any time n , the decision maker keeps an estimate, $\hat{\mu}_{kn}$, of the mean of $\mathcal{D}_k$. The error of estimate k is measured with the expected squared error:

$$L_{kn} = \mathbb{E} [(\hat{\mu}_{kn} - \mu_k)^2].$$

The overall loss is measured by the worst-case loss over the K options:

$$L_n = \max_{1 \leq k \leq K} L_{kn}.$$

This expresses the desire that all estimates are equally important. The goal of the decision maker is to make this loss as small as possible.

For the sake of simplicity assume that the estimates $\hat{\mu}_{kn}$ are produced by computing the sample means of the respective options:

$$\hat{\mu}_{kn} = \frac{1}{T_{kn}} \sum_{t=1}^{T_{kn}} X_{kt},$$

where T_{kn} denotes the number of times a sample was requested from option k .

Consider the non-sequential version of the problem, i.e., the problem of choosing $T_{1n}, \dots, T_{Kn}$ such that $T_{1n} + \dots + T_{Kn} = n$ so as to minimize the loss. Let us assume for a moment full knowledge of the distributions, so there is no value in making this choice data dependent. Due to the independence of samples

$$L_{kn} = \frac{\sigma_k^2}{T_{kn}},$$

where $\sigma_k^2 = \text{Var}[X_{k1}]$. For simplicity assume that $\sigma_k^2 > 0$ holds for all k . It is not hard to see then that the minimizer of $L_n = \max_k L_{kn}$ is the allocation $\{T_{kn}^*\}_k$ that makes all the losses L_{kn} (approximately) equal, hence (apart from rounding issues)

$$T_{kn}^* = n \frac{\sigma_k^2}{\Sigma^2} = \lambda_k n.$$

Here $\Sigma^2 = \sum_{j=1}^K \sigma_j^2$ is the sum of the variances and

$$\lambda_k = \frac{\sigma_k^2}{\Sigma^2}.$$

The corresponding loss is

$$L_n^* = \frac{\Sigma^2}{n}.$$

The optimal allocation is easy to extend to the case when some options have zero variance. Clearly, it is both necessary and sufficient to make a single observation on such options. The case when all variances are zero (i.e., $\Sigma^2 = 0$) is uninteresting, hence we will assume from now on that $\Sigma^2 > 0$.

We expect a good sequential algorithm $\mathcal{A}$ to achieve a loss $L_n = L_n(\mathcal{A})$ close to the loss L_n^* . We will therefore look into the excess loss

$$R_n(\mathcal{A}) = L_n(\mathcal{A}) - L_n^*.$$

Since the loss of option k can only decrease if we request a new sample from $\mathcal{D}_k$, one simple idea is to request the next sample from option k whose estimated loss, $\hat{\sigma}_{kn}^2/T_{kn}$, is the largest amongst all estimated losses. Here $\hat{\sigma}_{kn}^2$ is an estimate of the variance of the k^{th} option based on the history. The problem with this approach is that the variance might be underestimated in which case the option will not be selected for a long time, which prevents refining the estimated variance, ultimately resulting in a large excess loss. Thus we face a problem similar to the exploration-exploitation dilemma in bandit problems where a greedy policy might incur a large loss if the payoff of the optimal option is underestimated. One simple remedy is to make sure that the estimated variances converge to their true values. This can be ensured if the algorithm is forced to select all the options indefinitely in the limit, which is often called the method of forced selections in the bandit literature. One way to implement this idea is to introduce phases of increasing length. Then in each phase the algorithm could choose all options exactly once at the beginning, while in the rest of the phase it can sample all the options k proportionally to their respective variance estimates computed at the beginning of the phase. The problem then becomes to select the appropriate phase lengths to make sure that the proportion of forced selections diminishes at an appropriate rate with an increasing horizon n . (An algorithm along these lines have been described and analyzed by [5] in the context of stratified sampling. We shall discuss this further in Section 5.) While the introduction of phases allows a direct control of the proportion of forced selections, the algorithm is not incremental and is somewhat cumbersome to implement.

In this paper we propose and study an alternative algorithm that implements forced selections but remains completely incremental. The idea is to select the option with the largest estimated loss except if some of the options is seriously under-sampled, in which case the under-sampled option is selected. It turns out that a good definition for an option being under-sampled is $T_{kn} \leq c\sqrt{n}$ with some constant $c > 0$. (The algorithm will be formally stated in the next section.) We will show that the excess loss of this algorithm decreases with n as $\tilde{O}(n^{-3/2})$.¹

2 Algorithm

The formal description of the algorithm, that we call GAFS-MAX (greedy allocation with forced selections for max-norm value estimation), is as follows:

Algorithm GAFS-MAX

In the first K trials choose each arm once

Set $T_{k,K+1} = 1$ ($1 \leq k \leq K$), $n = K + 1$

At time n do:

Compute $\hat{\sigma}_{kn}^2 = \frac{1}{T_{kn}} \sum_{t=1}^{T_{kn}} X_{kt}^2 - \left(\frac{1}{T_{kn}} \sum_{t=1}^{T_{kn}} X_{kt} \right)^2$

Let $\hat{\lambda}_{kn} = \hat{\sigma}_{kn}^2 / (\sum_{j=1}^K \hat{\sigma}_{jn}^2)$ if $\sum_{j=1}^K \hat{\sigma}_{jn}^2 \neq 0$,
otherwise let $\hat{\lambda}_{kn} = 1/K$.

Let $U_n = \{k : T_{kn} < \sqrt{n} + 1\}$

Let

$$I_{n+1} = \begin{cases} \min U_n, & \text{if } U_n \neq \emptyset \\ \operatorname{argmax}_{1 \leq k \leq K} \frac{\hat{\lambda}_{kn}}{T_{kn}}, & \text{otherwise,} \end{cases}$$

where in the second case ties are broken in an arbitrary,
but systematic manner.

Choose option I_{n+1} , let $T_{k,n+1} = T_{k,n} + \mathbb{I}\{I_{n+1} = k\}$

Observe the feedback $X_{I_{n+1}, T_{I_{n+1}, n+1}}$.

Of course, the variance estimates can be computed incrementally. Further, it is actually not necessary to compute $\hat{\lambda}_{kn}$ because in the computation of the arm index $\hat{\lambda}_{kn}$ can be replaced by $\hat{\sigma}_{kn}^2$ without effecting the choices.

3 Main Results

The main result (Theorem 3) for GAFS-MAX is a bound of the form $L_n \leq L_n^* + \tilde{O}(n^{-3/2})$. We also prove high probability bounds on $T_{nk}/n - \lambda_k$ (Theorem 1). The proof is somewhat involved, hence we start with an outline: Clearly, the rate of growth of T_{kn} controls the rate of convergence of $\hat{\lambda}_{kn}$ to λ_k . In particular, we will show that given $T_{kn} \geq f(n)$ it follows that $\hat{\lambda}_{kn}$ converges to λ_k at a rate

¹ A nonnegative sequence (a_n) is said to be $\tilde{O}(f(n))$, where $f : \mathbb{N} \rightarrow \mathbb{R}^+$, if $a_n \leq Cf(n) \log(n)$ with a suitable constant $C > 0$.

of $O(1/f(n)^{1/2})$ (Lemma 2). The second major tool is a result (Lemma 3) that shows how a faster rate for $\hat{\lambda}_{kn}$ transforms into better bounds on T_{kn} . The actual proof is then started by observing that due to the forced selections $T_{kn} \geq \sqrt{n}$.

The proof is developed through a series of Lemmata. First, we state Hoeffding's inequality in a form that suits the best our needs:

Lemma 1 (Hoeffding's inequality, [6]). *Let Z_t be a sequence of zero-mean, i.i.d. random variables, where $a \leq Z_t \leq b$, $a < b$ reals. Then, for any $0 < \delta \leq 1$,*

$$\mathbb{P}\left(\frac{1}{n} \sum_{t=1}^n Z_t \geq \sqrt{\frac{1}{2} \frac{(b-a)^2}{n} \log(1/\delta)}\right) \leq \delta.$$

Let

$$\Delta(R^2, n, \delta) = R \sqrt{\frac{\log(1/\delta)}{2n}}.$$

Let $\mu_k^{(2)} = \mathbb{E}[X_{kt}^2]$, R_k be the size of the range of the random variables $\{X_{kt}\}_t$ (i.e., $|\text{supp}(X_{kt})| \leq R_k$), S_k be the size of the range of the random variables $\{X_{kt}^2\}_t$, and B_k be the size of the range of the random variables $\{|X_{kt}|\}_t$. Note that $B_k \leq R_k$ and $S_k \leq B_k^2$. Let

$$A_\delta = \left(\bigcap_{1 \leq k \leq K, n \geq 1} \left\{ \left| \frac{1}{n} \sum_{t=1}^n X_{kt}^2 - \mu_k^{(2)} \right| \leq \Delta(S_k^2, n, \delta_n) \right\} \right) \cap \bigcap_{1 \leq k \leq K, n \geq 1} \left\{ \left| \frac{1}{n} \sum_{t=1}^n X_{kt} - \mu_k \right| \leq \Delta(R_k^2, n, \delta_n) \right\},$$

where $\delta_n = \delta/(4K(n(n+1)))$. Note that δ_n is chosen such that $\sum_{k=1}^K \sum_{n=1}^\infty \delta_n = \delta/4$. Hence, we observe that by Hoeffding's inequality

$$\mathbb{P}(A_\delta) \geq 1 - \delta.$$

The sets $\{A_\delta\}_\delta$ will play a key role in the proof: Many of the statements will be proved on these set.

Our first result connects a lower bound on T_{kn} to the rate of convergence of $\hat{\lambda}_{kn}$. Let $a_k = |\mu_k| + B_k$, $b_k = S_k + a_k R_k$, and $a'_k = \sigma_k^4/(4b_k^2)$.

Lemma 2. *Fix $0 < \delta \leq 1$ and $n_0 > 0$, and assume that for $n \geq n_0$, $1 \leq k \leq K$, $T_{kn} \geq f(n) \geq 2$ holds on A_δ , where $f(n) \rightarrow \infty$. Then there exists constants $N_0 \geq n_0$ and $c > 0$ such that for any $n \geq N_0$, $1 \leq k \leq K$, on A_δ*

$$|\hat{\lambda}_{kn} - \lambda_k| \leq c \sqrt{\frac{\log(\delta_n^{-1})}{f(n)}} \quad (1)$$

holds. In particular, $c = \sqrt{2}(b_k + \lambda_k \sum_{j=1}^K b_j)/\Sigma^2 \leq 5\sqrt{2}(B_k^2 + \sum_{j=1}^K B_j^2)/\Sigma^2$.

If $f(n) = bn^p$ ($p > 0$) then $N_0 = \max(n_0, n_1)$, where n_1 is a number such that for $n \geq n_1$

$$\log n \leq \frac{ba'_k}{p} n^p - \frac{1 + \log(\frac{4K}{\delta}) + 2 \log b}{2p}. \quad (2)$$

Proof. First, we develop a bound on $|\hat{\sigma}_{kn}^2 - \sigma_k^2|$. Let $\hat{\mu}_{kn}^{(2)} = 1/T_{kn} \sum_{t=1}^{T_{kn}} X_{kt}^2$ and $\hat{\mu}_{kn} = 1/T_{kn} \sum_{t=1}^{T_{kn}} X_{kt}$. Consider any element of A_δ . Then by the definition of A_δ , $|1/m \sum_{t=1}^m X_{kt}^2 - \mu_k^{(2)}| \leq \Delta(S_k^2, m, \delta_m)$ holds simultaneously for any $m \geq 1$. Hence, for $n \geq n_0$ it also holds that

$$\left| \frac{1}{T_{kn}} \sum_{t=1}^{T_{kn}} X_{kt}^2 - \mu_k^{(2)} \right| \leq \Delta(S_k^2, T_{kn}, \delta_{T_{kn}}) \leq \Delta(S_k^2, f(n), \delta_{f(n)}),$$

where we have used that $\log(x(x+1)/\delta)/x$ is monotonically decreasing when $x \geq 2$ and $T_{kn} \geq f(n) \geq 2$. Similarly, we get that

$$\left| \frac{1}{T_{kn}} \sum_{t=1}^{T_{kn}} X_{kt} - \mu_k \right| \leq \Delta(R_k^2, f(n), \delta_{f(n)}).$$

Using $\hat{\sigma}_{kn}^2 = \hat{\mu}_{kn}^{(2)} - \hat{\mu}_{kn}^2$ and $\sigma_k^2 = \mathbb{E}[X_{kt}^2] - (\mathbb{E}[X_{kt}])^2 = \mu_k^{(2)} - \mu_k^2$, we get

$$\begin{aligned} |\hat{\sigma}_{kn}^2 - \sigma_k^2| &\leq |\hat{\mu}_{kn}^{(2)} - \mu_k^{(2)}| + |\hat{\mu}_{kn}^2 - \mu_k^2| \\ &\leq \Delta(S_k^2, f(n), \delta_{f(n)}) + \Delta(R_k^2, f(n), \delta_{f(n)})(|\mu_k| + B_k), \end{aligned} \quad (3)$$

where we used $|a^2 - b^2| \leq |a - b|(|a| + |b|)$.

Denote the right-hand side of (3) by $\Delta_k(n, \delta)$. Now, let us develop a lower bound on $\hat{\lambda}_{kn}$ in terms of λ_k . Then, for $n \geq n_0$,

$$\hat{\lambda}_{kn} = \frac{\hat{\sigma}_{kn}^2}{\sum_{j=1}^K \hat{\sigma}_{jn}^2} \geq \frac{\sigma_k^2 - \Delta_k(n, \delta)}{\Sigma^2 + \sum_{j=1}^K \Delta_j(n, \delta)} \geq \lambda_k \left(1 - \frac{\sum_{j=1}^K \Delta_j(n, \delta)}{\Sigma^2} \right) - \frac{\Delta_k(n, \delta)}{\Sigma^2},$$

where we used $1/(1+x) \geq 1-x$ that holds for $x > -1$.

An upper bound can be obtained analogously: For $n \geq n_0$, if

$$\Sigma^2 \geq 2 \sum_{j=1}^K \Delta_j(n, \delta) \quad (4)$$

then

$$\hat{\lambda}_{kn} = \frac{\hat{\sigma}_{kn}^2}{\sum_{j=1}^K \hat{\sigma}_{jn}^2} \leq \frac{\sigma_k^2 + \Delta_k(n, \delta)}{\Sigma^2 - \sum_{j=1}^K \Delta_j(n, \delta)} \leq \lambda_k \left(1 + 2 \frac{\sum_{j=1}^K \Delta_j(n, \delta)}{\Sigma^2} \right) + 2 \frac{\Delta_k(n, \delta)}{\Sigma^2},$$

where we used $1/(1-x) \leq 1+2x$ that holds for $0 \leq x \leq 1/2$. This constraint follows from (4), that is implied if n is big enough so that

$$2\Delta_j(n, \delta) \leq \sigma_j^2, \quad 1 \leq j \leq K. \quad (5)$$

The upper and lower bounds above together give

$$|\hat{\lambda}_{kn} - \lambda_k| \leq \frac{2}{\Sigma^2} \left(\lambda_k \sum_{j=1}^K \Delta_j(n, \delta) + \Delta_k(n, \delta) \right).$$

Noting that $\Delta_j(n, \delta)$ equals to

$$(S_j + R_j(|\mu_j| + B_j)) \sqrt{\frac{\log(\delta_{f(n)}^{-1})}{2f(n)}} = b_j \sqrt{\frac{\log(\delta_{f(n)}^{-1})}{2f(n)}}, \quad (6)$$

where $b_j = S_j + R_j(|\mu_j| + B_j)$, we get

$$|\hat{\lambda}_{kn} - \lambda_k| \leq \frac{\sqrt{2}}{\Sigma^2} \left(\lambda_k \sum_{j=1}^K b_j + b_k \right) \sqrt{\frac{\log(\delta_{f(n)}^{-1})}{f(n)}}.$$

Since $f(n) \leq T_{kn} \leq n$, $\delta_{f(n)}^{-1}$ can be upper bounded by δ_n^{-1} leading to (1).

At last, to satisfy (5), by (6), it suffices if

$$f(n) \geq \frac{2b_j^2}{\sigma_j^4} \log(\delta_{f(n)}^{-1}) = \frac{2b_j^2}{\sigma_j^4} (\log(f(n)(f(n) + 1)) + \log(4K/\delta))$$

that is guaranteed by $f(n) \rightarrow \infty$ for n large enough.

If $f(n) = bn^p$ then $bn^p \geq \frac{2b_j^2}{\sigma_j^4} (2p \log n + 2 \log b + 1 + \log(4K/\delta))$ will ensure that. Reordering this gives (2). $\square$

Now we show how a rate of convergence result for $\hat{\lambda}_{kn}$ can be turned into bounds on $T_{kn}/n - \lambda_k$. Let $\lambda_{\min} = \min_{1 \leq j \leq K} \lambda_j$. In what follows, unless otherwise stated, we will assume that $\lambda_{\min} > 0$.

Lemma 3. Fix $0 < \delta \leq 1$ and $n_0 > 0$. Assume that $f(n) \leq n$ such that $f(n)/n^2$ is monotone decreasing, and consider an event such that

$$|\hat{\lambda}_{kn} - \lambda_k| \leq c \sqrt{\log(\delta_n^{-1})/f(n)}, \quad 1 \leq k \leq K \quad (7)$$

holds with some $c \geq 1$, for all $n \geq n_0$. Let

$$H(n, \delta) = c \left(1 + \frac{2}{\lambda_{\min}} \right) n \sqrt{\frac{\log(\delta_n^{-1})}{f(n)}}.$$

Then the following inequalities hold for $n \geq n_0$ and $1 \leq k \leq K$:

$$\begin{aligned} T_{kn} &\leq n\lambda_k + \max(n_0, 1 + H(n, \delta)), \\ T_{kn} &\geq n\lambda_k - (K - 1) \max(n_0, 1 + H(n, \delta)). \end{aligned}$$

Proof. By definition $T_{k,n+1} = T_{kn} + \mathbb{I}\{I_{n+1} = k\}$. Let $E_{kn} = T_{kn} - n\lambda_k$. Note that

$$\sum_{k=1}^K E_{kn} = 0 \quad (8)$$

holds for any $n \geq 1$. Notice that the desired result can be stated as bounds on E_{kn} . Hence, our goal now is to study E_{kn} . If b_{jn} is an upper bound for E_{jn} ($1 \leq j \leq K$) then from (8) we get the lower bound $E_{kn} = -\sum_{j \neq k} E_{jn} \geq -\sum_{j \neq k} b_{jn} \geq -(K-1) \max_j b_{jn}$. Hence, we target upper bounds on $\{E_{kn}\}_k$.

From the definition of E_{kn} and T_{kn} we get

$$E_{k,n+1} = E_{k,n} - \lambda_k + \mathbb{I}\{I_{n+1} = k\}.$$

By the definition of the algorithm

$$\mathbb{I}\{I_{n+1} = k\} \leq \mathbb{I}\left\{T_{kn} \leq \lceil \sqrt{n} \rceil \text{ or } k = \underset{1 \leq j \leq K}{\operatorname{argmin}} \frac{T_{jn}}{\hat{\lambda}_{jn}}\right\},$$

with the understanding that $c/0 = +\infty$. Assume now that k is an index where $\{\frac{T_{jn}}{\hat{\lambda}_{jn}}\}_j$ takes its minimum, that is,

$$\frac{T_{kn}}{\hat{\lambda}_{kn}} \leq \min_j \frac{T_{jn}}{\hat{\lambda}_{jn}}.$$

Using $T_{jn} = E_{jn} + n\lambda_j$ and reordering the terms gives

$$E_{kn} + n\lambda_k \leq \hat{\lambda}_{kn} \min_j \frac{E_{jn} + n\lambda_j}{\hat{\lambda}_{jn}} \leq \hat{\lambda}_{kn} \left(\min_j \frac{E_{jn}}{\hat{\lambda}_{jn}} + n \max_j \frac{\lambda_j}{\hat{\lambda}_{jn}} \right).$$

By (8), there exists an index j such that $E_{jn} \leq 0$. Since $\hat{\lambda}_{jn} \geq 0$ for any j , it holds that $\min_j \frac{E_{jn}}{\hat{\lambda}_{jn}} \leq 0$. Hence, $E_{kn} + n\lambda_k \leq n\hat{\lambda}_{kn} \max_j \frac{\lambda_j}{\hat{\lambda}_{jn}}$. Using (7) and $1/(1-x) = 1 + x/(1-x) \leq 1 + 2x$, which holds for $x \leq 1/2$, provided that $n \geq n_0$, we get

$$\frac{\lambda_j}{\hat{\lambda}_{jn}} \leq \frac{\lambda_j}{\lambda_j - c\sqrt{\log(\delta_n^{-1})/f(n)}} \leq 1 + \frac{2c}{\lambda_j} \sqrt{\frac{\log(\delta_n^{-1})}{f(n)}}.$$

Using $\hat{\lambda}_{kn} \leq 1$ and (7) again,

$$E_{kn} \leq n(\hat{\lambda}_{kn} - \lambda_k) + \frac{2cn}{\lambda_{\min}} \sqrt{\frac{\log(\delta_n^{-1})}{f(n)}} \leq c \left(1 + \frac{2}{\lambda_{\min}}\right) n \sqrt{\frac{\log(\delta_n^{-1})}{f(n)}}.$$

Note that the right-hand side is $H(n, \delta)$. Hence,

$$\mathbb{I}\{I_{n+1} = k\} \leq \mathbb{I}\{T_{kn} \leq \lceil \sqrt{n} \rceil \text{ or } E_{kn} \leq H(n, \delta)\}.$$

Assume now that $T_{kn} \leq \lceil \sqrt{n} \rceil$. We want to show that in this case $E_{kn} \leq H(n, \delta)$. By the definition of E_{kn} , from $T_{kn} \leq \lceil \sqrt{n} \rceil$ it follows that $E_{kn} = T_{kn} - n\lambda_k \leq \lceil \sqrt{n} \rceil \leq \sqrt{2n}$. Hence, $E_{kn} \leq H(n, \delta)$ follows if $\sqrt{2n} \leq H(n, \delta)$. In particular, this follows from the bounds on c , $\lambda_{\min}$, $f(n)$, and δ . Therefore

$$\mathbb{I}\{I_{n+1} = k\} \leq \mathbb{I}\{E_{kn} \leq H(n, \delta)\}.$$

We need the following technical lemma:

Lemma 4. *Let $0 \leq \lambda \leq 1$. Consider the sequences $E_n, \tilde{E}_n, I_n, \tilde{I}_n$ ($n \geq 1$) where $I_n, \tilde{I}_n \in 0, 1$, $E_{n+1} = E_n + I_n - \lambda$, $\tilde{E}_{n+1} = \tilde{E}_n + \tilde{I}_n - \lambda$, $\tilde{E}_1 = E_1$ and assume that $I_n \leq \tilde{I}_n$ holds whenever $E_n = \tilde{E}_n$. Then $E_n \leq \tilde{E}_n$ holds for $n \geq 1$.*

Due to the lack of space we only sketch the proof of Lemma 4. The idea is that $P_n = \tilde{E}_n - E_n$ can only take on integer values and step 0 or 1. Then $P_n \geq 0$, $n \geq 1$ follows since $P_1 = 0$ and when in $P_n = 0$ then $P_{n+1} \geq 0$. Now, returning to the proof of Lemma 3, define $\tilde{E}_{kn}$ by

$$\begin{aligned}\tilde{E}_{k,n+1} &= \tilde{E}_{k,n} - \lambda_k + \mathbb{I}\left\{\tilde{E}_{kn} \leq H(n, \delta)\right\}, \quad n \geq n_0, \\ \tilde{E}_{k,n_0} &= E_{k,n_0}.\end{aligned}$$

The conditions of Lemma 4 are clearly satisfied from index n_0 . Consequently $E_{k,n} \leq \tilde{E}_{k,n}$ holds for any $n \geq n_0$. Further, since $H(n, \delta)$ is monotone increasing in n , $\tilde{E}_{k,n} \leq \max(E_{k,n_0}, 1 + H(n, \delta)) \leq \max(n_0, 1 + H(n, \delta))$, finishing the upper-bound. $\square$

Using the previous result we are now in the position to prove a linear lower bound on T_{kn} :

Lemma 5. *Let $0 < \delta \leq 1$ arbitrary. Then there exists an integer N_1 such that for any $n \geq N_1$, $T_{kn} \geq n\lambda_k/2$ holds on A_δ .*

In particular,

$$N_1 = \max\left(\frac{2(K-1)}{\lambda_{\min}} \max(3, N_0), D_2^2 \left[\log D_2^2 + \frac{1}{2} \left(\log\left(\frac{4K}{\delta}\right) + 1\right)\right]^2\right), \quad (9)$$

where $N_0 = \max\left(K^2, (1/a'_k)^2 [\log((1/a'_k)^2) + (1 + \log(4K/\delta))]\right)^2$ and $D_2 = 4(9c(K-1))^2/\lambda_{\min}^4$.

For the proof we need the following technical lemma that quantifies the point when for $a > 0$ the function $at^{1/2} + b$ overtakes $\log t$.

Lemma 6. *Let $q(t) = at^{1/2} + b$, $\ell(t) = \log t$, where $a > 0$. Then for any $t \geq (2/a)^2 [\log((2/a)^2) - b]^2$, $q(t) \geq \ell(t)$.*

The proof of this lemma is elementary and is hence omitted.

Proof (Lemma 5). Due to the forced selection of the options built into the algorithm, $T_{kn} \geq \sqrt{n}$ holds for $n \geq K^2$. Hence, we can apply Lemma 2 with $f(n) = n^{1/2}$. By Lemma 6, n_1 defined by (2) can be chosen to be

$$(1/a'_k)^2 [\log((1/a'_k)^2) + (1 + \log(4K/\delta))]^2.$$

Hence, for $n \geq N_0 = \max(K^2, n_1)$ and $c > 0$ as defined in Lemma 2, we get,

$$|\hat{\lambda}_{kn} - \lambda_k| \leq c \sqrt{\frac{\log(\delta_n^{-1})}{n^{1/2}}}. \quad (10)$$

Possibly replacing c with $\max(c, 1)$, we can assume that $c \geq 1$. By Lemma 3, for $n \geq \max(N_0, 1/\lambda_{\min})$, $T_{kn} \geq n\lambda_k - (K-1)\max(N_0, 1 + H(n, \delta))$, and $H(n, \delta) = D_1 n^{3/4} \sqrt{\log(\delta_n^{-1})}$, where $D_1 = c \left(1 + \frac{2}{\lambda_{\min}}\right) \leq 3c/\lambda_{\min}$. Hence, $T_{kn} \geq n\lambda_k/2$ by the time when $n \geq 2N_0(K-1)/\lambda_{\min}$ and $n \geq 2(K-1)(1 + H(n, \delta))/\lambda_{\min}$. Lemma 6 and some tedious calculations then show that these two constrained are satisfied when $n \geq N_1$, where N_1 is defined as in equation (9). $\square$

With the help of this result we can get better bounds on T_{kn} , resulting in our first main result:

Theorem 1. *Let $0 < \delta \leq 1$ be arbitrary. Then there exists an integer N_2 and a positive real number D_3 such that for any $n \geq N_2$,*

$$-(K-1) \frac{\max(N_2, 1 + G(n, \delta))}{n} \leq \frac{T_{kn}}{n} - \lambda_k \leq \frac{\max(N_2, 1 + G(n, \delta))}{n}$$

holds on A_δ , where

$$G(n, \delta) = D_3 \sqrt{n \log(\delta_n^{-1})}. \quad (11)$$

Here $D_3 \leq 3\sqrt{2}c/\lambda_{\min}^{3/2}$,

$$N_2 = \max \left(N_1, \left(\frac{4}{\lambda_{\min} a'_k} \right) \left[\log \left(\frac{2}{\lambda_{\min} a'_k} \right) + \frac{1}{2} + \frac{1}{2} \log \left(\frac{4K}{\delta} \right) \right] \right),$$

where N_1 is defined in Lemma 5.

The theorem shows that asymptotically the GAFS-MAX algorithm behaves the same way as an optimal allocation rule that knows the variances. It also shows that the deviation of the proportion of choices of any option from the optimal value decays as $\tilde{O}(1/\sqrt{n})$.

For the proof we need the counterpart of Lemma 6 for linear functions:

Lemma 7. *Let $q(t) = at + b$, $\ell(t) = \log t$, where $a > 0$. Then for any $t \geq (2/a)(\log((1/a)) - b)$, $q(t) \geq \ell(t)$.*

Proof (Theorem 1). The proof is almost identical to that of Lemma 5. The difference is that now we start with a better lower bound on T_{kn} . In particular, by Lemma 5 $T_{kn} \geq n\lambda_k/2$ holds whenever $n \geq N_1$. By Lemma 2, for some $N_2 \geq N_1$, $c \geq 1$,

$$\left| \hat{\lambda}_{kn} - \lambda_k \right| \leq \frac{c}{\lambda_k^{1/2}} \sqrt{\frac{\log(\delta_n^{-1})}{n}} \quad (12)$$

holds for all $n \geq N_2$. In particular, solving (2) for n_1 with $f(n) = n\lambda_k/2$ and Lemma 7 give that

$$N_2 = \max \left(N_1, \frac{4}{\lambda_{\min} a'_k} \left[\log \left(\frac{2}{\lambda_{\min} a'_k} \right) + \frac{1}{2} + \frac{1}{2} \log \left(\frac{4K}{\delta} \right) \right] \right)$$

will suffice. By Lemma 3, for $n \geq \max(N_2, \lambda_{\min}^{-1}) = N_2$,

$$\begin{aligned} T_{kn} &\leq n\lambda_k + \max(N_2, 1 + G(n, \delta)), \quad \text{and} \\ T_{kn} &\geq n\lambda_k - (K - 1) \max(N_2, 1 + G(n, \delta)), \end{aligned}$$

where $G(n, \delta)$ is given by (11), and $D_3 = \sqrt{\frac{c}{\lambda_{\min}}} c \left(1 + \frac{2}{\lambda_{\min}} \right)$. $\square$

This result yields a bound on the expected value of $\mathbb{E}[T_{kn}]$:

Theorem 2. *Let N'_2 be such that $N_2 \leq N'_2 \log^2(4K/\delta)$ holds for any $\delta > 0$, where N_2 is defined in Theorem 1. Then, there exists an index N_3 that depends only on N'_2 , D_3 and K , such for any $n \geq N_3$,*

$$\mathbb{E}[T_{kn}] \leq n\lambda_k + D_3 \sqrt{n(1 + \log(4Kn(n+1)))} + 2. \quad (13)$$

Proof. First note that N'_2 exists and $N_2 \leq N'_2 \log^2(\delta_n^{-1})$ holds for any $n \geq 2$. Fix $0 < \delta \leq 1$. If $n \geq N_2^2/(D_3^2 \log(\delta_n^{-1}))$, then $1 + G(n, \delta) \geq N_2$, thus it follows from Theorem 1 that for $n \geq \max(N_2, N_2^2/(D_3^2 \log(\delta_n^{-1})))$,

$$\mathbb{P} \left(\frac{T_{kn} - n\lambda_k - 1}{D_3 n^{1/2}} > \sqrt{\log(\delta_n^{-1})} \right) \leq \delta$$

where we used $\mathbb{P}(A_\delta) \geq 1 - \delta$. Let $Z = (T_{kn} - n\lambda_k - 1)/(D_3 n^{1/2})$ and $\epsilon = \sqrt{\log(\delta_n^{-1})}$. The above inequality is equivalent to

$$\mathbb{P}(Z > \epsilon) \leq 4Kn(n+1)e^{-\epsilon^2}.$$

By the constraints that connect n and δ , this inequality holds for any pair (n, ϵ) that satisfy

$$n \geq \max(N'_2 \log^2(\delta_n^{-1}), N_2'^2 \log^3(\delta_n^{-1})/D_3^2) = \max(N_2'^4 \epsilon^4, N_2'^2 \epsilon^6/D_3^2),$$

that is, for any (n, ϵ) such that

$$\epsilon \leq \min((n/N_2')^{1/4}, (nD_3^2/N_2'^2)^{1/6}).$$

Also, since $Z \leq n^{1/2}/D_3$ is always true, $\mathbb{P}(Z > \epsilon) = 0$ holds for $\epsilon \geq n^{1/2}/D_3$. We need the following technical lemma, a variant of which can be found, e.g., as Exercise 12.1 in [4]:

Lemma 8. *If $\mathbb{P}(Z > \varepsilon) \leq C \exp(-c\varepsilon^2)$ for any $\varepsilon \leq a$, $a > 0$, and $\mathbb{P}(Z > \varepsilon) = 0$ for any $\varepsilon \geq b$ ($\geq a$), then*

$$\mathbb{E}[Z] \leq \sqrt{(1 + \log C)/c + Cb^2e^{-ca^2}}. \quad (14)$$

Due to the lack of space the proof is omitted.

Applying Lemma 8 with $a = \min((n/N_2')^{1/4}, (nD_3^2/N_2'^2)^{1/6})$, and $b = n^{1/2}/D_3$, $C = 4Kn(n+1)$, $c = 1$,

$$\mathbb{E}[Z] \leq \sqrt{1 + \log(4Kn(n+1)) + 4Kn^2(n+1)e^{-\min((n/N_2')^{1/2}, (nD_3^2/N_2'^2)^{1/3})/D_3^2}}.$$

Equation (13) then follows by straightforward algebra.

In order to develop a bound on the loss $L_{n,k}$ we need Wald's (second) identity:

Lemma 9 (Wald's Identity, Theorem 13.2.14 of [1]). *Let $\{\mathcal{F}_t\}_t$ be a filtration and let Y_t be an $\mathcal{F}_t$ -adapted sequence of i.i.d. random variables. Assume that $\mathcal{F}_t$ and $\sigma(\{Y_s : s \geq t+1\})$ are independent and T is a stopping time w.r.t. $\mathcal{F}_t$ with a finite expected value: $\mathbb{E}[T] < +\infty$. Consider the partial sums $S_n = Y_1 + \dots + Y_n$, $n \geq 1$. If $\mathbb{E}[Y_1^2] < +\infty$ then*

$$\mathbb{E}[(S_T - T\mathbb{E}[Y_1])^2] = \text{Var}[Y_1] \mathbb{E}[T]. \quad (15)$$

The following theorem is the main result of the paper:

Theorem 3. *Fix k , $n \geq N_2$, where N_2 is as in Theorem 1. Then*

$$L_n \leq L_n^* + \tilde{O}(n^{-3/2}).$$

Proof. Let $S_{kn} = \sum_{t=1}^n X_{kt}$, $\hat{L}_{kn} = (S_{k,T_{kn}} - T_{kn}\mu_k)/T_{kn}$, $G'(n, \delta) = (K - 1)\max(N_2, 1 + G(n, \delta))$ and

$$G''(n) = D_3\sqrt{n(1 + \log(4Kn(n+1)))} + 2.$$

Note that by Theorem 1,

$$\mathbb{P}(T_{kn} \leq n\lambda_k - G'(n, \delta)) \leq P(n, \delta) \triangleq \mathbb{I}\{n < N_2\} + \mathbb{I}\{n \geq N_2\}\delta \quad (16)$$

holds for any $n \geq 1$ and $0 < \delta \leq 1$. Then, for any $0 < \delta \leq 1$,

$$\begin{aligned} L_{kn} &= \mathbb{E}[\hat{L}_{kn}^2] \\ &= \mathbb{E}[\hat{L}_{kn}^2 \mathbb{I}\{T_{kn} > n\lambda_k - G'(n, \delta)\}] + \mathbb{E}[\hat{L}_{kn}^2 \mathbb{I}\{T_{kn} \leq n\lambda_k - G'(n, \delta)\}] \\ &\leq \frac{\mathbb{E}[(S_{k,T_{kn}} - T_{kn}\mu_k)^2]}{(n\lambda_k - G'(n, \delta))^2} + R^2 \mathbb{P}(T_{kn} \leq n\lambda_k - G'(n, \delta)) \\ &= \frac{\sigma_k^2 \mathbb{E}[T_{kn}]}{(n\lambda_k - G'(n, \delta))^2} + R^2 \mathbb{P}(T_{kn} \leq n\lambda_k - G'(n, \delta)) \quad (\text{by Lemma 9}) \\ &= \frac{\sigma_k^2 \mathbb{E}[T_{kn}]}{(n\lambda_k - G'(n, \delta))^2} + R^2 P(n, \delta) \quad (\text{by (16)}) \\ &\leq \frac{\sigma_k^2 (n\lambda_k + G''(n))}{(n\lambda_k - G'(n, \delta))^2} + R^2 P(n, \delta) \quad (\text{by 13}) \\ &= \frac{\sigma_k^2}{n\lambda_k} \frac{1}{(1 - G'(n, \delta)/(n\lambda_k))^2} + \frac{\sigma_k^2 G''(n)}{(n\lambda_k - G'(n, \delta))^2} + R^2 P(n, \delta). \end{aligned}$$

Now choose $\delta = n^{-3/2}$. Then, for n sufficiently large, $G'(n, n^{-3/2})/(n\lambda_k) \leq 1/2$. Further, since $N_2 \leq N'_2 \log(4K/\delta)$, for n sufficiently large $\mathbb{I}\{n < N_2\} \leq \mathbb{I}\{n < N'_2 \log(4Kn^{3/2})\} = 0$ and thus $P(n, \delta) = \delta$.

Therefore, for n sufficiently large, using $1/(1-x) \leq 1+2x$ ($|x| \leq 1/2$) we get,

$$L_{kn} \leq \frac{\sigma_k^2}{n\lambda_k} \left(1 + 2\frac{G'(n, n^{-3/2})}{n\lambda_k}\right)^2 + \frac{\sigma_k^2 G''(n)}{(n\lambda_k - G'(n, n^{-3/2}))^2} + R^2 n^{-3/2},$$

which gives

$$L_{kn} \leq \frac{\sigma_k^2}{n\lambda_k} + \tilde{O}(n^{-3/2}) = \frac{\Sigma^2}{n} + \tilde{O}(n^{-3/2}) = L_n^* + \tilde{O}(n^{-3/2}).$$

Taking the maximum with respect to k yields the desired result. $\square$

With a little extra work the case when for some options $\lambda_k = 0$ can also be handled and we can get identical bounds. Due to the lack of space this is not considered here.

4 Illustration

In addition to theory, empirical experiments show that our method indeed performs better than the non-adaptive solution. Further, our experiments verified that the allocation strategy found by our algorithm converges to the optimal allocation strategy at the rate predicted by the theory.

Here we illustrate the behavior of these algorithms in a simple problem with $K = 2$, with the random responses modeled as Bernoulli random variables for each of the options. In order to estimate the expected squared loss between the true mean and the estimated mean we repeat the experiment 100,000 times, then take the average. The error bars shown on the graphs show the standard deviations of these averages. The algorithms compared are GAFS-MAX (the algorithm studied here), GFSP-MAX (the algorithm described in the introduction that works in phases) and “UNIF”, the uniform allocation rule. In order for an adaptive algorithm to have any advantage the two options have to have different variances. For this purpose we chose $p_1 = 0.8$, $p_2 = 0.9$ so that $\lambda_1 = 0.64$ and $\lambda_2 = 0.36$.

Figure 1 shows the rescaled excess loss, $n^{3/2}(L_n - L_n^*)$, for the three algorithms. We see that the rescaled excess losses of the adaptive algorithms stay bounded, as predicted by the theory, while the rescaled loss of the uniform sampling strategy grows as $\sqrt{n}$. It is remarkable that the limit of the rescaled loss seems to be a small number, showing the efficiency of the algorithm. Note that this example shows that the uniform allocation initially performs better than the adaptive rules. This is because the adaptive algorithms need to get a good estimate of the statistics before they can start exploiting.

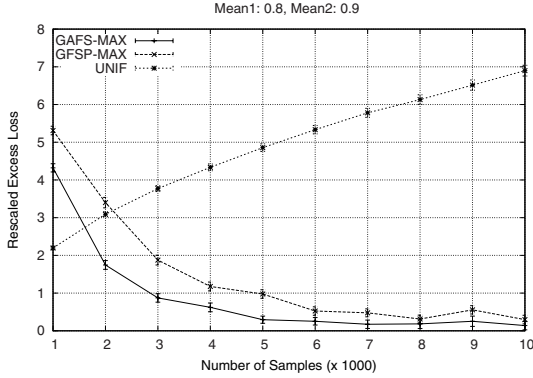


Fig. 1. The rescaled excess loss against the number of samples

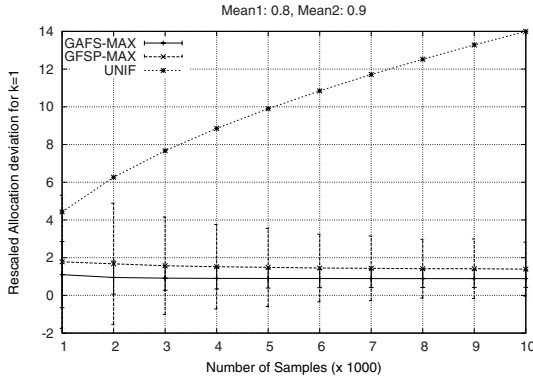


Fig. 2. The rescaled allocation deviation for $k = 1$ against the number of samples

Figure 2 shows and the rescaled allocation ratio deviations, $\sqrt{n}(T_{kn}/n - \lambda_k)$, for $k = 1$. Again, as predicted by the theory, the rescaled deviations stay bounded for the adaptive algorithms, while, due to mismatch of the allocation ratios, grows as $\sqrt{n}$ for the uniform sampling method. In this case the incremental method (GFSP-MAX) performs better than the algorithm that works in phases (GAFS-MAX), although their performance is quite similar.

5 Related Work

This work is closely related to active learning in a regression setting (e.g., [3]). Interestingly, in the by now rather extensive active learning literature to the best of our knowledge no one looked into the problem of learning in a situation where the noise in the dependent variable varies in space, i.e., under *heteroscedastic noise*. Although the rate of convergence of a method that pays attention to heteroscedasticity will not be better than that of the one that does not, the

finite-time performance can be improved greatly by such adaptive algorithms. This has been demonstrated convincingly in the related problem of actively deciding about the proportions of samples to be used in stratified sampling [5]. Interestingly, this application is very closely related to the problem studied here. The only difference is that the loss is measured by taking the weighted sum of the losses of the individual prediction errors with some fix set of weights that sum to one. With obvious changes, the algorithm presented here can be modified to work in this setting and the analysis carries through with almost no changes. The algorithm studied in [5] is the phase-based algorithm. The results in this paper are weak consistency results, i.e., no rate of convergence is derived. In fact, the only condition the authors pose on the proportion of forced selections is that this proportion should go to zero such that the total number of forced selections for any option goes to infinity.

6 Conclusions and Future Work

When finite sample performance is important, one may exploit heteroscedasticity to allocate more samples to parts of the input space where the variance is larger. In this paper we designed an algorithm for such a situation and showed that the excess loss of this algorithm compared with that of an optimal rule, that knows the variances, decays as $\tilde{O}(n^{-3/2})$. We conjecture that the optimal minimax rate is in fact $O(n^{-3/2})$. Our analysis can probably be improved. In particular, the dependence of our constants on $\lambda_{\min}^{-1}$ can probably be improved by a great extent.

Although in this paper we have not considered the full non-parametric regression problem, we plan to extend the algorithm and the analysis to such problems. We also plan to apply the technique to stratified sampling.

Acknowledgements

This research was funded in part by the National Science and Engineering Research Council (NSERC), iCore and the Alberta Ingenuity Fund and by the Hungarian Academy of Sciences (Bolyai Fellowship for András Antos).

References

- [1] Athreya, K.B., Lahiri, S.N.: Measure Theory and Probability Theory. Springer, Heidelberg (2006)
- [2] Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite time analysis of the multiarmed bandit problem. *Machine Learning* 47(2-3), 235–256 (2002)
- [3] Castro, R., Willett, R., Nowak, R.D.: Faster rates in regression via active learning. In: *Advances in Neural Information Processing Systems* 18 (NIPS-2005) (2005)
- [4] Devroye, L., Györfi, L., Lugosi, G.: A Probabilistic Theory of Pattern Recognition. In: *Applications of Mathematics: Stochastic Modelling and Applied Probability*. Springer, New York (1996)

- [5] Eto, P., Jourdain, B.: Adaptive optimal allocation in stratified sampling methods (2007), <http://www.citebase.org/abstract?id=oai:arXiv.org:0711.4514>
- [6] Hoeffding, W.: Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association* 58, 13–30 (1963)
- [7] Lai, T.L., Robbins, H.: Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics* 6, 4–22 (1985)

Query Learning and Certificates in Lattices

M. Arias and J.L. Balcázar

LARCA Research Group, Departament LSI
Universitat Politècnica de Catalunya
{marias,balqui}@lsi.upc.edu

Abstract. We provide an abstract version, in terms of lattices, of the Horn query learning algorithm of Angluin, Frazier, and Pitt. To validate it, we develop a proof that is independent of the propositional Horn logic structure. We also construct a certificate set for the class of lattices that generalizes and improves an earlier certificate construction and that relates very clearly with the new proof.

1 Introduction

In the area of Learning via Queries, the most successful protocol so far is via Membership and Equivalence queries. Three major algorithms for this model are the L^* algorithm for learning regular sets in terms of deterministic finite automata [1]; the algorithm that learns monotone DNF (closely related to its corresponding PAC version by Valiant [20]); and a sophisticated evolution of the latter which allows for learning Horn formulas [3] (the two variants of the algorithm were named HORN and HORN1 in [3] but most researchers chose to call it “AFP algorithm” after the initials of the authors’ names).

Each of these has had a number of interesting applications. In particular, the AFP algorithm turned out to be crucial for developments in other areas such as Knowledge Compilation [19], Intuitionistic Logic Programming [14] or even Databases [18], and similar algorithms exist for learning from entailment [11] and for learning certain description logics [12]. Related research questions, with answers to a couple of them, are proposed in [6] and [17]; one of these contributions is an alternative way of proving the correctness of the AFP algorithm.

Several studies proceeded to analyze combinatorial dimensions of learnability problems (see the survey [4]); initially, they were ways of stating lower bounds aiming at nonlearnability results [2]; it was later found that these dimensions characterized polynomial query learnability, for specific protocols first ([9], [10], [15], [16]) and in more generality later on ([4], [7], [8]). Specifically, for Membership and Equivalence protocols, the certificate size was shown to characterize query complexity [16]. Certificates are, essentially, sets of labeled examples whose correct classification requires a certain size of the representation mechanism under study (precise definitions are given below).

The algorithm for learning deterministic finite automata proceeds by gathering strings that lead to each of the states, plus additional strings showing how each pair of states must be distinguished: in essence, they show how many states

are required, and can be seen as certificates. The same consideration applies to monotone DNFs, where the minterms, plus the negative examples just below them (or, in other formulations, their pairwise intersections), are both the crucial queries and the certificates lower-bounding the number of necessary terms. Nevertheless, the study of certificates for Horn formulas [5] was somewhat less tightly connected with the learning algorithm, and in fact an interesting research problem is to close the existing gap between the lower bound on the number of queries given by the certificates and the upper bound attained by the currently known algorithms.

Our contributions here are as follows: first, we abstract both the AFP algorithm and its proof into the more abstract problem of learning sublattices; translating the algorithm is an easy task but our correctness proof has to be very different from the original one (and is also much simpler than the proof given in [6]). Second, we show that our proof makes explicit how the AFP algorithm is indeed constructing certificates, and we take advantage of this explanation to prove bounds on certificate size for learning sublattices. Third, we show that the translation of our bounds back into the Horn clause setting gives a bound applicable to the more demanding setting of strong proper learning [16], thus strengthening the result of [5].

2 Preliminaries

We are learning finite lower-sub-semi-lattices. This means that we have a finite lower-semi-lattice $\mathcal{L}$: a partial order relation on a finite carrier set, where the meet (or: greatest lower bound) of two elements is always well-defined. The target to be identified is a subset of $\mathcal{L}$ that is closed under meet: therefore, the subset is itself a lower semi-lattice. However, the connection between lattices and lower semi-lattices is very strong: every lattice is a lower semi-lattice, of course, but the converse direction only has to discuss the presence of a top element. Therefore, for the rest of the paper, we assume, as a minor simplification, that we are working in a lattice, learning a sublattice that includes the top element.

The particular case that guides us is as follows: the carrier lattice is the hypercube $\{0, 1\}^n$ with bitwise-comparison as ordering, and bitwise-and as meet. The target, as a set of bit-vectors, is thus a propositional theory; being closed under bitwise-and is equivalent to requiring that it can be axiomatized by a conjunction of Horn clauses. The learning algorithm for this case is given in [3].

For this particular case, the equivalence queries hypothesized by the algorithm are thus Horn formulas, that we assume written in the form of implications: all clauses with the same left hand side α are treated together in a single implication $\alpha \rightarrow \beta$ ¹. Here α and β are terms, and β stands in fact for $\alpha \cup \beta$; note that terms are in bijective correspondence with bit-vectors, and an implication thus

¹ Notice that this differs from an alternative existing interpretation [21] in which $\alpha \rightarrow \beta$ represents the clause $(\bar{x}_1 \vee \dots \vee \bar{x}_k \vee y_1 \vee \dots \vee y_{k'})$, where $\alpha = \{x_1, \dots, x_k\}$ and $\beta = \{y_1, \dots, y_{k'}\}$. Though identical in syntax, the semantics are different; in particular, ours can only represent a conjunction of definite Horn clauses whereas the other represents a general possibly non-Horn clause.

correspond to a pair of bit-vectors, one above the other. The implication $\alpha \rightarrow \beta$ can be seen as a conjunction of clauses with the same antecedent: we propose to call these conjunctions para-clauses. A bit-vector x satisfies the implication, denoted $x \models \alpha \rightarrow \beta$, if it either fails the antecedent or satisfies the consequent, that is, $x \not\models \alpha$ or $x \models \beta$ respectively. The target can be seen as a conjunction of implications or, alternatively, as the set of all bit-vectors that satisfy these implications: a propositional theory. We call *cost* of the theory the minimum number of implications needed to axiomatize it. The original Horn learning algorithm is polynomial in the cost and in the number of propositional variables, which bounds how many times one can step down along the boolean hypercube.

In the more general case, we learn sublattices. We describe a sublattice via a *sublattice basis* (abbreviated from now on as *subbasis*): a set of pairs of elements of the lattice, abstracting the notion of implications. Let x, x' be elements of the lattice $\mathcal{L}$, with $x \leq x'$: we say that x and x' are a *comparable pair*, and write always second the larger element. The elements y that *respect* the comparable pair, denoted $y \models (x, x')$, are those for which the implication “if $x \leq y$ then $x' \leq y$ ” holds, that is, either y is not above x , or is above x' . A subbasis, that is, a set of pairs, then defines the subset of $\mathcal{L}$ consisting of the elements that respect every pair. It turns out that:

Proposition 1. *Given a set of comparable pairs (a subbasis), the set of all elements that respect all of them is closed under meet; and every sublattice has a subbasis, that is, a set of comparable pairs such that the sublattice is exactly the set of elements that respect all of them.*

Proof. To see that if two elements y, y' respect a comparable pair (x, x') , then their meet still respects it, note that if either of y or y' is not above x the meet is not above x either, and that if both are above x , both must be above x' to respect the pair, and so must be their meet, as greatest lower bound. To construct a subbasis for an arbitrary sublattice, take every element that is not in the sublattice and use it to construct a comparable pair, by pairing it to the meet of all the elements above it in the sublattice. $\square$

The *height* of the lattice $\mathcal{L}$ is the length of the longest descending chain. The *cost* of a sublattice of $\mathcal{L}$ is the minimum cardinality of a subbasis.

3 The Algorithm

The lattice $\mathcal{L}$ is assumed known, and a target sublattice $\mathcal{L}^*$ is to be identified via subbases. We assume throughout that the top of $\mathcal{L}$ belongs to the target, and we denote it $\top$. The learning algorithm is almost exactly the main one in [3], except that it is formulated in lattice-theoretic terms. The correctness proof in the next section is, however, purely lattice-theoretic and is very different from the one in [3] (and is also different from, and much simpler than, the ones in [6], although along similar lines).

For $x \in \mathcal{L}$, we denote $x^* = \bigwedge \{y \in \mathcal{L}^* \mid x \leq y\}$; since the target is a sublattice, thus closed under meet, always $x^* \in \mathcal{L}^*$, and $x \leq x^*$, with $x = x^*$ if and only

if $x \in \mathcal{L}^*$. It is easy to see that the $\star$ operator is extensive (that is, $x \leq x^*$), monotonic (if $x \leq y$ then $x^* \leq y^*$) and idempotent ($x^{**} = x^*$); that is, a closure operator.

In the case of propositional theories and Horn formulas, x^* has true all the variables that can be inferred to be true from the variables true in x , using the target Horn clauses as rules.

Examples for the learning algorithm are elements of $\mathcal{L}$; they are positive or negative according to their membership into the target. Everywhere in this section, the inequality is the comparison according to $\mathcal{L}$'s partial order, and the meet operation is $\mathcal{L}$'s too.

The algorithm maintains a set P of all the positive examples seen so far. The algorithm maintains also a sequence $N = (x_1, \dots, x_t)$ of negative examples. The argument of an equivalence query is prepared from the list $N = (x_1, \dots, x_t)$ of negative examples combined with the set P of positive examples. The query corresponds to the following intuitive bias: everything is assumed positive unless some x_i suggests otherwise, and everything that some x_i suggests negative is assumed negative unless some positive example suggests otherwise. This is exactly the intuition in the hypothesis constructed by the AFP algorithm.

More precisely, the equivalence query is represented as a subbasis. For the set of positive examples P , denote $P_x = \{y \in P \mid x \leq y\}$. Observe, for later use, that since $P \subseteq \mathcal{L}^*$, we have:

$$\bigwedge P_x = \bigwedge \{y \in P \mid x \leq y\} \geq \bigwedge \{y \in \mathcal{L}^* \mid x \leq y\} = x^*$$

The hypothesis to be queried, given the set P and the list $N = (x_1, \dots, x_t)$, will be denoted $H(N, P)$. It is specified by the subbasis constructed as follows: for each x_i in N , add to the subbasis the pair $(x_i, \bigwedge P_{x_i})$. For the initial status of an empty N , this is an empty basis, which is respected by the whole lattice $\mathcal{L}$; that will be the first equivalence query issued.

A positive counterexample is treated just by adding it to P . A negative counterexample y is used to either refine some x_i into a smaller negative example, or to add x_{t+1} to the list. Specifically, let

$$i := \min(\{j \mid (x_j \wedge y) \notin \mathcal{L}^* \text{ and } x_j \wedge y < x_j\} \cup \{t+1\})$$

and then refine x_i into $x'_i = x_i \wedge y$, in case $i \leq t$, or else make $x_{t+1} = y$, subsequently increasing t . The value of i is found through membership queries on all the $x_j \wedge y$.

Putting all together, the algorithm, which is called here AFP-L since it is the Lattice version of the algorithm by Angluin, Frazier, and Pitt [3], is:

```

Algorithm AFP-L:
   $N =$  empty list
   $P = \{\top\}$ 
   $t = 0$ 
  while EQ( $H(N, P)$ ) = ("no",  $y$ ):
    if  $y \not\models H(N, P)$ :
      add  $y$  to  $P$ 

```

```

else: /*  $N = (x_1, \dots, x_t)$  */
      use MQ to find the first  $i$  such that:
           $x_i \wedge y$  is negative
           $x_i \wedge y < x_i$ , that is,  $x_i \not\leq y$ 
      if found, refine: replace  $x_i$  by  $x_i \wedge y$  in  $N$ 
      if not found: /* append  $y$  to the end of  $N$  */
           $t = t + 1$ 
           $x_t = y$ 

```

4 The Correctness Proof

Like in usual query learning algorithms, in the presence of equivalence queries, the algorithm only stops upon receiving a positive answer to an equivalence query, so that termination implies correctness. We will prove termination of the algorithm by showing that the number t of negative examples x_i in use never exceeds the cost of the target. First, we discuss separately the following easy property:

Lemma 1. *Let P be a set of positive examples, and let x be a negative example. Consider the comparable pair $(x, \bigwedge P_x)$, and assume that $y \models (x, \bigwedge P_x)$ and that $x \leq y$. Then $x^* \leq y$.*

Proof. Simply notice that, from the definition of $y \models (x, \bigwedge P_x)$, we get that $y \geq x$ implies $y \geq \bigwedge P_x \geq x^*$. $\square$

4.1 Invariant

We prove that the list N of negative examples maintains always a specific property of existence of certain positive examples. Then we will explain that N , jointly with these positive examples, are precisely certificates for the target lattice, with respect to its cost [16]: this will imply that the size t of the list of negative examples will never exceed the cost (minimal size of a subbasis) of the target.

The main result for the proof is therefore as follows:

Lemma 2. *Along the running of the algorithm in the previous section, at the point of issuing the equivalence query, for every x_i and x_j in N with $i < j$ there exists an element $z \in \mathcal{L}^*$ such that $x_i \wedge x_j \leq z \leq x_j$.*

Here is where we depart from the proof of the AFP algorithm given in [6], which stated a property similar to this lemma but much weaker, that had to be complemented through complex case analysis and additional facts.

Proof. We need to establish the fact at the time of creating a new element of N , that is, for each $i \leq t$ with respect to x_{t+1} , and we need to argue that the refinements that take place when no such new element is created maintain the

fact stated. Both are argued similarly. If a positive counterexample is received, no element of N changes; thus we only need to consider a negative counterexample y .

First note the easiest case whereby x_i gets refined into $x'_i = x_i \wedge y$. This leaves x_j untouched, and brings down $x_i \wedge x_j$ into $x'_i \wedge x_j$; the same value of z will do: $x'_i \wedge x_j \leq x_i \wedge x_j \leq z \leq x_j$.

Now consider the case in which x_j is refined into $x'_j = x_j \wedge y$. We provide a positive example z' for which $x_i \wedge y \leq z' \leq y$. Then, considering its meet with the already existing z gives

$$x_i \wedge x'_j = x_i \wedge x_j \wedge y \leq z \wedge z' \leq x_j \wedge y = x'_j$$

Moreover, both z and z' being positive, and the target being closed under meet, ensures that $z \wedge z'$ is positive.

To find z' , observe that x_i came before but was not chosen for refinement; either $x_i \wedge y$ is itself positive, and can be chosen for z' , or $x_i \leq y$, in which case we use lemma 1: $(x_i, \bigwedge P_{x_i})$ is part of the query, and y was a negative counterexample so it must satisfy the query, including that pair: $y \models (x_i, \bigwedge P_{x_i})$. Lemma 1 tells us $x_i^* \leq y$, whence $x_i \wedge y \leq x_i \leq x_i^* \leq y$; and x_i^* is of course positive.

The case of creation of $x_{t+1} = y$ is handled in the same way: the z' obtained fulfills the condition $x_i \wedge y \leq z' \leq y$ which is what we need. $\square$

4.2 Termination

It remains to show how the lemma just proved guarantees that t is bounded by the dimension of the target. Essentially, the proof says that the elements of N , together with their corresponding values of z considered pairwise as per the lemma, establish a lower bound on the cost of the target sublattice, exactly in the same way as the combinatorial notion of certificates used for nonlearnability proofs [16].

Lemma 3. *Let B be a subbasis of the target. Consider two different negative examples x_i and x_j . Each of them has a comparable pair in B that they do not respect: but it cannot be the same pair.*

Therefore, if the target has cost m , then it has a subbasis with m pairs, and the number of examples x_i will not exceed m .

Proof. Consider a pair $(x, y) \in B$ such that $x_i \not\models (x, y)$ and, likewise, $x_j \not\models (x, y)$, for $i \neq j$. That is, $x \leq x_i$, and $x \leq x_j$, which implies that $x \leq x_i \wedge x_j \leq z$ for the value z given by lemma 2; however, z is a positive example, and must respect the pair (x, y) . Then, from $x \leq z$ immediately follows $y \leq z \leq x_j$, as given by the same lemma, and therefore $x_j \models (x, y)$ actually. $\square$

4.3 Final Analysis

It is straightforward to implement in polynomial time the algorithm; the analysis of the number of queries is also easy now: let h be the height of $\mathcal{L}$, and let m

be the cost of the target. Each negative counterexample either increases N , which can happen at most m times, or refines some x_i , which can happen at most h times each, for a total of at most $h \times m$ negative counterexamples. Likewise, each positive counterexample added to P must change at least one of the comparable pairs in the query, by reducing $\bigwedge P_x$, which again can happen at most h times each: we just need to add a constant factor to $h \times m$. Finally, between counterexamples, at most m membership queries are asked.

Therefore, we can state:

Theorem 1. *Let $\mathcal{L}$ be a lattice of height h , and let $\mathcal{L}^*$ be any sublattice of $\mathcal{L}$ including the top of $\mathcal{L}$, and of cost m . Then the algorithm AFP-L learns $\mathcal{L}^*$ in polynomial time, with at most $O(h \times m^2)$ queries.*

5 Certificates

The certificate size [16, 5] $q(m, n)$ of a given class of sublattices $\mathcal{L}_n^{\leq m}$ of cost at most m and height n with respect to hypothesis expansions² of cost at most $p(m, n)$ is the minimum cardinality of a set of elements such that for any given sublattice f which is not in the class $\mathcal{L}_n^{\leq p(m, n)}$, there is a set Q_f of cardinality at most $q(m, n)$ such that no sublattice in the class $\mathcal{L}_n^{\leq m}$ is consistent with f over Q_f . In other words, f differs with every sublattice in $\mathcal{L}_n^{\leq m}$ on at least one element in Q_f . In a way, Q_f is “rejecting” all the sublattices in $\mathcal{L}_n^{\leq m}$ and “certifying” that f is not in $\mathcal{L}_n^{\leq m}$.

Our construction is very similar to the one for Horn CNFs in [5], but improves on it by constructing certificates for $p(m, n) = m$ thus applying to the more demanding strong proper learning model and generalizing it to the more abstract setting of lattices used in this paper.

5.1 Useful Facts

Assume throughout this section that $(\alpha_1, \beta_1), \dots, (\alpha_k, \beta_k)$ is a subbasis of minimum cost of a lattice $\mathcal{L}^* \subseteq \mathcal{L}$. Now, consider the subbasis $(\alpha_1, \beta_1^*), \dots, (\alpha_k, \beta_k^*)$. Clearly, this is a subbasis for $\mathcal{L}^*$ as well: $\mathcal{L}^* \models (\alpha_i, \beta_i^*)$ for each $i = 1, \dots, k$ and any example x respecting all of $\{(\alpha_i, \beta_i)\}_i$ must respect all $\{(\alpha_i, \beta_i^*)\}_i$ as well.

Remark 1. Without loss of generality we assume that the subbasis of minimum cost used throughout the proof $\{(\alpha_i, \beta_i)\}_i$ is such that $\beta_i^* = \beta_i$, that is, $\beta_i \in \mathcal{L}^*$. By the observations above, such subbasis always exists, represents $\mathcal{L}^*$ exactly, and is of minimum cost as well. It is also immediate to verify that, in this case, $\beta_i^* = \beta_i = \alpha_i^*$.

² The fact that hypotheses can have an expansion of cost at most $p(m, n)$ means that algorithms are allowed to present hypotheses of cost at most $p(m, n)$. It is assumed that $m \leq p(m, n)$.

For each $i = 1, \dots, k$, let $\mathcal{L}_i^*$ be the sublattice described by the same comparable pairs with the exception of (α_i, β_i) . Note that $\mathcal{L}^* \subseteq \mathcal{L}_i^*$ and, further, that the inclusion has to be proper since, otherwise, the given subbasis would not be of minimum cost. For each (α_i, β_i) , define $x_i \in \mathcal{L}$ as follows:

$$x_i = \bigwedge \{y \in \mathcal{L}_i^* \mid \alpha_i \leq y\} \quad (1)$$

The following series of lemmas is going to be useful for our main Theorem 2; in several cases the essentials of the proofs are already in the literature.

Lemma 4. *For all i , $x_i \not\models (\alpha_i, \beta_i)$; but $x_i \models (\alpha_j, \beta_j)$ for every $j \neq i$.*

Proof. The proof generalizes from that of Lemma 10 in [5]. Clearly $x_i \geq \alpha_i$. If $x_i \models (\alpha_i, \beta_i)$, then all $y \in \mathcal{L}_i^*$ with $y \geq \alpha_i$ fulfill $y \geq x_i \geq \beta_i$ so that $\mathcal{L}_i^*$ satisfies (α_i, β_i) , contradicting the strict inequality $\mathcal{L}_i^* \subset \mathcal{L}^*$. On the other hand, x_i is the meet of a number of elements in $\mathcal{L}_i^*$ which is a sublattice, thus closed under meet, and $x_i \in \mathcal{L}_i^*$ so that $x_i \models (\alpha_j, \beta_j)$ for every $j \neq i$. $\square$

Lemma 5. *For all $i = 1, \dots, k$ it holds that $x_i^* = \bigwedge \{y \in \mathcal{L}^* \mid \alpha_i \leq y\} = \alpha_i^*$ and $x_i^* \in \mathcal{L}^*$.*

Proof. We prove $\alpha_i \leq x_i \leq \alpha^*$; then, by monotonicity and idempotency, $\alpha^* \leq x_i^* \leq \alpha^{**} = \alpha^*$. The first inequality is from the definition of x_i , whereas the second one holds because α^* is itself among the y 's whose meet is taken to define x_i . Also, $\mathcal{L}^*$ being closed under meet implies that x_i^* is positive. $\square$

Lemma 6. *If $x_i \leq x_j$, then $x_i^* \leq x_j$.*

Proof. Variants of this lemma have been stated in previous work in various guises, e.g. [22]. Suppose that $x_i \leq x_j$. Since $x_j \in \mathcal{L}_j^*$, it must respect the pair (α_i, β_i) . Then, since $\alpha_i \leq x_i \leq x_j$ then $\beta_i \leq x_j$. It only remains to notice that $\beta_i = \beta_i^* = \alpha_i^* = x_i^*$ by Remark 1 and Lemma 5 so that $x_i^* \leq x_j$. $\square$

Lemma 7. *$x_i \wedge x_j \notin \mathcal{L}^*$ iff $x_i \leq x_j$ or $x_j \leq x_i$.*

Proof. The direction from right to left is clear: if $x_i \leq x_j$ (or $x_j \leq x_i$), then $x_i \wedge x_j = x_i$ (or $x_i \wedge x_j = x_j$). In either case, we know from Lemma 4 that $x_i \notin \mathcal{L}^*$ and $x_j \notin \mathcal{L}^*$.

For the other direction, assume that $x_i \wedge x_j \notin \mathcal{L}^*$. The comparable pairs (α, β) of $\mathcal{L}^*$ that are neither (α_i, β_i) nor (α_j, β_j) are satisfied by $x_i \wedge x_j$, since $\mathcal{L}^*$ is closed under meet and Lemma 4 guarantees that both x_i and x_j respect (α, β) . Therefore, it must be that either $x_i \wedge x_j \not\models (\alpha_i, \beta_i)$ or $x_i \wedge x_j \not\models (\alpha_j, \beta_j)$. Assume w.l.o.g. that $x_i \wedge x_j \not\models (\alpha_i, \beta_i)$. This implies that $\alpha_i \leq x_i \wedge x_j$ so that $\alpha_i \leq x_j$. Since x_j respects (α_i, β_i) , it must hold that $\beta_i \leq x_j$. Moreover, it always holds that $x_i \leq x_i^*$. Putting these two together and using the fact that $x_i^* = \beta_i$ (see Remark 1 and Lemma 5), we get $x_i \leq x_j$ as required. $\square$

5.2 The Certificate Set

Theorem 2. *Sublattices have polynomial size certificates with*

$$q(m, n) = \binom{m+1}{2} + m + 1 .$$

Proof. Fix $n, m > 0$. Let $\mathcal{L}^*$ be a lattice of height n that is not representable by a subbasis of cost m . First, we construct a certificate set Q of cardinality at most $\binom{m+1}{2} + 2m + 2$ for every such $\mathcal{L}^*$; we will see later that we can remove some elements from Q and apply a finer count to obtain the bound in the statement.

We are given a minimal subbasis $(\alpha_1, \beta_1), \dots, (\alpha_k, \beta_k)$ where $k \geq m+1$. Let x_i be defined as in Equation (1). Now, define the certificate set $Q = Q^- \cup Q^+ \cup Q^\wedge$, where $Q^- = \{x_i\}_i$, $Q^+ = \{x_i^*\}_i$, and $Q^\wedge = \{x_i \wedge x_j\}_{i < j}$, where $1 \leq i, j \leq k$. We prove that Q is indeed a certificate set.

Take any sublattice $\mathcal{L}'$ of cost at most m . By way of contradiction, assume that $\mathcal{L}'$ is consistent with $\mathcal{L}^*$ over all the elements in Q . Lemmas 4 and 5 guarantee that $x_i \not\models \mathcal{L}'$ but $x_i^* \models \mathcal{L}'$ for all $1 \leq i \leq m+1$. There must be one particular comparable pair (α, β) in $\mathcal{L}'$ violated by both x_i and x_j for some i, j s.t. $1 \leq i < j \leq m+1$. Clearly $x_i \wedge x_j \not\models (\alpha, \beta)$; therefore $x_i \wedge x_j \not\models \mathcal{L}'$. Furthermore, $\alpha \leq x_i \leq x_i^*$ and thus $\beta \leq x_i^*$, otherwise we would have that $x_i^* \not\models (\alpha, \beta)$.

If x_i and x_j are incomparable, then Lemma 7 guarantees that $x_i \wedge x_j \models \mathcal{L}^*$, thus $\mathcal{L}'$ and $\mathcal{L}^*$ differ on $x_i \wedge x_j \in Q^\wedge$. If $x_i \leq x_j$, then by Lemma 6 and our observation above we obtain that $\beta \leq x_i^* \leq x_j$ and thus x_j respects (α, β) , which contradicts $x_j \not\models (\alpha, \beta)$. The case where $x_j \leq x_i$ is analogous.

We conclude that Q is a certificate set of cardinality at most $q(m, n) = \binom{m+1}{2} + 2m + 2$. Notice however that the proof above only uses x_i^* when there is a x_j s.t. $x_i \leq x_j$. Moreover, in this case, it is clear that $x_i \wedge x_j = x_i$ so we do not need to count it in $Q^\wedge$. Therefore it is sufficient to include in Q^+ only those x_i^* such that $x_i \leq x_j$ for some $i < j \leq k$. The resulting set Q is also a certificate and has the desired cardinality, which concludes our proof. $\square$

To apply this theorem in the Horn formulas, we need to discuss also the case of a set of propositional models that does not allow for Horn axiomatization: but, in such a case, it is not closed under meet; then there must exist 3 examples $x, y, x \wedge y$ such that $x, y \in \mathcal{L}^*$ but $x \wedge y \notin \mathcal{L}^*$. Therefore the certificate set Q can be constructed as $Q = \{x, y, x \wedge y\}$ and $|Q| \leq 3$. Clearly, no lattice, closed under meet, can be consistent over these three examples.

The certificates of [5] for Horn CNF use $p(m, n) = m(n+1)$. In contrast, here we only require that $p(m, n) = m$ thus applying to the more stringent model of strong proper learning [16]. On the other hand, in [5] m counts the number of clauses whereas here m is the number of comparable pairs, which translates to using *para-clauses* or implications in the Horn logic setting. Notice that the size of a Horn expression in terms of para-clauses is at most its size in clauses. The authors feel that the right thing to count in the case of Horn expressions is the number of para-clauses, especially in the context of analyzing the AFP algorithm that learns using the para-clause representation.

Notice also that this upper bound matches *exactly* the lower bound from [5] in the case that $m < n$. If $m > n$, there is still a gap between the lower bound of $\Omega(mn)$ in [5] and our upper bound of $O(m^2)$.

6 Certificates and the Learning Algorithm

We are going to illustrate the link between our certificate construction and the learning algorithm with an example where the lattice to be learned is a Horn formula $\mathcal{H}^*$. The example is taken from [13], where a canonical representation for definite Horn theories is introduced. The propositional variables are going to be a, b, c, d, e, f (and so the height of the lattice is 6). Suppose that our target concept is

$$\mathcal{H}^* = \{e \rightarrow d, bc \rightarrow d, bd \rightarrow c, cd \rightarrow b, ad \rightarrow bce, ce \rightarrow ab\}.$$

First let us compute the examples $x_1, \dots, x_6$ from Lemma 4, and their closure $x_1^*, \dots, x_6^*$.

i	$\alpha_i \rightarrow \beta_i$	$\alpha_i \rightarrow \beta_i^*$	x_i	x_i^*
1	$e \rightarrow d$	$e \rightarrow ed$	00001	00011
2	$bc \rightarrow d$	$bc \rightarrow bcd$	01100	01110
3	$bd \rightarrow c$	$bd \rightarrow bcd$	01010	01110
4	$cd \rightarrow b$	$cd \rightarrow bcd$	00110	01110
5	$ad \rightarrow bce$	$ad \rightarrow abcde$	10010	11111
6	$ce \rightarrow ab$	$ce \rightarrow abcde$	01111	11111

We can compute x_i and x_i^* by setting to one the bits corresponding to the antecedent of para-clause 1 and then doing forward chaining exhaustively with all the other clauses to obtain x_i , and with all the clauses to obtain x_i^* . It is worth noting that for $x_1, \dots, x_5$ the forward chaining is not needed since no antecedents of the other clauses are satisfied. However, when computing x_6 we start with the assignment 00101 but need to set to 1 the bits bd since the clauses 1 and 4 are triggered in the deduction process.

Notice also that the comparable pairs corresponding to a minimal subbasis of this concept are (x_i, x_i^*) for $i = 1, \dots, 6$.

In our example simulation, we are going to feed

$$x_1, x_1^*, x_2, x_2^*, x_3, x_4, x_5, x_6$$

as counterexamples. From the simulation it will become clear why we do not need to present some of the positive x_i^* .

Let $t = 11111$ be the top assignment. We assume that we are dealing with definite clauses, so t is always positive. The following table captures the internal state of the algorithm and the dynamics of the simulation:

N	P	$EQ(H(N, P))$	MQs issued
1 $()$	$\{t\}$	$x_1 = 00001$	
2 (x_1)	$\{t\}$	$x_1^* = 00011$	
3 (x_1)	$\{t, x_1^*\}$	$x_2 = 01100$	$x_1 \wedge x_2 = 00000, +$
4 (x_1, x_2)	$\{t, x_1^*\}$	$x_2^* = 01110$	
5 (x_1, x_2)	$\{t, x_1^*, x_2^*\}$	$x_3 = 01010$	$x_1 \wedge x_3 = 00000, +$ $x_2 \wedge x_3 = 01000, +$
6 (x_1, x_2, x_3)	$\{t, x_1^*, x_2^*\}$	$x_4 = 00110$	$x_1 \wedge x_4 = 00000, +$ $x_2 \wedge x_4 = 00100, +$ $x_3 \wedge x_4 = 00010, +$
7 (x_1, x_2, x_3, x_4)	$\{t, x_1^*, x_2^*\}$	$x_5 = 10010$	$x_1 \wedge x_5 = 00000, +$ $x_2 \wedge x_5 = 00000, +$ $x_3 \wedge x_5 = 00010, +$ $x_4 \wedge x_5 = 00010, +$
8 $(x_1, x_2, x_3, x_4, x_5)$	$\{t, x_1^*, x_2^*\}$	$x_6 = 01111$	$x_5 \wedge x_6 = 00010, +$
9 $(x_1, x_2, x_3, x_4, x_5, x_6)$	$\{t, x_1^*, x_2^*\}$	<i>Yes</i>	

The table is to be read as follows: first $P = \{11111\}$ and N is empty. The first counterexample received by the algorithm is $x_1 = 00001$, it is negative and so it is added to N . Clearly x_1 was a negative counterexample since initially the hypothesis corresponds to the concept that classifies every example as positive. The next counterexample is $x_1^* = 00011$, which is indeed a positive counterexample since $H(\{00001\}, \{11111\})$ classifies this example as positive. After adding this positive counterexample to P , the algorithm has discovered exactly the first paracause and will include in its hypothesis the correct comparable pair (x_1, x_1^*) . The next counterexample is x_2 which is indeed a counterexample since it satisfies the comparable pair (Lemma 4) but falsifies the target concept. Since $x_1 \not\leq x_2$, the algorithm needs to check that their intersection $x_1 \wedge x_2 = 00000$ is not negative, which indeed is not. So x_2 is appended to N .

It is worth noticing that all membership queries are responded with an affirmative answer because we only ask when $x_i \wedge x_j \not\leq x_i$, for $i < j$ (Lemma 7), so negative counterexamples are always appended to N . For example, when receiving x_6 as counterexample, we do not need to check the intersections $x_i \wedge x_6$ for $i = 1, \dots, 4$ because we have that $x_i \leq x_6$; only $x_5 \wedge x_6$ is checked. Also, we do not need to present positive examples $x_3^*, x_4^*, x_5^*, x_6^*$ because they are already included in P .

In general, one can always do such a simulation with any hidden target sublattice $\mathcal{L}^*$. Assume that $\mathcal{L}^*$ is represented by a minimal subbasis of cost k and $x_1, \dots, x_k$ are the examples satisfying Lemma 4. Without loss of generality, assume that the examples follow a particular order: x_i appears before x_j if $x_i \leq x_j$. Notice that this set of preferences induce a DAG among the x_i (no cycles) and therefore such an ordering is always possible. Notice that $\{(x_i, x_i^*) \mid 1 \leq i \leq l\}$ is also a minimal basis for $\mathcal{L}^*$.

The counterexamples are going to be the a subsequence of:

$$x_1, x_1^*, x_2, x_2^*, \dots, x_k, x_k^*.$$

The positives x_i^* are presented only if they are not already in P , namely, if x_i^* is not the *top* element and $x_{i'}^* \neq x_i^*$ for all $i' < i$.

So, after presenting counterexamples x_l and x_l^* (if not in P already), we are guaranteed that $N = (x_1, \dots, x_l)$ and therefore all comparable pairs (x_i, x_i^*) up to l are exactly discovered. When the last pair of negative and positive counterexamples is issued (if the positive one is needed), the equivalence query will be done with the exact same hypothesis as the target concept, so that the simulation will terminate in success.

Clearly, the elements in N and P are exactly those of Q^- and Q^+ of our certificate construction, and the membership queries represent the positive ones in $Q^\wedge$.

Notice also that the order in which we try to refine the elements of N is irrelevant, since with this particular sequence of examples we never refine any existing counterexample in N . In general, this is obviously not the case, and keeping the elements in N in order is important. More particularly, the order is important for pairs of elements $n_{i'}, n_{j'}$ in N that are violating pairs of $\mathcal{L}^*$ with corresponding x_i and x_j s.t. $x_i \leq n_{j'}$.

Certificates and the Invariant of Section 4.1. Lemma 2 states the following: for every x_i and x_j in N with $i < j$ there exists an element $z \in \mathcal{L}^*$ such that $x_i \wedge x_j \leq z \leq x_j$. Let us analyze what the z witnessing the Lemma are. For those pairs that satisfy $x_i \leq x_j$, the witnessing z is x_i^* . Lemma 6 guarantees that $x_i \wedge x_j = x_i \leq x_i^* \leq x_j$. If x_i and x_j are incomparable, then by Lemma 7 $x_i \wedge x_j$ is going to be positive and thus z is $x_i \wedge x_j$ itself, and trivially we get: $x_i \wedge x_j \leq x_i \wedge x_j \leq x_j$. Notice that, again, the x_i , x_i^* , and $x_i \wedge x_j$ used in Lemma 2 constitute precisely our certificates.

Acknowledgements. For many related discussions, in person or remote, the authors are particularly indebted to Jaume Baixeries, Montserrat Hermo, and Josefina Sierra.

References

- [1] Angluin, D.: Learning Regular Sets from Queries and Counterexamples. *Information and Computation* 75, 87–106 (1987)
- [2] Angluin, D.: Negative Results for Equivalence Queries. *Machine Learning* 5, 121–150 (1990)
- [3] Angluin, D., Frazier, M., Pitt, L.: Learning Conjunctions of Horn Clauses. *Machine Learning* 9, 147–164 (1992)
- [4] Angluin, D.: Queries revisited. *Theor. Comput. Sci.* 313, 175–194 (2004)
- [5] Arias, M., Feigelson, A., Khardon, R., Servodio, R.: Polynomial Certificates for Propositional Classes. *Information and Computation* 204, 816–834 (2006)
- [6] Balcázar, J.L.: Query learning of Horn formulas revisited. In: *Computability in Europe Conference*, Amsterdam (2005)
- [7] Balcázar, J.L., Castro, J., Guijarro, D.: A New Abstract Combinatorial Dimension for Exact Learning via Queries. *J. Comput. Syst. Sci.* 64, 2–21 (2002)

- [8] Balcázar, J.L., Castro, J., Guijarro, D., Köbler, J., Lindner, W.: A general dimension for query learning. *J. Comput. Syst. Sci.* 73, 924–940 (2007)
- [9] Balcázar, J.L., Castro, J., Guijarro, D., Simon, H.-U.: The consistency dimension and distribution-dependent learning from queries. *Theor. Comput. Sci.* 288, 197–215 (2002)
- [10] Bshouty, N.H., Cleve, R., Gavaldà, R., Kannan, S., Tamon, C.: Oracles and Queries That Are Sufficient for Exact Learning. *J. Comput. Syst. Sci.* 52, 421–433 (1996)
- [11] Frazier, M., Pitt, L.: Learning From Entailment: An Application to Propositional Horn Sentences. In: *Int. Conference Machine Learning 1993*, pp. 120–127 (1993)
- [12] Frazier, M., Pitt, L.: CLASSIC Learning. *Machine Learning* 25, 151–193 (1996)
- [13] Guiges, J.L., Duqenne, V.: Familles minimales d'implications informatives resultants d'un tableau de données binaires. *Math. Sci. Hum.* 95, 5–18 (1986)
- [14] Gaintzarain, J., Hermo, M., Navarro, M.: On Learning Conjunctions of Horn $\supset$ clauses. In: *Computability in Europe Conference, Amsterdam* (2005)
- [15] Hegedűs, T.: On generalized teaching dimensions and the query complexity of learning. In: *Proceedings of the Conference on Computational Learning Theory*, pp. 108–117. ACM Press, New York (1995)
- [16] Hellerstein, L., Pillaipakkamnatt, K., Raghavan, V., Wilkins, D.: How Many Queries Are Needed to Learn? *Journal of the ACM* 43, 840–862 (1996)
- [17] Hermo, M., Lavín, V.: Negative Results on Learning Dependencies with Queries. In: *Proceedings on Seventh International Symposium on Artificial Intelligence and Mathematics* (2002)
- [18] Kivinen, J., Mannila, H.: Approximate Inference of Functional Dependencies from Relations. *Theoretical Computer Science* 149, 129–149 (1995)
- [19] Selman, B., Kautz, H.: Knowledge Compilation and Theory Approximation. *Journal of the ACM* 43, 193–224 (1996)
- [20] Valiant, L.: A Theory of the Learnable. *Communications of the ACM* 27, 1134–1142 (1984)
- [21] Wang, H.: Toward mechanical mathematics. *IBM Journal for Research and Development* 4, 2–22 (1960); In: Wang, H.: *A survey of mathematical logic*, Peking and Amsterdam. Science Press and North Holland, pp. 224–268 (1963)
- [22] Wild, M.: A theory of finite closure spaces based on implications. *Advances in Mathematics* 108, 118–139 (1994)

Clustering with Interactive Feedback

Maria-Florina Balcan and Avrim Blum

Carnegie Mellon University
Pittsburgh, PA 15213-3891
{ninamf, avrim}@cs.cmu.edu

Abstract. In this paper, we initiate a theoretical study of the problem of clustering data under interactive feedback. We introduce a query-based model in which users can provide feedback to a clustering algorithm in a natural way via **split** and **merge** requests. We then analyze the “clusterability” of different concept classes in this framework — the ability to cluster correctly with a bounded number of requests under only the assumption that each cluster can be described by a concept in the class — and provide efficient algorithms as well as information-theoretic upper and lower bounds.

1 Introduction

Clustering is often a highly under-specified problem: given a set of data items, there may be many different possible clusterings a user might be interested in. For instance, given a set of documents or news articles, should all those about sports go into a single cluster or should there be different clusters for football, baseball, hockey and so on? Should articles on salaries paid to sports figures be classified under sports or under business? Or perhaps, completely orthogonally, the user wants articles clustered by length or by writing style and not by topic. Most theoretical work “wishes away” this under-specification by making strong distributional assumptions, such as the data distribution being a mixture of Gaussians with each Gaussian as one of the clusters (e.g., [9, 3]). In this work, we instead embrace the idea that a given set of data might have multiple plausible clusterings, and consider the problem of *clustering under feedback*. That is, we imagine users are willing to help a clustering algorithm arrive at their own desired answer with a small amount of additional prodding, and ask what kinds of algorithms can take advantage of such feedback and under what conditions can they succeed. In this paper, we consider the problem of clustering under a quite natural type of feedback in the form of **split** and **merge** requests. Specifically, given a proposed clustering of the dataset, the user responds with either:

Split: The user identifies a cluster c in the algorithm’s clustering that contains points from multiple target clusters (and therefore should be split), or

Merge: The user identifies two clusters c, c' in the algorithm’s clustering that are both subsets of the same target cluster and therefore should be merged.

Or, if neither of these applies, the user responds that the algorithm's clustering is correct. Thus, this model is similar to the standard model of *learning with equivalence queries* [10], except that rather than return a misclassified data point, the user instead responds with the more vague request that, say, some cluster should be split (but without saying *how* that split should be done, or where in that cluster a mistake was made). We then show, perhaps surprisingly, that a number of interesting positive results can be had in this model under only the assumption that each cluster in the target is a member of some given concept class C , without any distributional assumptions on the data. In contrast, as mentioned above, most work on clustering has focused on *generative* models, such as mixtures of Gaussian or logconcave distributions, in which the underlying data distribution is effectively committing to only a single answer [1, 8, 7, 9, 14, 12].

Our model can be illustrated by the following simple example. Suppose our given dataset S consists of m points on the real line. We are told that each cluster is an interval, and let us say for simplicity of discussion we also are told there are only $k \leq 2$ clusters. In this case we could begin by proposing a single cluster with all the points and proposing it to the user. If we are incorrect, the user will respond with a **split** request, in which case we split the cluster into two intervals of $m/2$ points each and present the result to the user again. In general, if the user asks us to split a cluster, we partition it exactly in half (by cardinality), and if the user asks us to merge two clusters, we merge them. Since at most one of the algorithm's intervals can contain points from both of the user's intervals, and since each split request causes the number of points in the offending interval to drop by a factor of 2, the total number of **split** requests is at most $\lg m$. Therefore, the total number of **merge** requests is at most $\lg m$ as well, and so the overall number of requests is at most $2 \lg m$.

Note that clustering in this model with m requests is trivial for any concept class C : just begin with each point in its own cluster and merge as requested. So, our goal will be to develop algorithms whose query complexity is logarithmic in m and polynomial in parameters such as $\log |C|$ and (ideally) k . Note also that if we strengthened the model to allow the algorithm to specify *which* cluster the user should focus on, then we could simulate membership queries [2, 11];¹ indeed, one of the key difficulties in our model will be designing algorithms that can make progress no matter which clusters are asked to be split or merged.

The main results we show in this model are as follows (here, m is the total number of data points and k is the number of clusters in the target):

1. For the case of points on the line and the class of intervals, we give a simple algorithm that requires only $O(k \log m)$ requests to cluster correctly.

¹ Suppose inductively we have determined points $x_1, \dots, x_{k'}$ that are all known to be in different clusters. Then, in this strengthened model, given a new point x we could query the sequence of clusters $\{x, x_1\}, \{x, x_2\}, \dots$. The first one of these that does not produce a **split** request is the label of x (or if all produce a **split** request, we assign x to label $k' + 1$).

2. For the case of points in $\{0, 1\}^n$ and the class of conjunctions (each cluster in the target is specified by a conjunction, and each data point satisfies exactly one of these conjunctions) we give an efficient algorithm with query complexity $O(n^k)$. Thus, this is polynomial when k is constant. For the case of disjunctions, we give an efficient algorithm with query complexity $O(n)$, independent of k . We do not know if there is an efficient algorithm for conjunctions with query complexity $\text{poly}(n, k)$.
3. For a general class C , (each cluster is a member of C , so there are at most $|C|^k$ possible clusterings overall), we give a generic but computationally inefficient algorithm with query complexity $O(k^3 \log |C|)$.
4. The generic algorithm mentioned above requires the algorithm, as it is learning, to produce clusterings in which there are many more than k clusters. If instead we restrict the algorithm to producing clusterings with only $\text{poly}(k, \log m, \log |C|)$ clusters (e.g., we allow the user a third option of refusing to split or merge and instead just saying “way too many clusters”) then even for simple classes no algorithm can succeed.

1.1 Related Work and Motivation

There has, of course, been substantial theoretical work on clustering, e.g., [3, 1, 5, 8, 7, 9, 14, 12]. Much of this work assumes a generative model in which the distribution of data contains enough information at least in principle to reconstruct the single correct answer. Our model is motivated by recent work [4] that considers a relaxation of this setting in which the assumption is only that the target clustering satisfies certain natural relations with respect to the data. For instance, the target clustering might have the property that data points are closer to points in their own cluster than to points in any other cluster. This condition and others considered in [4] are not sufficient to uniquely identify the target clustering directly, but they *are* sufficient as shown in [4] to produce a *tree* (a hierarchical clustering) such that the desired clustering is some pruning of this tree. The idea is then that a user, given such a tree, could begin at the root and “click” on any node that is too broad to navigate down. This model is similar to clustering with **split** requests only, since the tree can be viewed as a pre-specification of what the algorithm would do on any given such request. Unfortunately, this approach is not sufficient to handle even the case of intervals described above, since for intervals we have multiple different possible clusterings even for $k = 2$, and none of these are refinements of the others (so no single such tree is possible). By considering an interactive model with both **split** and **merge** requests, we are able to avoid this difficulty and analyze a broader class of settings.

2 Notation and Definitions

We are given a set S of m points from some instance space X , and we assume that a user has in mind some partition of S into k disjoint clusters $c_1, \dots, c_k$.

We call $\{c_1, \dots, c_k\}$ the *target clustering*. Our goal will be to identify this target clustering from a limited number of **split** and **merge** requests (so, in learning-theoretic terms, we are considering a *transductive* problem). In particular, given a proposed clustering $\{h_1, h_2, \dots, h_{k'}\}$ of S produced by the algorithm, if this clustering is not correct then the user will respond with either **split**(h_i) for some h_i that contains points from multiple target clusters, or **merge**(h_i, h_j) for two clusters h_i, h_j that are both subsets of the same target cluster. Note that if none of these conditions hold for any of h_i then the proposed clustering must be correct (to be clear, we say that $\{h_1, \dots, h_k\}$ is correct if there is some permutation σ on $\{1, \dots, k\}$ such that $h_i = c_{\sigma(i)}$ for all i). We think of the act of proposing a clustering as making a “query” (much like the notion of an equivalence query in learning, except the response is **split** or **merge** rather than a labeled example), and our goal is to identify the target from a small number of queries. As in the standard equivalence query model, we view the user as adversarial in the sense that we want to identify the target with a small number amount of feedback no matter how the user chooses which request to make if multiple **split** or **merge** requests apply.

If C is a concept class, we say that the clustering $\{c_1, \dots, c_k\}$ is in class C if each $c_i \in C$. We say that an algorithm clusters class C with Q queries if it is able to identify the target with at most Q queries for any clustering in C . Our goal will be to do this for a number of queries that is polynomial in k , $\log |C|$, and $\log m$. As noted above, clustering with m queries is trivial: simply begin with m clusters each containing a single point in S and then merge as requested.

Note that because we assume each point $x \in S$ belongs to exactly one cluster c_i of the target, not every clustering in a given class C may be consistent with a given dataset S and vice-versa. E.g., if C is the class of disjunctions and S contains the point 10010, then the target could not, for instance, have $x_1 \vee x_2$ as one cluster and $x_4 \vee x_5$ as another. In fact, because of this issue, we will also consider what we call the *extended* model, where we allow the final cluster c_k to equal $S - (c_1 \cup \dots \cup c_{k-1})$, even if this cannot be written as a concept in C . So, for $k = 2$, the set of possible clusterings of the given dataset S in the extended model is exactly $\{(c \cap S, \bar{c} \cap S) : c \in C\}$. All of our results can be made to hold for the extended model as well.

We have not yet specified whether or not algorithms are allowed to produce clusterings $\{h_1, \dots, h_{k'}\}$ in which the clusters h_i overlap. We find that in some cases (e.g., Section 4) it is easier to design algorithms if we allow overlap, but we are also able to remove this at the expense of somewhat worse bounds. In all cases, however, we require the hypothesis clustering to cover all the points of S .

One final point: we have not yet placed any conditions on the number of clusters k' that the algorithm may use in its clusterings. Ideally, an algorithm should use $k' = O(k)$ or perhaps $k' = \text{poly}(k, \log m)$ clusters, since this quantity in some sense determines the “cognitive load” on the user. In fact, for most of our algorithms this is indeed the case. However, our generic algorithm (Section 5) may need substantially more clusters, and in Section 6 we show that in fact this is necessary in our framework to achieve a good bound in general.

3 Intervals

To illustrate the general setting, in this section we present a simple algorithm for the class C of intervals on the line, that requires at most $k \log m$ requests to cluster correctly. Specifically, the algorithm is as follows.

Algorithm Cluster-Intervals:

- Begin with a single cluster containing all m points of S .
- On a **split** request to a cluster c , partition c into two clusters of equal cardinality.
- On a **merge** request, merge the two clusters indicated.

Theorem 1. *Algorithm Cluster-Intervals requires only $O(k \log m)$ requests to cluster the class of intervals on the line.*

Proof: Since the target consists of k intervals, we can identify $k - 1$ decision boundaries $a_1, \dots, a_{k-1}$, where a_i is an arbitrary position between the largest point in S in target interval i and the smallest point in S in target interval $i + 1$. If a_i lies inside some cluster c of the algorithm's clustering, define $\text{size}(a_i)$ to be the number of points of S in c ; else define $\text{size}(a_i) = 0$. (For concreteness, if a hypothesis cluster c contains points $p_1 < p_2 < \dots < p_t \in S$, we define $c = [p_1, p_t]$.) So, initially, we have $\text{size}(a_i) = m$ for all i .

Now, a **split** request can only be made to a cluster c that contains at least one of the a_i (otherwise c would contain points from only one target interval). Since the result of a **split** is to replace c with two clusters of half as many points, this means that each **split** request reduces $\text{size}(a_i)$ by a factor of 2 for at least one of the decision boundaries a_i . Furthermore, a **merge** request can only be made on two clusters c, c' such that $c \cup c'$ does not contain any of the a_i . Therefore, the total number of **split** requests can be at most $k \log m$ total, which implies a total of at most $k \log m$ **merge** requests as well. $\square$

Note that for intervals, the extended model with k clusters (where the last cluster can be a “default bucket”) can be expressed using the standard model with $2k - 1$ clusters, so Theorem 1 applies to the extended model as well.

4 Conjunctions and Disjunctions

We now present an algorithm for clustering the class C of disjunctions over $\{0, 1\}^n$. That is, each target cluster can be described by some disjunction of literals, and each point $x \in S$ satisfies exactly one of the target disjunctions. We show we can do this making only $O(n)$ queries if we allow the clusters in the hypothesis clusterings to overlap. We then show how this algorithm can be adapted to remove this overlap (so that each proposed clustering is indeed a partition of S), but at the expense of now making $O(n^2)$ queries. We finally show how these can be used to yield algorithms for the class C of conjunctions that are polynomial for constant k . Specifically, these algorithms make $O(n^{k-1})$ queries

and $O(n^{2(k-1)})$ queries respectively depending on whether or not disjointness is required.

We begin with a simple $O(n)$ -query algorithm for disjunctions, if we allow hypothesis clusters to overlap. Without loss of generality we may assume the target disjunctions are monotone (by the standard trick of introducing variables $y_i = 1 - x_i$ for every variable x_i).

Algorithm Cluster-Disjunction:

- Begin with one cluster for each variable x_i , containing all points $x \in S$ such that $x_i = 1$. (Note that these may overlap).
- On a **split** request to a cluster x_i , simply delete that cluster.
- On a **merge** request, merge the two clusters indicated.

Theorem 2. *Algorithm Cluster-Disjunction requires at most $n - k$ requests to cluster the class of disjunctions over $\{0, 1\}^n$.*

Proof: First, notice that whenever two clusters are merged, they can never be split, since by definition, the result of a merge operation is pure (contains points from only one target cluster). Therefore, all **split** requests are made to clusters corresponding to single variables. Second, note that if x_i is a relevant variable (belongs to one of the target disjunctions), then its associated cluster is pure and so will not be split. Since each point $x \in S$ must satisfy one of the target disjunctions, this means we maintain the (crucial) invariant that our clustering is *legal*: every point $x \in S$ belongs to at least one cluster in the hypothesis. Thus, we ensure that if our current hypothesis is incorrect, there must be a **split** or **merge** request the user can make. Since each request results in reducing the number of clusters by 1, this means that the total number of requests is at most $n - k$. $\square$

Notice that the above algorithm produces hypothesis clusterings in which the clusters are non-disjoint (because a given point $x \in S$ may have several variables set to 1). If we want the hypothesis clusters to be disjoint, a natural approach to doing so is to just arbitrarily remove each example x from all but one of the clusters i such that $x_i = 1$. However, this may cause the result of a **split** request to no longer be a legal clustering since some points x may no longer belong to any clusters, and arbitrarily re-assigning them may break the invariant that the clusters resulting from **merge** requests can never be split later. We can fix this problem, but at a loss of using $O(n^2)$ requests, as follows.

Algorithm Disjoint-Disjunction:

- Begin with one cluster for each variable x_i , containing all points $x \in S$ such that $x_i = 1$ and $x_j = 0$ for all $j < i$.
- On a **split** request to a cluster x_i , delete that variable from every point in S and restart the entire algorithm from the beginning (with $n \leftarrow n - 1$).
- On a **merge** request, merge the two clusters indicated.

Theorem 3. *Algorithm Disjoint-Disjunction maintains a disjoint clustering and requires at most $O(n^2)$ requests to cluster the class of disjunctions over $\{0, 1\}^n$.*

Proof: The initialization step of the algorithm insures that clusters are disjoint and furthermore include all points $x \in S$. As before, since the result of a **merge** operation must contain points in only one target cluster, any **split** request must be to a cluster corresponding to a single variable x_i . By assumption that the target clusters are disjunctions, any hypothesis clusters corresponding to *relevant* variables are pure, and therefore any **split** request must be to an irrelevant variable. So, deleting such variables maintains the invariant that each target cluster can be expressed as a disjunction. Since each **split** request reduces the total number of variables by 1, and there can be at most $n - k$ **merge** requests in a row, the total number of requests is $O(n^2)$. $\square$

We now show how the above algorithms can be used to cluster the class of conjunctions, making $O(n^{k-1})$ queries and $O(n^{2(k-1)})$ queries respectively depending on whether or not disjointness is required. In particular, let $c_1, c_2, \dots, c_k$ denote the target clusters and assume without loss of generality that each conjunction is monotone. Because each point $x \in S$ lies in exactly one target cluster, this means we can equivalently write cluster c_i as $\bar{c}_1 \wedge \dots \wedge \bar{c}_{i-1} \wedge \bar{c}_{i+1} \wedge \dots \wedge \bar{c}_k$. Since each c_j is a conjunction, this means we can write c_i as a $(k-1)$ -DNF, or equivalently as a disjunction over a space of n^{k-1} variables $y_{i_1 \dots i_{k-1}} = \bar{x}_{i_1} \bar{x}_{i_2} \dots \bar{x}_{i_{k-1}}$. Thus, we can cluster by expanding to this space of n^{k-1} variables and running the disjunction algorithms given above. The bounds then follow immediately.

4.1 Conjunctions and Disjunctions in the Extended Model

In the extended model (where we allow c_k to be a default cluster not necessarily in C), Algorithm **Cluster-Disjunction** “almost” works. The only problem is that deleting a cluster x_i may cause some points to become completely uncovered, because points in c_k do not have any relevant variables set to 1. However, since all points with no relevant variables must be in the same cluster c_k , we can fix this problem with a small modification to the algorithm. Specifically, we just create an extra “default bucket” containing all the points not covered by any of the other hypothesis clusters. This bucket will never be split (since all such points must be in c_k) so the analysis proceeds exactly as before. The same modification and analysis applies to Algorithm **Disjoint-Disjunction**: in particular, the default bucket will just contain all points that have no variables set to 1. Thus we have the following theorem.

Theorem 4. *The above modification to Algorithm **Cluster-Disjunction** requires at most $n - k + 1$ requests to cluster the class of disjunctions over $\{0, 1\}^n$ in the extended model. The modification to Algorithm **Disjoint-Disjunction** maintains disjoint clusters and requires at most $O(n^2)$ requests to cluster disjunctions in the extended model.*

For conjunctions, the difficulty with the reduction given in the non-extended model is that the expression $\bar{c}_1 \wedge \dots \wedge \bar{c}_{i-1} \wedge \bar{c}_{i+1} \wedge \dots \wedge \bar{c}_k$ for cluster c_i is no longer a $(k-1)$ -DNF, except for the case $i = k$. However, we can deal with this problem with the following procedure.

Algorithm Extended-Cluster-Conjunctions:

- Begin with one cluster for each term $y_{i_1 \dots i_{k-1}} = \bar{x}_{i_1} \bar{x}_{i_2} \dots \bar{x}_{i_{k-1}}$, containing all points in S satisfying this term. Instantiate a default bucket with all points not covered by any other cluster.
- On a **split** request to one of the y clusters, or a **merge** request to a y cluster and a cluster in the default bucket, delete the y cluster. If any points become uncovered, insert them into the default bucket and instantiate (or restart from scratch) a $(k-1)$ -cluster conjunction algorithm for the non-extended model on the points in that bucket.
- On a **merge** request to two y clusters, merge the two clusters indicated.
- On a **split** or **merge** request to two clusters within the default bucket, send them to the conjunction-learning algorithm being run on the datapoints in that bucket.

Theorem 5. *Algorithm Extended-Cluster-Conjunctions requires at most $O(n^{2k-3})$ requests to cluster the class of conjunctions over $\{0, 1\}^n$ in the extended model.*

Proof: Because cluster c_k can be written as a disjunction over the y variables, the relevant y variables for c_k will never receive **split** requests or be asked to be merged with clusters within the default bucket. Therefore, the above algorithm maintains the invariant that only points within $c_1 \cup \dots \cup c_{k-1}$ are ever placed into the default bucket. This implies that at most $O(n^{k-2})$ requests will be made to clusters inside that bucket between any two consecutive restarts to the $(k-1)$ -cluster conjunction-learning algorithm in that bucket. There can be at most n^{k-1} restarts, so overall the number of requests will be at most $O(n^{2k-3})$. $\square$

One can also consider a disjoint-cluster version of Algorithm **Extended-Cluster-Conjunctions**, by deleting y variables and restarting on **split** requests as in Algorithm **Disjoint-Disjunction**, and by using a disjoint clustering algorithm for the default bucket. In this case, we have at most $O(n^{2(k-2)})$ requests to the default bucket between restarts, and at most n^{k-1} restarts, resulting in at most $O(n^{3k-5})$ requests total.

5 A General Upper Bound

We now describe a generic but computationally inefficient algorithm that will cluster any concept class C using $O(k^3 \log |C|)$ queries. The high-level idea is that ideally we would like to use some form of halving algorithm, except that because feedback is in the form of **split** and **merge** requests rather than labeled examples, it is not so clear how to do this directly. What we will do instead is carefully construct a clustering such that any split or merge request removes at least a $1/k^2$ fraction of the version space, leading to the bound given above. We point out that the clustering constructed in each step may have many more than k clusters in it. However, this is unavoidable: as we show in Section 6, it is not possible to achieve a query complexity polynomial in k , $\log |C|$ and $\log m$ if we require the algorithm to use only $\text{poly}(k, \log |C|, \log m)$ clusters.

The generic algorithm is as follows. We begin with the simple “wrapper” and then present the main “engine” of the algorithm.

Algorithm Generic-Clustering:

1. Let C^{VS} denote the current version space: the set of clusterings consistent with the results of all queries so far. So, initially we have $|C^{VS}| \leq |C|^{k-1}$.
2. Run Algorithm **Generate-Interesting-Clustering**(C^{VS}, S) described below to produce a clustering that is guaranteed to have the property that any **split** or **merge** request removes a substantial fraction of the version space, and propose it to the user.
3. When a **split** or **merge** request is received, remove from C^{VS} all clusterings inconsistent with the request and go to (2).

We now give the main “engine” of the algorithm. Here we say that a cluster b is *consistent* with a clustering $\{c_1, \dots, c_k\}$ if $b \subseteq c_i$ for some i .

Algorithm Generate-Interesting-Clustering(C^{VS}, S):

1. Initialize k buckets $B_1, \dots, B_k$ (initially all empty) and let $\alpha = 1/k^2$. We will maintain the invariant that each B_i is consistent with at least a $1 - \alpha$ fraction of the clusterings in C^{VS} .
For each point $x \in S$ (in an arbitrary order) do:
 - (a) Insert x into the bucket B_i of least index such that $B_i \cup \{x\}$ is consistent with at least an α fraction of the clusterings in C^{VS} . If no such B_i exists, then halt with failure.
 - (b) If B_i is now consistent with less than a $1 - \alpha$ fraction of clusterings in C^{VS} , then output B_i as one of the hypothesis clusters, and replace it with a new empty bucket, putting the new bucket at the end of the list. That is, let $B_i \leftarrow B_{i+1}, B_{i+1} \leftarrow B_{i+2}, \dots, B_{k-1} \leftarrow B_k$ and call the new empty bucket B_k .
2. If we reach this point (no more points $x \in S$ and all buckets B_i are consistent with at least a $1 - \alpha$ fraction of C^{VS}), output the clusters $B_1, B_2, \dots, B_k$, ignoring empty buckets.

Theorem 6. *Algorithm Generic-Clustering succeeds in clustering any class C using at most $O(k^3 \log |C|)$ requests, and furthermore this holds in the extended model as well.*

Proof: We show that Algorithm **Generate-Interesting-Clustering** outputs a clustering such that any **split** or **merge** request is guaranteed to remove at least a $1/k^2$ fraction of clusterings from the version space C^{VS} . This will immediately imply that the total number of queries of Algorithm **Generic-Clustering** is at most $O(k^2 \log |C^{VS}|) = O(k^3 \log |C|)$ (in both standard and extended models) as desired.

First, we argue that the algorithm will never halt with failure in Step 1(a). By construction, we maintain the invariant that each bucket B_i is consistent with at least a $1 - \alpha$ fraction of clusterings in C^{VS} , since otherwise we would have already

outputted it in Step 1(b) (and by definition, an empty bucket is consistent with the entire C^{VS}). Therefore, at least a $1 - k\alpha$ fraction of C^{VS} is consistent with all B_i simultaneously. In addition, for each pair $i < j$ for which B_j is non-empty, at most an α fraction of C^{VS} can be consistent with $B_i \cup B_j$ (because each point $x \in B_j$ has the property that at most an α fraction of C^{VS} is consistent with $B_i \cup \{x\}$, else it would have been inserted into B_i instead). Moreover, we only care about the case that no buckets are empty since otherwise we could always put x into the first empty bucket. Therefore at least a $1 - k\alpha - \binom{k}{2}\alpha$ fraction of C^{VS} is consistent with each B_i and has all k buckets B_i as subsets of *distinct* clusters. By definition of α , this is at least a $k\alpha$ fraction of C^{VS} . Now, each of these clusterings has x in one of its k clusters, and that cluster is associated with a single bucket B_i . So, there must exist an index i such that at least a $k\alpha/k = \alpha$ fraction of C^{VS} has x in a cluster consistent with B_i , and therefore Step 1(a) succeeds.

We now argue that the clustering produced has the desired property that any **split** or **merge** request removes at least an α fraction of the version space. First, if the algorithm outputs a cluster in Step 1(b), then the fraction of clusterings in C^{VS} consistent with the cluster B_i produced is in the range $[\alpha, 1 - \alpha]$, and therefore any **split** or **merge** request involving it removes at least an α fraction of the version space. So, all clusters produced in Step 1(b) have the desired property. Next, we need to consider the clusters output in Step 2. For those, the situation is even better: any **split** or **merge** request involving only these clusters removes at least a $1 - \alpha$ fraction of the version space. In particular, for **split** requests this is because each B_i is consistent with at least a $1 - \alpha$ fraction of C^{VS} , and for **merge** requests this is because at most an α fraction of C^{VS} is consistent with $B_i \cup \{x\}$ for all $x \in B_j, j > i$. Thus, overall the clustering produced has the property that any request removes at least an $\alpha = 1/k^2$ fraction of the version space as desired. $\square$

6 Lower Bounds

We now show that if we restrict the algorithm to only producing clusterings with $\text{poly}(k, \log m)$ clusters, then there exist classes for which no algorithm can succeed. Thus, the use of what might potentially be a large number of small clusters in the generic algorithm of Section 5 is in fact necessary.

We begin first with a much easier statement to prove.

Theorem 7. *There exist classes C of size m such that, even for $k = 2$, any algorithm that is restricted to producing k -clusterings will require $\Omega(m)$ queries.*

Proof: Let S consist of m points on the circle, and let C be the class of intervals. Assume the target clustering is a random partition of S into two contiguous intervals of $m/2$ points. If the algorithm proposes a 2-clustering in which the two clusters have different sizes, then the larger cluster must contain points from both target clusters. So the user can issue a **split** request on that cluster providing the algorithm with no information (the algorithm could simulate the

user itself). Alternatively, if the algorithm proposes two equal-size clusters, the user can issue a **split** request on either cluster unless the algorithm is exactly correct. Since the target was chosen to be a random partition, this implies any algorithm must make $\Omega(m)$ queries in expectation. $\square$

We now present our main result of this section.

Theorem 8. *There exist classes C of size $O(m)$ such that, even for $k = 2$, no algorithm that is restricted to producing clusterings with only $\text{poly}(k, \log m)$ clusters can have even a $1/\text{poly}(k, \log m)$ chance of success after $\text{poly}(k, \log m)$ queries.*

Proof: Consider the set $S = \{0, 1\}^n$, so $m = 2^n$, and let C be the class of parity functions and their negations. (So, for some parity function c , one cluster will consist of all points $x \in \{0, 1\}^n$ such that $c(x) = 1$ and the other will consist of all points $x \in \{0, 1\}^n$ such that $c(x) = -1$.) We claim that for any query made by the algorithm, if the target is determined by a random parity c , the user will be able to issue a **split** request on the largest cluster with probability exponentially close to 1. Thus, the algorithm receives no information (since it could simulate such a user itself) and therefore exponentially many queries will be needed to cluster correctly (exponential in k and $\log m$).

Specifically, suppose the largest cluster c' in the algorithm's proposed clustering has size αm . Define the boolean function $h(x) = 1$ if $x \in c'$ and $h(x) = -1$ if $x \notin c'$. The user will be able to issue a **split** request on c' unless $c' \subseteq c$ or $c' \subseteq \neg c$; in the former case we have $\Pr[h(x) = c(x)] = 1/2 + \alpha$ and in the latter case we have $\Pr[h(x) = \neg c(x)] = 1/2 + \alpha$. Either of these cases imply that the magnitude of correlation between h and c satisfies:

$$\begin{aligned} |\mathbf{E}_x[h(x)c(x)]| &= |\Pr[h(x) = c(x)] - \Pr[h(x) \neq c(x)]| \\ &= |1/2 + \alpha - (1/2 - \alpha)| = 2\alpha, \end{aligned}$$

or in Fourier notation, we have $|\langle h, c \rangle| = 2\alpha$. However, by Parseval's identity [13], we know that h can have correlation of magnitude 2α with at most $1/(2\alpha)^2$ different parity functions. Thus, if c was chosen at random from among all 2^n parity functions, the probability that the algorithm's largest cluster c' truly is a subset of one of the target clusters is at most $1/(4\alpha^2 2^n)$. Since we assumed the algorithm produced clusterings with only $\text{poly}(k, \log m)$ clusters, it must be the case that $\alpha \geq 1/\text{poly}(k, \log m)$, and so the probability the user is not able to issue a **split** request on the largest cluster is exponentially small in k and $\log m$, as desired. $\square$

7 Relation to Equivalence Query Model

In the standard model of learning with equivalence queries, any class C can be learned using at most $\log |C|$ queries via the halving algorithm. However, some classes can be learned with many fewer queries, such as the class of concepts

having at most one positive example which requires only one query to learn. In contrast, we show here that in our framework, for *any* class C , for the case $k = 2$ there is a lower bound of $\Omega(\log |C^{VS}|)$ on the number of queries needed to cluster, where C^{VS} is the initial version space. Thus, for the extended model, this gives a lower bound of $\Omega(\log |C|)$ on the number of queries needed to cluster.

Theorem 9. *For any class C , for $k = 2$, any clustering algorithm must make $\Omega(\log |C^{VS}|)$ queries in the worst case, where C^{VS} is the initial version space.*

Proof: Let $\{h_1, h_2, \dots, h_t\}$ be a clustering produced by the clustering algorithm. We show there must exist a **split** or **merge** request that removes at most a $5/6$ fraction of the version space.

First, suppose the algorithm's clustering has only two clusters ($t = 2$). In that case, every clustering in C^{VS} except for (a) the trivial clustering that puts all points into a single cluster or (b) a clustering identical to $\{h_1, h_2\}$ must split either h_1 or h_2 or both. Without loss of generality, say that a majority of those $|C^{VS}| - 2$ clusterings split h_1 . Thus, a **split** request on h_1 must be consistent with at least $(|C^{VS}| - 2)/2$ clusterings in the version space.

Now, for the case $t > 2$, consider the first three clusters h_1, h_2, h_3 in the algorithm's clustering. Since all clusterings in C^{VS} have only two clusters, each must either split one of the h_i or else have two of the h_i inside the same cluster. Thus, for each clustering in C^{VS} , at least one of the 6 possible **split** or **merge** requests on $\{h_1, h_2, h_3\}$ must apply. Therefore there must exist some request that is consistent with at least a $1/6$ fraction of C^{VS} as desired. $\square$

8 Conclusions

In this paper we have analyzed the problem of determining the correct clustering of data from a bounded number of **split** and **merge** requests. We have provided efficient algorithms for several natural classes including disjunctions, conjunctions (for bounded k), and intervals, as well as a generic $O(k^3 \log |C|)$ upper bound for clustering any given class C . We also provide lower bounds for algorithms that use a bounded number of clusters and a separation result with respect to the standard model of learning with equivalence queries.

This model brings up several interesting open questions. First, can one improve the generic upper bound from $O(k^3 \log |C|)$ to $O(k \log |C|)$, i.e., gain a constant amount of information per query. Second, can one devise an efficient algorithm for conjunctions whose query complexity is polynomial in both k and n . Our generic algorithm implies this is possible information-theoretically but we do not know any efficient procedure. Finally, a natural domain for clustering with **split** and **merge** requests is image segmentation. From this perspective, it would be interesting to generalize the 1-dimensional results to 2 dimensions, ideally to the case where each cluster is a region defined by a limited number s of axis-parallel line segments as in the results on learning discretized geometric concepts using equivalence queries of Bshouty et al. [6].

More broadly, it would be interesting to further explore clustering with other natural forms of interactive feedback.

Acknowledgments. This work was supported in part by the National Science Foundation under grant CCF-0514922, by an IBM Graduate Fellowship, and by a Google Research Grant.

References

- [1] Achlioptas, D., McSherry, F.: On spectral learning of mixtures of distributions. In: Proceedings of the 18th Annual Conference on Learning Theory (2005)
- [2] Angluin, D.: Queries and concept learning. *Machine Learning* 2, 319–342 (1998)
- [3] Arora, S., Kannan, R.: Learning mixtures of arbitrary gaussians. In: Proceedings of the 33rd ACM Symposium on Theory of Computing (2001)
- [4] Balcan, M.-F., Blum, A., Vempala, S.: A Discriminative Framework for Clustering via Similarity Functions. In: Proceedings of the 40th ACM Symposium on Theory of Computing (2008)
- [5] Brubaker, S.C., Vempala, S.: Isotropic PCA and affine-invariant clustering. In: Proceedings of the 49th ACM Symposium on Foundations of Computer Science (2008)
- [6] Bshouty, N.H., Goldberg, P.W., Goldman, S.A., Mathias, H.D.: Exact learning of discretized geometric concepts. *SIAM J. Computing* 28(2), 674–699 (1998)
- [7] Dasgupta, A., Hopcroft, J., Kleinberg, J., Sandler, M.: On learning mixtures of heavy-tailed distributions. In: 46th IEEE Symposium on Foundations of Computer Science (2005)
- [8] Dasgupta, A., Hopcroft, J.E., Kannan, R., Mitra, P.P.: Spectral clustering by recursive partitioning. In: Proceedings of the 14th European Symposium on Algorithms, pp. 256–267 (2006)
- [9] Dasgupta, S.: Learning mixtures of gaussians. In: Proceedings of the 40th Annual Symposium on Foundations of Computer Science (1999)
- [10] Hellerstein, L., Pillaipakkamnatt, K., Raghavan, V.V., Wilkins, D.: How many queries are needed to learn? In: Proceedings of the 27th ACM Symposium on Theory of Computing (1995)
- [11] Jackson, J.: An efficient membership-query algorithm for learning dnf with respect to the uniform distribution. *Journal of Computer and System Sciences* 57(3), 414–440 (1995)
- [12] Kannan, R., Salmasian, H., Vempala, S.: The spectral method for general mixture models. In: Proceedings of the 18th Annual Conference on Learning Theory (2005)
- [13] Mansour, Y.: Learning boolean functions via the fourier transform. *Theoretical Advances in Neural Computation and Learning*, 391–424 (1994)
- [14] Vempala, S., Wang, G.: A spectral algorithm for learning mixture models. *Journal of Computer and System Sciences* 68(2), 841–860 (2004)

Active Learning of Group-Structured Environments

Gábor Bartók, Csaba Szepesvári*, and Sandra Zilles

University of Alberta, Department of Computing Science,
Edmonton, Alberta, Canada
{bartok,szepesva,zilles}@cs.ualberta.ca

Abstract. The question investigated in this paper is to what extent an input representation influences the success of learning, in particular from the point of view of analyzing agents that can interact with their environment. We investigate learning environments that have a group structure. We introduce a learning model in different variants and study under which circumstances group structures can be learned efficiently from experimenting with group generators (actions). Negative results are presented, even without efficiency constraints, for rather general classes of groups showing that even with group structure, learning an environment from partial information is far from trivial. However, positive results for special subclasses of Abelian groups turn out to be a good starting point for the design of efficient learning algorithms based on structured representations.

1 Introduction

The question investigated in this paper is to what extent an input representation influences the success of learning, in particular from the point of view of analyzing agents that can interact with their environment. For simplicity, we assume that the agent has a finite number of actions, the execution of which results in a change in the state of the environment. The goal of the agent is to learn to predict the outcomes of its actions in terms of the changes of the environment's state. Of course, the agent has to have some inputs or sensations that reveal information about the state of the environment. In the simplest case the sensations reveal all the details of the state. Then the question is if the representation of the sensations influences the speed of learning. Is it necessary to assume a good representation, or can we design learning methods capable of efficiently learning in a broad class of environments irrespective of the input representation? This is a fundamental question in learning (and more broadly, in artificial intelligence).

In search problems for example, if the state is given with a factored (vectorial) representation then memory-based heuristics can be used to create good heuristics that lead to efficient planning methods (*e.g.*, [4, 9, 10]). Further, with a factored input representation predictive models can be learned efficiently in

* Csaba Szepesvári is on leave from MTA SZTAKI, Budapest, Hungary.

some non-trivial classes of environments, such as when the environment can be represented as a Bayes net described with bounded depth decision trees [16].

At first sight efficient learning given an arbitrary input representation seems impossible. By efficient learning we mean learning well ahead of the time before the agent has seen all the states. The conventional way to evaluate if an agent has learnt its environment is to look at if the agent is able to predict its future sensations given any (hypothetical) action sequence [13, 11]. This model leads to an easy negative result: If the sensations of the agent are some arbitrary codes (names) for the states then it is clearly impossible for the agent to disambiguate between state names not encountered before.

Luckily, predicting the names of states is not necessary for an agent to be successful. Take, *e.g.*, a reinforcement learning agent whose primary interest is to predict rewards and eventually gather them. By building an internal model of the environment, such agents might be able to create plans for gathering rewards in a successful way. One (admittedly contrived) example that shows why predicting names is not necessary is as follows: Assume the agent has two actions and a reward is achieved iff the two actions are taken in a certain combination. The agent that discovers this action sequence can succeed without ever memorizing any of the state names. (However, the observations now have an internal structure, *i.e.*, the reward is separated from the state names.) A different example is a nicely structured environment, *e.g.*, when the agent has d actions and the state can be represented with a vector over a finite set and the actions influence mutually disjoint sets of the components of this vector. In this case the agent that assumes “independence” of the actions unless some experience contradicts this, can learn a representation that can be used to predict outcomes after $O(d)$ interactions, while the size of the state space is exponential in d .

1.1 The Approach of Learning Group-Structured Environments

Since learning without seeing all the states is impossible when the environment lacks structure, a natural goal is to require the time needed for learning to scale with how well-structured the environment is. We investigate this under the simplifying assumption that the environments can be represented as (mathematical) groups. This implies that these environments are deterministic and that for any sequence of actions the length of the orbit generated with the resulting composite action is independent of the starting state. Due to their strong structure, these environments look ideal for initiating the study. Moreover, there are many interesting environments which belong to this class, consider, *e.g.*, permutation games, such as Rubik’s cube [9], the Topspin Puzzle or the Pancake Puzzle [7].

We introduce a model for efficient learning of group-structured environments by exploration, imposing a bound on the number of actions the agent can take before converging to a correct conjecture about the target group environment.

Learnability results turn out to strongly depend on the underlying presentation and thus the set of generators given as input to the learner, both if efficiency constraints are maintained and if they are dropped. However, the strength of the negative results actually will show exactly what motivates our research, namely

that the success of efficient learning algorithms is affected by the choice of the representation. Our results indicate that even group structure is not enough to guarantee success—the choice of generators presented to the learner is also essential. We study this for finitely generated and for finite groups, for Abelian groups, and for the special case of dihedral groups. In particular the Abelian groups have a structure that is more promising for efficient learning and we show possible steps towards exploiting their structure.

1.2 Related Work

The closest to the present work is the work of Rivest and Schapire [13] who investigate the problem of inferring a finite automaton by interaction. (For extensions to stochastic environments see [8, 11].) They assume that the agent’s sensations come from a finite set, which might be different from the set of states. (In [13] the observations are binary vectors, but this assumption can be lifted.) According to the definition in [13] an environment is learned when the agent can predict future sensations for an arbitrary sequence of actions given its current state. They give a randomized algorithm that is capable of learning permutation automata (with high probability) in time that is polynomial in the “diversity” associated with the environment. The diversity is the number of equivalence classes of tests in the environment. A test is a sequence of actions and sensations. The outcome of a test in a given state is true or false depending on whether the test’s observations are sensed in the order specified in the test provided that the test’s actions are executed, again, in the order specified in the test. Two tests are equivalent if they give the same outcomes independently of the start state.

The diversity d of an environment is (tightly) bounded by $\log_2(n) \leq d \leq 2^n$, where n the number of states of the environment [13]. The diversity depends to a large extent on how the sensations (*i.e.*, the inputs) are chosen, underlining again the importance of working with the right inputs.

Our framework corresponds to the case when the sensations have a one-to-one correspondence with the states. Thus the diversity of the resulting environment is n . As we are interested in learning with $o(n)$ interactions, the present work can be viewed as an attempt to improve upon the results of [13].

We are not aware of any research on our setting of group learning. Related work concerning groups and learning has a focus completely different from that of the framework we introduce, see, *e.g.*, models of learning algebraic structures from positive data [15]. Studies on *black-box groups* [17], [2] analyze objects different from those in our group environment setting. Related, but not focusing on learning is, *e.g.*, [5] that introduced a probabilistic algorithm to decide whether an algebraic structure is an Abelian group and [?] that investigated how to derive the irreducible decomposition of a given linear representation of a group.

2 Preliminaries

In this section we introduce the basic notions used throughout the paper. We assume the reader to be familiar with a few basic group theoretic notions; those used without any further explanation are taken from [14].

$\mathbb{N}$ denotes the set of all natural numbers, $\mathbb{Z}$ the set of all integers. Let $\mathcal{G} = (S_{\mathcal{G}}, \circ_{\mathcal{G}})$ be a group, where $S_{\mathcal{G}}$ is the domain of $\mathcal{G}$ and $\circ_{\mathcal{G}}$ the group operation. We always use $\lambda_{\mathcal{G}}$ to denote the neutral element in $\mathcal{G}$, but drop the subscript $\mathcal{G}$ in $\lambda_{\mathcal{G}}$, in $S_{\mathcal{G}}$, and in $\circ_{\mathcal{G}}$ if the underlying group is clear from the context. If $A \subseteq S_{\mathcal{G}}$ is any subset of $S_{\mathcal{G}}$ then by A^* we denote the set $\{(a_1, \dots, a_m) \mid m \in \mathbb{N}\}$ of all finite sequences of elements in A , including the empty sequence. With every sequence $\alpha = (a_1, \dots, a_m)$ we associate a group element $\mathcal{G}(\alpha) \in S_{\mathcal{G}}$, namely

$$\mathcal{G}(\alpha) = \begin{cases} \lambda_{\mathcal{G}}, & \text{if } m = 0, \\ a_1 \circ_{\mathcal{G}} (a_2 \circ_{\mathcal{G}} (\dots \circ_{\mathcal{G}} (a_{m-1} \circ_{\mathcal{G}} a_m) \dots)), & \text{if } m > 0. \end{cases}$$

From now on we will omit the symbol for group operations, wherever it is clear from the context, *e.g.*, for two elements $a, b \in S$ we write ab rather than $a \circ_{\mathcal{G}} b$.

The elements of a set $A \subseteq S_{\mathcal{G}}$ are called *generators* of $\mathcal{G}$ if every element in S can be written as a product of elements in A , *i.e.*, if $S_{\mathcal{G}} = \{\mathcal{G}(\alpha) \mid \alpha \in A^*\}$. A *relation* in $\mathcal{G}$ is a sequence $\alpha \in S_{\mathcal{G}}^*$ such that $\mathcal{G}(\alpha_1) = \lambda_{\mathcal{G}}$. A pair $\langle A \mid R \rangle$ is called a *presentation* of $\mathcal{G}$ iff A is a set of generators for $\mathcal{G}$ and R is a set of relations in $\mathcal{G}$ such that $\mathcal{G} = F_A/(R)$, *i.e.*, $\mathcal{G}$ is the factor group of the free group on A and its smallest normal subgroup that contains R . For ease of presentation, we usually omit set brackets when writing $\langle A \mid R \rangle$ explicitly. For instance, a presentation for the (so-called) Klein group is $\langle a, b \mid a^2, b^2, (ab)^2 \rangle$ (rather than $\langle \{a, b\} \mid \{a^2, b^2, (ab)^2\} \rangle$). A presentation $\langle A \mid R \rangle$ is finite if both A and R are finite.

$\mathcal{G}$ is called (i) *finitely generated* if $\mathcal{G}$ has a finite set of generators; (ii) *finitely presented* if $\mathcal{G}$ has a finite presentation; (iii) *finite* if $S_{\mathcal{G}}$ is finite.

A *representation* of $\mathcal{G}$ over a vector space V is a homomorphism $\Phi : S_{\mathcal{G}} \mapsto \text{GL}(V)$, where $\text{GL}(V)$ is the *general linear group* of V .

For any $a \in S_{\mathcal{G}}$, $\langle a \rangle$ denotes the *cyclic subgroup* generated by a . C_k denotes the *cyclic group* of order k . A *p-group* is a finite group of order p^k , where p is a prime and k is a positive integer.

The *order* of an element $g \in S_{\mathcal{G}}$ is the lowest positive integer k such that $g^k = \lambda_{\mathcal{G}}$. We denote the order of g by $\sigma(g)$.

For any subset $\mathcal{H}$ we write $\mathcal{H} \leq \mathcal{G}$ if $\mathcal{H}$ is a subgroup of $\mathcal{G}$, and $\mathcal{H} \triangleleft \mathcal{G}$ if $\mathcal{H}$ is a normal subgroup of $\mathcal{G}$. If $\mathcal{H}_1 = (S_{\mathcal{H}_1}, \circ_{\mathcal{G}})$ and $\mathcal{H}_2 = (S_{\mathcal{H}_2}, \circ_{\mathcal{G}})$ are two subgroups of $\mathcal{G}$ then we define $S_{\mathcal{H}_1} S_{\mathcal{H}_2} = \{h_1 \circ_{\mathcal{G}} h_2 \mid h_1 \in S_{\mathcal{H}_1} \text{ and } h_2 \in S_{\mathcal{H}_2}\}$. The automorphism group of $\mathcal{G}$ is referred to by $\text{Aut}(\mathcal{G})$.

If $\mathcal{G}_1$ and $\mathcal{G}_2$ are two groups, then $\mathcal{G}_1 \times \mathcal{G}_2$ denotes their direct product. Here note again that for ease of presentation we identify groups with their domains.

Definition 1 (Semi-direct product: inner definition). Let $\mathcal{G} = (S_{\mathcal{G}}, \circ_{\mathcal{G}})$ be a group and $\mathcal{N} \triangleleft \mathcal{G}, \mathcal{H} \leq \mathcal{G}$ such that $S_{\mathcal{N}} S_{\mathcal{H}} = S_{\mathcal{G}}, S_{\mathcal{N}} \cap S_{\mathcal{H}} = \{\lambda_{\mathcal{G}}\}$. Let $\phi : S_{\mathcal{H}} \mapsto \text{Aut}(\mathcal{N})$ be the group homomorphism such that

$$\forall n \in S_{\mathcal{N}} \forall h \in S_{\mathcal{H}} [\phi(h)(n) = hnh^{-1}].$$

Then $\mathcal{G}$ is the semi-direct product of $\mathcal{N}$ and $\mathcal{H}$ with respect to ϕ , written $\mathcal{G} = \mathcal{N} \rtimes_{\phi} \mathcal{H}$.

3 A Model for Learning Group-Structured Environments

In this section we introduce our basic model of learning group-structured environments, the underlying scenario of which is as follows:

An agent is exploring a (finite or infinite) state environment. There is a finite set of actions that the agent can take in every state of the environment; taking an action usually causes the state of the environment to change. Now assume the agent can always observe the name of the state the environment is currently in and thus can always recognize when it gets back to some previously visited state. This allows the agent to find out relations between actions.

We assume the environment to be static and deterministic, *i.e.*, there is one function that determines for every state s and every action a the successor state after taking action a in state s . Formally this can be defined as follows:

Definition 2 (Environment). *An environment is a triple $\mathcal{E} = (S, A, T)$, where S is a countable set, A is a finite set, and $T: S \times A \mapsto S$ is a mapping.*

The elements of S are called states; the elements of A are called actions. For every $s \in S$ and every $a \in A$ we denote by $a(s)$ the state $T(s, a)$.

Fix an environment (S, A, T) . We now extend T to action sequences. For this we identify the set of action sequences with the free monoid $(A^*, \circ)$ (where $\circ$ is the concatenation over A^*). The empty action sequence, denoted by λ , is the identity element of this monoid. We extend the definition of T by $T(s, a_1 \dots a_m) = a_m(\dots(a_2(a_1(s))\dots))$ for all $(a_1 \dots a_m) \in A^*$. We use $(a_1 \dots a_m)(s)$ as a shorthand for $T(s, a_1 \dots a_m)$.

Definition 3 (Equivalence of action sequences). *Let $\mathcal{E} = (S, A, T)$ be an environment and $\alpha_1, \alpha_2 \in A^*$ be action sequences. Then α_1 and α_2 are equivalent in $\mathcal{E}$ (denoted by $\alpha_1 \equiv_{\mathcal{E}} \alpha_2$) iff $\alpha_1(s) = \alpha_2(s)$ for all $s \in S$. Let $S_{\mathcal{E}}$ denote the corresponding set of equivalence classes over A^* .*

The concatenation on A^* induces an operator $\circ$ on $S_{\mathcal{E}}$, which we will call concatenation, too. The subscript $\mathcal{E}$ in $\equiv_{\mathcal{E}}$ and/or $S_{\mathcal{E}}$ may be omitted when unambiguous.

We focus on scenarios in which the environment obeys a group structure.

Definition 4 (Group environment). *Let $\mathcal{E} = (S, A, T)$ be an environment. $\mathcal{E}$ is called a group environment if $(S_{\mathcal{E}}, \circ)$ is a group. With a slight abuse of notation we refer to $(S_{\mathcal{E}}, \circ)$ also by $\mathcal{E}$.*

If $\mathcal{E} = (S, A, T)$ is a group environment then A is a set of generators for the corresponding group $\mathcal{E}$, *i.e.*, we are always considering finitely generated groups.

We will analyze the learning problem of determining the structure of an unknown group environment $\mathcal{E} = (S, A, T)$ by exploration. In particular, we assume that the agent can take any action $a \in A$ in any state $s \in S$. After taking action a , the agent will observe the name of the state $a(s)$. In every step the agent outputs a hypothesis about the environment. The basic question we pose is: for certain classes of group environments, can the agent—in a “reasonable” number of steps—learn to solve the *word problem* for $\mathcal{E} = (S_{\mathcal{E}}, \circ)$?

Definition 5 (Word problem). Let $\mathcal{G}$ be a group and A a set of generators for $\mathcal{G}$. The word problem for $\mathcal{G}$ over A is solvable if there is a recursive decision procedure $d : A^* \rightarrow \{0, 1\}$ such that for all $w \in A^*$

$$d(w) = \begin{cases} 1, & \text{if } w \equiv \lambda_{\mathcal{G}}, \\ 0, & \text{if } w \not\equiv \lambda_{\mathcal{G}}. \end{cases}$$

Clearly, if the agent possesses d as defined above then he knows what action sequences are equivalent. This is enough for the agent to construct a consistent representation of the environment.

Consequently, we model the agent as a learning algorithm as follows:

Definition 6 (Learning algorithm). A learning algorithm is an algorithm L that fulfills the following properties.

1. L takes as its initial input a finite set A of actions and the name of a state $s_0 \in S$, where $\mathcal{E} = (S, A, T)$ is an unknown group environment.
2. L operates in steps, starting in Step 0, where in Step n for $n \in \mathbb{N}$ either (i) or (ii) holds.
 - (i) L sends some $a \in A$ to an oracle and receives $a(s_n)$ as a reply. Then L returns a recursive decision procedure D_n^1 and goes to Step $n + 1$ with $s_{n+1} = a(s_n)$.
 - (ii) L stops and never goes to Step $n + 1$.

Note that the group corresponding to a group environment could be any finitely generated group; so when defining learning models and making formal statements about learning group environments, we assume a one-one correspondence between finitely generated groups and group environments. Thus our learning criterion is defined as follows, based on Gold's [6] model of learning languages.

Definition 7 (Learning group environments). Let $\mathcal{C}$ be a class of groups and P a set of presentations such that $\mathcal{C} = \{\mathcal{G} \mid \mathcal{G} = \langle A \mid R \rangle \text{ for some } \langle A \mid R \rangle \in P\}$. For every $\mathcal{G} \in \mathcal{C}$, let $P_{\mathcal{G}} = \{\langle A \mid R \rangle \in P \mid \mathcal{G} = \langle A \mid R \rangle\}$.

We say that $\mathcal{C}$ is learnable $(*)$ in the limit, $(**)$ finitely, $(***)$ efficiently with respect to P if there is a polynomial q and a learning algorithm L that, for every group $\mathcal{G} \in \mathcal{C}$, every presentation $\langle A \mid R \rangle \in P_{\mathcal{G}}$, and every state s in the group environment corresponding to $\mathcal{G}$, the followings hold: If L is given (A, s) as an input then there is a decision procedure D that solves the word problem for $\mathcal{G}$ over A and there is an $n_0 \in \mathbb{N}$ such that the output of L in Step n equals D and

- $(*)$ for all $n \geq n_0$ the output of L in Step n equals D .
- $(**)$ L stops in Step $n + 1$.
- $(***)$ for all $n \geq n_0$ the output of L in Step n equals D and $n_0 \leq q(\log |S_{\mathcal{G}}| + \sum_{a \in A} \sigma(a))$.

¹ More precisely, L will return a program (on some suitable language) that represents the recursive decision procedure.

In the third point, $\sum_{a \in A} \sigma(a)$ appears in the upper bound because the learning algorithms should be allowed to determine the orders of the actions. One could think intuitively that $\sum_{a \in A} \sigma(a)$ and $\log(|S_G|)$ correlate polynomially, but in the case of non-Abelian groups, $|S_G|$ can be arbitrarily large, while $\sum_{a \in A} \sigma(a)$ is fixed. That instead of $|S_G|$ the bound includes $\log |S_G|$ is motivated by the desire to learn well before seeing all the states.

If $\mathcal{C}$ is a class of groups and P a set of presentations such that $\mathcal{C} = \{\mathcal{G} \mid \mathcal{G} = \langle A \mid R \rangle \text{ for some } \langle A \mid R \rangle \in P\}$ then we call $(\mathcal{C}, P)$ a *group learning problem*.

It does not make sense to consider efficient learning of infinite groups, because then the bound on the number of steps is infinite. The models of finite learning and learning in the limit of course can be studied also for infinite groups.

We close this section with a first result, namely that in certain cases learning algorithms can be normalized to operate with a restricted form of queries. For this purpose we introduce the notion of *0/1-learning algorithms*.

A 0/1-learning algorithm is defined very similarly to a learning algorithm in Definition 6, the only difference is that upon a query a sent to the oracle in Step n and state s_n , instead of $a(s_n)$ it receives the reply 1 if $a(s_n) = s_0$ and 0 if $a(s_n) \neq s_0$. The notion of 0/1-learnability (in the limit, finite, or efficient) can then immediately be derived from Definition 7 by replacing “learning algorithm” by “0/1-learning algorithm”. We call this concept learning with *binary observation*.

Lemma 1. *Let $(\mathcal{C}, P)$ be a group learning problem such that $\sigma(a)$ is finite for all $\langle A \mid R \rangle \in P$ and all $a \in A$. $\mathcal{C}$ is learnable in the limit (finitely learnable, efficiently learnable) with respect to P iff $\mathcal{C}$ is 0/1-learnable in the limit (finitely 0/1-learnable, efficiently 0/1-learnable) with respect to P .*

Proof. We only show that a 0/1-learning algorithm can be constructed from a learning algorithm according to Definition 6; the other direction is trivial.

The idea is that all the information gathered by the original learner up to some stage is whether a subsequence of the action sequence posed leaves a state unchanged. This information can be recovered with the binary observations by testing all subsequences from the initial state. This way the number of queries is blown up only polynomially.

Formally, the 0/1-learner works as follows: Initially, starting from s_0 , the learner experiments with all the actions:

- For each $a \in A$ repeatedly query a until the answer 1 is received and then set $\sigma(a)$ equal to the number of times a was queried.

Now, assume a learning algorithm L as in the original definition poses the query a_n in Step n . As a response the 0/1-learner does the following:

- Let $\alpha = (x_0, \dots, x_t)$ be the action sequence

$$\circ_{i=0}^n [(a_i, a_{i+1}, \dots, a_n, a_n^{-1}, \dots, a_{i+1}^{-1}, a_i^{-1})].$$

Query the actions in this sequence one by one.

- Let M be the set of all subsequences $x_j, \dots, x_{j+z}$ of α such that
 - $j = 0$ or the reply after querying x_{j-1} was 1; and
 - the reply after querying x_{j+z} was 1.
- If there exists an $i \leq n$ such that $(a_i, \dots, a_n) \in M$ then return the state name returned in Step $i - 1$ for the minimal such i (or return s_0 if $i = 0$). If there is no such i then return a new state name.
- Return the hypothesis that L returns.

Obviously this 0/1-learner solves all the learning problems that L solves at the price of a polynomial increase of execution time. $\square$

4 An Analysis of the Group Learning Model

In this section we analyze our learning models first for the classes of all finitely generated and all finite groups. The results and proofs will motivate the analysis of some special subgroups like the dihedral groups and Abelian groups.

4.1 General Results on Learning Group Environments

Starting with a quite general case, the class of all finitely generated groups, one easily obtains a negative result, independent of the set of presentations chosen.

Theorem 1

1. *The class of finitely generated groups is not learnable in the limit with respect to any set of presentations.*
2. *The class of finitely presented groups is not learnable in the limit with respect to any set of presentations.*

Proof. Both assertions follow immediately from the fact that there is a finitely presented (and thus a finitely generated group) $\mathcal{G}$ such that, for every presentation $\langle A \mid R \rangle$ of $\mathcal{G}$, the word problem over A^* is unsolvable [3, 12]. $\square$

Restricting our focus to classes of finite groups (for which the word problem is always solvable) we at least get learnability in the limit, yet efficient learnability is not achievable.

Theorem 2. *Let $\mathcal{C}$ be the class of all finite groups and P the set of all presentations of finite groups.*

1. *$\mathcal{C}$ is not learnable efficiently with respect to P .*
2. *$\mathcal{C}$ is learnable finitely with respect to P .*
3. *$\mathcal{C}$ is learnable in the limit with respect to P .*

Proof. The third assertion is easy to prove; a learning algorithm can exhaustively explore the effects of the actions.

The proof of the second assertion requires a dove-tailing argument by interleaving two procedures: one exploring action sequences in order to determine

group relations, the other trying to find a $k \in \mathbb{N}$ such that all action sequences of length k are equivalent to a shorter action sequence. Such a k must exist and can be found in a finite number of steps after enough relations are known. Knowing such a k , exploration using all sequences of length up to k will yield all further relations necessary to uniquely identify the target group.

A detailed proof we only give for the first assertion.

Assume $\mathcal{C}$ is learnable efficiently with respect to P . Then there is a learning algorithm L and a polynomial q such that L learns every finite group $\mathcal{G}$ with domain $S_{\mathcal{G}}$ efficiently from any of its sets A of generators, using no more than $q(\log(|S_{\mathcal{G}}|) + \sum_{a \in A} \sigma(a))$ many steps. By Lemma 1 we can assume without loss of generality that L is a 0/1-learning algorithm.

Note that there is an $m_* \in \mathbb{N}$ such that $2q(m) > q(\log(2q(m) + 2) + 4)$ for all $m \geq m_*$.

We define two groups and show that they cannot be distinguished by L :

$$\begin{aligned} - \mathcal{G}_1 &= \langle a, b \mid a^2, b^2, (ab)^{q(m_*)} \rangle \\ - \mathcal{G}_2 &= \langle a, b \mid a^2, b^2, (ab)^{q(m_*)+1} \rangle \end{aligned}$$

We will show below that (i) the size of the domain of $\mathcal{G}_1$ is $2q(m_*)$, (ii) the size of the domain of $\mathcal{G}_2$ is $2q(m_*) + 2$, (iii) for all $\alpha \in \{a, b\}^*$ with $|\alpha| < 2q(m_*)$:

$$\alpha \equiv_{\mathcal{G}_1} \lambda_{\mathcal{G}_1} \iff \alpha \equiv_{\mathcal{G}_2} \lambda_{\mathcal{G}_2}$$

This implies that $\mathcal{G}_1$ and $\mathcal{G}_2$ are distinct finite groups but L cannot distinguish $\mathcal{G}_1$ from $\mathcal{G}_2$ by asking $q(\log(|S_{\mathcal{G}_2}|) + \sigma(a) + \sigma(b)) = q(\log(2q(m_*) + 2) + 4) (< 2q(m_*))$ many queries.² This is a contradiction, so the class of all finite groups is not learnable efficiently with respect to all presentations of finite groups.

Claim 1. Let $\alpha \in \{a, b\}^*$ with $|\alpha| < 2q(m_*)$. Then $\alpha \equiv_{\mathcal{G}_1} \lambda_{\mathcal{G}_1} \iff \alpha \equiv_{\mathcal{G}_2} \lambda_{\mathcal{G}_2}$.

Proof of Claim 1. We show one direction only, the other one is similar.

Let α be as given and assume that $\alpha \equiv_{\mathcal{G}_2} \lambda_{\mathcal{G}_2}$. We know that α can be reduced to a $\mathcal{G}_1$ -equivalent sequence α' with $0 \leq |\alpha'| \leq |\alpha|$, such that α' does not contain the substrings aa or bb . If $|\alpha'| = 0$ then $\alpha \equiv_{\mathcal{G}_1} \lambda_{\mathcal{G}_1}$. We show that if $0 < |\alpha'| < 2q(m_*)$ then $\alpha \not\equiv_{\mathcal{G}_2} \lambda_{\mathcal{G}_2}$. This is a contradiction and we are done.

So assume $0 < |\alpha'| < 2q(m_*)$. α' results from α by iteratively applying the following rules.

(I) insert or delete aa (II) insert or delete bb (III) insert or delete $(ab)^{2q(m_*)}$

We introduce a function $\mu : \{a, b\}^* \mapsto \mathbb{Z}$. For any $\alpha = (a_1, \dots, a_n) \in \{a, b\}^*$ let $\alpha_{\text{odd}} = (a_1, a_3, \dots, a_{n_1})$ and $\alpha_{\text{even}} = (a_2, a_4, \dots, a_{n_2})$, where $n_1 = n_2 + 1 = n$ for odd n and $n_1 + 1 = n_2 = n$ for even n . Let $\mu(w) = (\# \text{ of 'a's in } \alpha_{\text{odd}}) - (\# \text{ of 'a's in } \alpha_{\text{even}}) + (\# \text{ of 'b's in } \alpha_{\text{even}}) - (\# \text{ of 'b's in } \alpha_{\text{odd}})$. For example, if $\alpha = aaababbbabab$, then $\alpha_{\text{odd}} = aaabbb$ and $\alpha_{\text{even}} = abbaa$, so $\mu(\alpha) = -1$.

When modifying α to α' , the parities of the old/remaining letter positions do not change. Therefore, only the new/disappearing letters will affect the value of

² Note here that $\sigma(a) = \sigma(b) = 2$ holds in both groups; technically we should in fact have subscripts with σ in order to relate it to a specific group.

μ . Using rules (I) or (II), the value of μ will not change at all. The use of rule (III), depending on its type (insert or remove) and the position (odd or even) chosen, will either increase or decrease the value of μ by $2q(m_*)$.

This implies that if $\alpha_1 \equiv_{\mathcal{G}_1} \alpha_2$ then $\mu(\alpha_1) \equiv \mu(\alpha_2) \pmod{2q(m_*)}$.

Since α' does not contain the substrings aa or bb , we have $\mu(\alpha') = \pm|\alpha'|$. Obviously, $\mu(\lambda) = 0$. Since $0 < |\alpha'| < 2q(m_*)$, $\mu(\alpha') \not\equiv \mu(\lambda) \pmod{2q(m_*)}$. Therefore $\alpha' \not\equiv_{\mathcal{G}_2} \lambda_{\mathcal{G}_2}$. $\square$ *Claim 1.*

Claim 2. The size of the domain of $\mathcal{G}_1$ is $2q(m_*)$; the size of the domain of $\mathcal{G}_2$ is $2q(m_*) + 2$.

Proof of Claim 2. We prove the statement only for $\mathcal{G}_1$; for $\mathcal{G}_2$ the proof is similar. Note that two action sequences α and β are equivalent in $\mathcal{G}_1$, if β results from α by iteratively eliminating all substrings aa , bb , $(ab)^{q(m_*)}$. Hence every element in the domain of $\mathcal{G}_1$ can be written as a product in which none of these subsequences occur. These are

$$a(ba)^k, b(ab)^k, (ab)^k, (ba)^k$$

for $0 \leq k < q(m_*)$ (note that $(ab)^0 = (ba)^0 = \lambda$).

Now note that $(ba)^k \equiv_{\mathcal{G}_1} (ab)^{2q(m_*)-k}$ for $0 \leq k \leq 2q(m_*)$ (both forms obviously have the inverse $(ab)^k$). Hence also $a(ba)^k \equiv_{\mathcal{G}_1} a(ab)^{2q(m_*)-k} \equiv_{\mathcal{G}_1} (ba)^{2q(m_*)-k-1}b \equiv_{\mathcal{G}_1} b(ab)^{2q(m_*)-k-1}$ for $0 \leq k < q(m_*)$.

Hence the domain of $\mathcal{G}_1$ has exactly $2q(m_*)$ elements. $\square$ *Claim 2.*

This completes the proof. $\square$

$\mathcal{G}_1$ and $\mathcal{G}_2$ belong to the well-known class of finite dihedral groups. So not even those are learnable efficiently if all possible presentations are taken into account in the group learning problem. Since the dihedral groups illustrate a few principled properties of our learning model, we study them in more detail.

4.2 Learning Dihedral Groups

We will use the dihedral groups to illustrate two general phenomena in learning theory, namely the impact of the representation scheme for the information given to the learning algorithm and the impact that slight changes to the class of target concepts can have on learnability.

For every $k \geq 1$, let D_k denote the finite dihedral group with $2k$ elements. Note that there is only one infinite dihedral group, namely $D_\infty = \langle a, b \mid a^2, b^2 \rangle$.

Theorem 2 and its proof immediately yield the following corollary.

Corollary 1. Let $\mathcal{C}$ be the class of finite dihedral groups and $P = \{ \langle A \mid R \rangle \mid \langle A \mid R \rangle \in \mathcal{C} \text{ and } |A| = 2 \}$.

1. $\mathcal{C}$ is not learnable efficiently with respect to P .
2. $\mathcal{C}$ is learnable finitely with respect to P .
3. $\mathcal{C}$ is learnable in the limit with respect to P .

In fact it is not very surprising that the finite dihedral groups cannot be learned efficiently with respect to presentations with 2 generators. The reason is simply

that any learning algorithm would have to make a number of experiments that is linear in the size of the group (rather than logarithmic). This leads us to a nice observation showing how much the choice of generator systems, *i.e.*, the representation of the input to the learning algorithm, influences learnability. In fact the finite dihedral groups can be learned efficiently if we choose a set of presentations in which the size of the group is linear in the order of one of the generators, thus allowing for enough experiments to identify the target group.

Fact 1. *Let $\mathcal{C}$ be the class of finite dihedral groups and $P = \{\langle c, d \mid c^2, d^k, cdcd \rangle \mid k \geq 1\}$. Then $\mathcal{C}$ is learnable efficiently with respect to P .*

The proof is omitted; note that with $a = c$, $b = cd$, and $ab = d$ we have the equivalence of the group presentations $\langle a, b \mid a^2, b^2, (ab)^k \rangle$ and $\langle c, d \mid c^2, d^k, cdcd \rangle$.

The second phenomenon that can be easily illustrated using dihedral groups is how unstable learnability results can be with respect to slight changes to the target class. The class of all finite dihedral groups can be learned finitely from all binary generator systems, but this is no longer true if we add just a single group to this class, namely the infinite dihedral group.³

Theorem 3. *Let $\mathcal{C}^+$ be the class of all dihedral groups and $P^+ = \{\langle A \mid R \rangle \mid \langle A \mid R \rangle \in \mathcal{C}^+ \text{ and } |A| = 2\}$.*

1. $\mathcal{C}^+$ is not learnable finitely with respect to P^+ .
2. $\mathcal{C}^+$ is learnable in the limit with respect to P^+ .

Proof. *ad 1.* Assume $\mathcal{C}^+$ is learnable finitely with respect to P^+ . Then there is a 0/1-learner L that learns every dihedral group finitely from any set of two generators. Simulate L on input $\{a, b\}$ as follows (simulate an oracle in parallel).

Always reply 0 for the first action L takes. If the first two actions by L are equal, *i.e.*, they are aa or bb , then reply 1 after the second action. In any other case, for growing action sequences, reply 0.

This scenario is valid if D_∞ is the target group. Thus eventually L will return a decision procedure for D_∞ and stop the process. At this point of the learning process there are infinitely many finite dihedral groups consistent with the scenario. These are not identified by L although they are in $\mathcal{C}^+$ —a contradiction.

ad 2. A learning algorithm on input $\{a, b\}$ initially tries to determine the order of both of the generators. In case one of these orders is $k \neq 2$, the algorithm will return a decision procedure for D_k forever. As long as one of the orders is not yet determined, the algorithm returns a decision procedure for D_∞ .

In case both generators turn out to be of order 2, the algorithm tries to determine the minimal k such that $\lambda \equiv (ab)^k$, if such a k exists. As long as no such k is found, the algorithm returns a decision procedure for D_∞ . As soon as such a k is found, the algorithm will start returning a decision procedure for D_k forever.

It is not hard to see that this algorithm witnesses Assertion 2. □

³ This very much resembles results in Inductive Inference, where Gold [6] showed that no class of languages that contains all finite languages and at least one infinite language, can be learned in the limit.

4.3 Learning Abelian Groups

For finitely generated Abelian groups, a reasoning similar to the proof of Theorem 3 shows similar differences between learning in the limit and finite learning.

Theorem 4. *Let $\mathcal{C}$ be the class of all finitely generated Abelian groups. Let $P = \{\langle a_1, \dots, a_k \mid R \rangle \mid \langle a_1 \rangle \times \dots \times \langle a_k \rangle \in \mathcal{C}\}$.*

1. $\mathcal{C}$ is learnable in the limit with respect to P .
2. $\mathcal{C}$ is not learnable finitely with respect to P .

Proof. *ad 1.* According to the fundamental theory of finitely generated Abelian groups every group in $\mathcal{C}$ has the form $\mathbb{Z}^k \times C_{n_1}^{k_1} \times \dots \times C_{n_z}^{k_z}$ for some $k, z, k_i, n_i \in \mathbb{N}$. The number of generators here is $k + k_1 + \dots + k_z$.

A learning algorithm L witnessing Theorem 4 works as follows: Given a set $A = \{a_1, \dots, a_k\}$ of generators, L initially always outputs a decision procedure for $\mathbb{Z}^{|A|}$. Given a canonical enumeration of all pairs (a_l, t) for $t \geq 1$ and $l \in \{1, \dots, k\}$, L then queries the state names while taking action sequences $(a_l)^t$ for all (l, t) in canonical order. Whenever a sequence $(a_l)^t$ takes L back to the state it was in before this sequence, L changes its output from $\mathbb{Z}^{k'} \times C_{n_1}^{k'_1} \times \dots \times C_{n_z}^{k'_z}$ to $\mathbb{Z}^{k'-1} \times C_{n_1}^{k'_1} \times \dots \times C_{n_j}^{k'_j+1} \times \dots \times C_{n_z}^{k'_z}$, where $n_j = t$. Moreover, from then on all pairs (a_l, t') in the enumeration will be skipped.

It is easy to prove that L learns all groups in $\mathcal{C}$ in the limit.

ad 2. Assume to the contrary that $\mathcal{C}$ is finitely learnable with respect to P , witnessed by a learning algorithm L .

Consider the behaviour of L for the target group $\mathbb{Z}$, generated by a single element. After finitely many experiments using this generator element, each of which leads L to a new state, L returns a decision procedure for $\mathbb{Z}$ and terminates.

At this point there are still infinitely many finite cyclic groups $C_n \in \mathcal{C}$ consistent with the scenario experienced by L ; L fails to identify them. $\square$

Concerning efficient learning, some properties of finite Abelian groups motivate the study of different kinds of learners, as introduced in the next section.

5 Learning Injective Representations

Every finite Abelian group can be written as a direct product of p -groups (for different primes p). This kind of group presentation turns out to be well suited for efficient learning, even for a specific kind of learning algorithms, namely some that return representations instead of decision procedures. Note that a decision procedure for a group can easily be obtained from a representation of the group.

A *representation-learning algorithm* is defined similarly to a learning algorithm (see Definition 6), except for that the outputs of the algorithm are always injective representations over $\mathbb{C}$ (of course with a correct representation in the end). Definitions of representation-learning are derived as usual.

Theorem 5. *Let $\mathcal{C}$ be the class of all finite Abelian groups. Let $P = \{\langle a_1, \dots, a_k \mid R \rangle \mid \mathcal{G} = \langle a_1 \rangle \times \dots \times \langle a_k \rangle \text{ and } \sigma(a_i) < \infty \text{ for all } i \in \{1, \dots, k\}\}$. $\mathcal{C}$ is efficiently representation-learnable from P .*

Proof. (Sketch.) We define a learning algorithm L on input A as follows:

For all $a \in A$, L determines $\sigma(a)$ by taking the action a until s_0 is reached.

Then L outputs a representation Φ , where $\Phi(a_i)$ is defined as a diagonal matrix where the i^{th} element of the diagonal is a primitive $\sigma(b_i)^{\text{th}}$ complex unit root; the other diagonal entries are 1. $\square$

Here again we assume a specific underlying presentation. In the case of learning with respect to all possible presentations this result can still be used to show a reducibility between our learning problem and the problem of cutting down a set of generators for a p -group $\mathcal{G}$ to an independent set of generators for $\mathcal{G}$.

Formally we define the problem of finding independent generators as follows:

Given a prime p , given a generator system A for an unknown p -group $\mathcal{G}$, find a subset of A that is an independent generator system for $\mathcal{G}$.

Theorem 6. *Let $\mathcal{C}$ be the class of all finite Abelian groups. Let P be the set of all presentations of finite Abelian groups. The problem of learning $\mathcal{C}$ efficiently with respect to P by representation is polynomially reducible to the problem of efficiently finding independent generators.*

Proof. (Sketch.) The idea is—given a generator system A —to construct generators for p -groups (with different primes p) such that the target group can be written as a direct product of these p -groups. In order to apply Theorem 5 we need to make sure that the generators for the p -groups are independent from each other, using an efficient algorithm for finding independent generators. $\square$

To close the section on representation-learning we discuss a result not related to Abelian groups but especially motivated by our original scenario of an agent exploring an unknown environment. Assume the agent has successfully learned a group environment but after that a new action is introduced to the environment. We model this as a problem of finding a representation of a single extension of a group $\mathcal{G}$ (for instance in the form of a semi-direct product) if a representation of $\mathcal{G}$ is known.

Suppose we have a group $\mathcal{G}$ with a generator system A and a d -dimensional linear representation Φ . Our aim is to extend $\mathcal{G}$ by a new generator element a , i.e., we want to construct a representation for a single extension $\mathcal{G}' = \mathcal{G} \rtimes_{\phi} \langle a \rangle$. Let us assume in addition that $\sigma(a) = \sigma(\phi(a)) = \sigma$. The new representation Φ' can then be defined as follows, where we abbreviate $\phi(a)$ by ϕ :

$$\Phi'(g) = \begin{pmatrix} \Phi(g) & 0 & \cdots & 0 \\ 0 & \Phi(\phi(g)) & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & \Phi(\phi^{\sigma-1}(g)) \end{pmatrix} \text{ if } g \in S_{\mathcal{G}}, \quad \Phi'(a) = \begin{pmatrix} 0 & I_d & 0 & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & I_d \\ I_d & 0 & \cdots & 0 \end{pmatrix}$$

That this indeed yields the desired representation is implied by the following easy to verify properties.

1. $\sigma(\Phi'(a)) = \sigma$.
2. The group resulting from restricting Φ' to $\mathcal{G}$ is isomorphic to $\mathcal{G}$.
3. $\Phi'^{-1}(a)\Phi'(g)\Phi'(a) = \Phi'(\phi(g))$ for all $g \in S_{\mathcal{G}}$.
4. $\Phi'(a)$ is not contained in the image of $\mathcal{G}$ under Φ' .

The dimension of the constructed representation is $d\sigma$. This construction is in general not trivial because of the problem of calculating $\Phi(\phi(g))$. However, a special case of this construction can be used for proving the following result.

Theorem 7. *Let $P = \{\langle a_1, a_2 \mid R \rangle \mid \langle a_1, a_2 \mid R \rangle = \langle a_1 \rangle \rtimes_{\phi} \langle a_2 \rangle, \sigma(\phi(a_2)) = \sigma(a_2), \sigma(a_1) < \infty, \sigma(a_2) < \infty\}$. Let $\mathcal{C} = \{\mathcal{G} \mid \mathcal{G} = \langle A \mid R \rangle \text{ for some } \langle A \mid R \rangle \in P\}$. $\mathcal{C}$ is efficiently representation-learnable with respect to P .*

Proof. (Sketch.) A learning algorithm L on input $\{a_1, a_2\}$ works as follows:

First, L determines $\sigma(a_1)$ and $\sigma(a_2)$ in the usual way. Second, L experiments with

$$a_1 a_2 a_1^{-1} \underbrace{a_2 a_2 \dots a_2}_{\sigma(a_2)} a_1 a_2^{-1} a_1^{-1};$$

similarly with a_1 and a_2 swapped. (For example, if $a_1 a_2 a_1^{-1} a_2^6 \equiv \lambda$, then $\mathcal{G} = \langle a_2 \rangle \rtimes_{\phi} \langle a_1 \rangle$ with $\phi(a_2) = a_2^{-6}$.) Third, L constructs the linear representation as described above Theorem 7, knowing that $\Phi(a_2) = (\sqrt[\sigma(a_2)]{1}) \in \mathbb{C}^{1 \times 1}$ is a primitive complex unit root. $\square$

6 Conclusions

We introduced and analyzed a model for (efficient) learning of group-structured environments by exploration. In order to capture the idea that an agent should learn its environment without visiting all the states, we imposed a bound on the number of actions the agent can take up to convergence to a correct conjecture about the target group environment.

Learnability results strongly depend on the underlying presentation and thus the set of generators given as input to the learner, both if efficiency constraints are maintained and if they are dropped. Our negative results suggest that it is in general too strong a requirement to learn with respect to all possible presentations of a group—which is in fact not surprising and gives answers to some of the questions we posed in the introduction.

A direction for future work clearly is to characterize cases in which the size of a minimal representation of a group (in a general coding scheme for finite or finitely generated groups) is logarithmic instead of linear in the size of the group. This is for instance the essence of some of the contrasting results concerning dihedral groups. Relaxations of the learning model would be a further natural and quite promising extension of our approach.

Acknowledgements

We thank Barnabás Póczos, András Antos and Marcus Hutter for helpful discussions. Thanks are also due to the anonymous referees for their insightful comments.

This research was funded in part by the National Science and Engineering Research Council (NSERC), iCore and the Alberta Ingenuity Fund.

References

- [1] Babai, L., Fried, K.: Approximate representation theory of finite groups. In: Proc. 32nd Annual Symposium on Foundations of Computer Science, pp. 733–742 (1991)
- [2] Babai, L., Szemerédi, E.: On the complexity of matrix group problems I. In: IEEE Symposium on Foundations of Computer Science (1984)
- [3] Boone, W.W.: The Word Problem. *The Annals of Mathematics* 70, 207–265 (1959)
- [4] Culberson, J.C., Schaeffer, J.: Pattern databases. *Computational Intelligence* 14, 318–334 (1998)
- [5] Friedl, K., Ivanyos, G., Santha, M.: Efficient testing of groups. In: Proc. 37th Annual ACM Symposium on Theory of Computing, pp. 157–166 (2005)
- [6] Gold, E.M.: Language identification in the limit. *Inform. Control* 10, 447–474 (1967)
- [7] Holte, R., Grajkowski, J., Tanner, B.: Hierarchical heuristic search revisited. In: Symposium on Abstraction, Reformulation and Approximation (2005)
- [8] Jaeger, H.: Observable operator models for discrete stochastic time series. *Neural Computation* 12, 1371–1398 (2000)
- [9] Korf, R.E.: Finding optimal solutions to Rubik’s cube using pattern databases. In: AAAI/IAAI, pp. 700–705 (1997)
- [10] Korf, R.E.: Analyzing the performance of pattern database heuristics. In: Proc. 22nd AAAI Conference on Artificial Intelligence, pp. 1164–1170 (2007)
- [11] Littman, M.L., Sutton, R., Singh, S.: Predictive representations of state. In: Advances in Neural Information Processing Systems 14, pp. 1555–1561 (2002)
- [12] Novikov, P.S.: On the algorithmic undecidability of the word problem in group theory. In: Proc. Steklov Institute of Mathematics, vol. 44, pp. 1–143 (1955) (in Russian)
- [13] Rivest, R.L., Schapire, R.E.: Diversity-based inference of finite automata. *J. ACM*, 555–589 (1994)
- [14] Rothman, J.J.: *An Introduction to the Theory of Groups*. Springer, Heidelberg (1995)
- [15] Stephan, F., Ventsov, Y.: Learning algebraic structures from text. *Theoret. Comput. Sci.* 268(2), 221–273 (2001)
- [16] Strehl, A.L., Diuk, C., Littman, M.L.: Efficient structure learning in factored-state MDPs. In: Proc. 22nd AAAI Conference on Artificial Intelligence, pp. 645–650 (2007)
- [17] Vinodchandran, N.V.: *Counting Complexity and Computational Group Theory*. PhD thesis, Institute of Mathematical Sciences, Chennai, India (1999)

Finding the Rare Cube

Shlomo Hoory and Oded Margalit

IBM Haifa Research Labs
{shlomoh, odedm}@il.ibm.com

Abstract. In this paper we investigate the problem of active learning the partition of the n -dimensional hypercube into m cubes, where the i -th cube has color i . The model we are using is **exact** learning via **color evaluation queries**, without equivalence queries, as proposed by the work of Fine and Mansour.¹ We give a randomized algorithm solving this problem in $O(m \log n)$ expected number of queries, which is tight, while its expected running time is $O(m^2 n \log n)$.

Furthermore, we generalize the problem to allow partitions of the cube into m monochromatic parts, where each part is the union of p cubes. We give two randomized algorithms for the generalized problem. The first uses $O(mp^2 2^p \log n)$ expected number of *queries*, which is almost tight with the lower bound. However, its naïve implementation requires an exponential running time in n . The second, more practical, algorithm achieves a better running time complexity of $\tilde{O}(m^2 n^2 2^{2p})$. However, it may fail to learn the correct partition with an arbitrarily small probability and it requires slightly more expected number of queries: $\tilde{O}(mn 4^p)$, where the $\tilde{O}$ represents a poly logarithmic factor in $m, n, 2^p$.

1 Introduction

Scientific research is the investigation of phenomena to acquire knowledge. In the process we gather information on the world and try to fit it into our understanding of it. We are looking for examples of different (interesting) phenomena. Usually such examples are rare, otherwise they would have been discovered sooner. In this paper we consider a simple mathematical model of the world in which we can formally quantify the cost of finding such examples. The motivating problem that led to this work, comes from the domain of hardware verification.

Hardware verification is the process of verifying that the actual hardware built complies with its intended design. One of the standard methods of achieving this goal, is to generate stimuli for the design under test (DUT), and compare the actual behavior of the DUT to a reference model. Since it is not feasible to cover the exponentially large set of possible stimuli, usually one samples that set using a distribution that favours “interesting” stimuli, see [WGR05] for more information. In this work, we consider a simplified version of the problem where the DUT is modeled by an unknown deterministic function F (blackbox) mapping length n binary string (the stimulus) into one of m possible colors (behaviour). Our goal is to obtain at least one representative input for each of the m possible output values of F , while using a reasonably small number of queries to the function F . When each possible color is quite frequent, the problem can be easily

¹ Evaluation queries are membership queries for the binary, $m = 2$, case.

solved by uniform random sampling. The more interesting cases are when some of the colors are rare. An extreme example for that case is the function F where a single point is colored red, and the rest are colored blue. In this case, any algorithm that finds the red point must use $\Omega(2^n)$ color queries. Therefore, if one desires to learn F using a significantly smaller number of queries, one must impose restrictions on the concept class from which F is drawn. In this paper we propose a parametrized restriction which we hope to be rich enough to contain some real hardware verification problems.

In the spirit of parametrized complexity study of various intractable computational problems, such as coloring, vertex cover, and satisfiability (see [DF99]) – we propose a parametrized family of concept classes. Depending on the parameter p , the problem transitions from efficiently learnable concepts ($p = O(1)$), to hopeless cases ($p = \Omega(n)$, which require $\Omega(2^n)$ queries).

The problem of obtaining one example from each color, **representative discovery**, is closely related to the more difficult problem of obtaining a full description of the partition, **partition discovery**. In some cases, there is a substantial difference between the two problems. For example, if F is the partition of the hypercube into two cubes, then the representative discovery problem can be solved without performing any query, as any two antipodes must have different colors. On the other hand, since any partition discovery algorithm has n possible outputs and gains at most one bit of information from each query, it cannot discover the partition using less than $\log_2 n$ queries.

In this work, we consider specific concept classes (families of partitions), and show that the two problems are virtually equivalent for these concept classes. Namely, the algorithms we describe solve the more difficult partition discovery problem, while being almost optimal with respect to the easier representative discovery problem.

We first consider the concept class, denoted by $\mathcal{P}$, consisting of the partitions in which each color class is a sub-cube. This concept class was previously considered by Fine and Mansour [FM06]. We give a partition discovery algorithm that uses at most $m(2 + \log n)$ expected number of queries. This result significantly improves upon the mn upper bound of [FM06] and we prove its optimality up to a constant factor, even as a representative discovery algorithm.

Next, we consider a generalization, $\mathcal{P}_p$, of the concept class $\mathcal{P}$ that allows each color class to be the (not necessarily disjoint) union of at most p sub-cubes. We give a partition discovery algorithm for $\mathcal{P}_p$ using at most $O(mp^2 2^p \log n)$ expected number of queries, and we show an almost matching lower bound of $\Omega(m2^p \log n)$, again for the easier problem of representative discovery.

The running time of each algorithm comprises of two parts — the time needed for the oracle to answer the queries, and the time required for choosing the queries. For our algorithms to be practical, we cannot ignore the second part. We show a bound of $O(m^2 n \log n)$ on the running time for the concept class $\mathcal{P}$. Unfortunately, we do not know how to provide an efficient implementation for the above algorithm for $\mathcal{P}_p$. We give a different, more practical, algorithm for the concept class $\mathcal{P}_p$ at the expense of allowing an arbitrarily small probability of error ϵ^2 , and using more queries. The expected running time of this algorithm is bounded by $\tilde{O}(m^2 n^2 2^{2p} \log(1/\epsilon))$, while its expected

² Note that with probability $1 - \epsilon$ the algorithm must produce the exact partition, which is very different from models such as PAC that allow the output to be an approximation.

number of queries is $\tilde{O}(mn4^p \log(1/\epsilon))$ where the tilde denotes the suppression of a polylogarithmic factor in $m, n, 2^p$.

These results prove that the problem is fixed parameter tractable, i.e. polynomial in all input parameters for a fixed value of p , where the degree of the polynomial is a constant independent of p . This shows that for any constant p , the problem is polynomial in both expected running time and the expected number of queries³.

The problems of learning a partition of the n -cube into m p -cubes, is closely related to the problem of learning decision trees and to the problem of learning a DNF. Both are well known problems for their theoretical and practical value. For work on decision tree learning, see [SL91], [KM91], [Mur98], [Kal07], and for work on DNF learning see [BR92], [KV94], [Bsh95], [BGHM96], [Kus97], [Jac94], [Bsh97], [BBB⁺00], [HR05], [Fel07]. We differentiate the algorithmic setups of the above algorithms as follows:

1. Exact learning vs. Approximate (PAC) learning, see [Val84].
2. Passive learning, where the labeled samples are drawn from a fixed distribution, or active learning, where the algorithm can choose its color queries.
3. Equivalence and membership queries vs. color evaluation queries only.

The algorithmic setup required by the hardware verification domain is that of exact learning via active color queries only. The exact learning requirement follows from the fact that we must hit all colors including the rare ones. The active learning via color queries only, follows from the black-box nature of our hardware model for the function F . Note that while the active part makes the problem easier (so ignoring it make the problem hopeless), the exclusion of equivalence queries makes it much harder.

Another difference between DNF learning and hypercube partition is the fact that even though each p -cube is a p -term DNF formula, our problem is not the same as learning m independent p -term DNF formulas since we demand that all colors partition the n -cube. Therefore, our problem is that of trying to simultaneously learn m disjoint DNF formulas.

All of the above mentioned algorithms fail to meet the real life requirements of our setup. For example Jackson's algorithm for learning DNF [Jac94] is a PAC learning algorithm; Bshouty's algorithm [Bsh95] uses equivalence queries.

2 Learning Partitions to Cubes

Suppose the n dimensional cube is partitioned into m sub-cubes C_1 through C_m . For any point $x \in \{0, 1\}^n$ let $c(x)$ denotes its color, which is the unique i satisfying $x \in C_i$. We seek a (possibly randomized) algorithm that uses a small (expected) number of color queries to determine the m sub-cubes.

Before stating the algorithm we introduce some notation. The projection along the j -th coordinate is denoted π_j , and in general π_J for projection on a set of coordinates J .

³ As mentioned above, we expect both the running time, and the required number of samples to deteriorate quite rapidly with p (certainly faster than polynomial). Therefore although not optimal, one should not be daunted by 2^{2^p} since the practical cases may have a very small p and large m, n .

A sub-cube is a non-empty set $T \subseteq \{0, 1\}^n$ that can be written as the Cartesian product $\pi_1(T) \times \cdots \times \pi_n(T)$. The support of T , denoted $\text{supp}(T)$, is the set of coordinates j with $|\pi_j(T)| = 2$, so $\dim(T) = |\text{supp}(T)|$. It is sometimes convenient to represent the cube T by a string $\alpha_T \in \{0, 1, *\}^n$, where $\alpha_j = *$ if $j \in \text{supp}(T)$. Otherwise, α_j is the single element in $\pi_j(T)$. Another way to look at a sub-cube is as a **monomial** $\prod_{\alpha_j=1} x_i \cdot \prod_{\alpha_j=0} \bar{x}_i$. The **convex hull** of a non-empty set $S \subset \{0, 1\}^n$, is the intersection of all the sub-cubes containing S . Equivalently, $\text{conv}(S) = \pi_1(S) \times \cdots \times \pi_n(S)$.

Consider the randomized algorithm in Figure 1.⁴ We claim that Algorithm A is both efficient in terms of the expected number of color queries and in terms of running time. Moreover, if m is not too large compared with n , the expected number of queries is best possible up to a constant factor.

Input: Integers n, m; a coloring oracle c Output: $S_1, \dots, S_m \subset \{0, 1\}^n$ such that $\text{conv}(S_i) = C_i$ for all i 1. $S_1, \dots, S_m \leftarrow \emptyset$ 2. $X \leftarrow \{0, 1\}^n$ 3. While $X \neq \emptyset$ 4. Choose random $x \in X$ 5. $i \leftarrow c(x)$ 6. $S_i \leftarrow S_i \cup \{x\}$ 7. $X \leftarrow X \setminus \text{conv}(S_i)$

Fig. 1. Algorithm A

Theorem 1. *Algorithm A is a partition discovery algorithm for the concept class $\mathcal{P}$, using at most $m(2 + \log n)$ expected number of queries.*

Theorem 2. *Algorithm A can be efficiently implemented, so that its expected running time is $O(m^2 n \log n)$.*

Theorem 3. *Any partition discovery algorithm for the concept class $\mathcal{P}$, requires at least $\Omega(m \log n)$, as long as $2 \leq m \leq 2^{n/2}$. The same bound holds also for representative discovery, as long as $3 \leq m \leq 2^{n/2}$.*

Note that, as mentioned before, if $m = 2$, we cannot hope to get a non-trivial lower bound for the representative discovery problem, since any two antipodes have different colors.

Proof (Proof of Theorem 1). First observe that after any iteration of the algorithm, we have $X = \{0, 1\}^n \setminus \bigcup_{i=1}^m \text{conv}(S_i)$. Since all points in S_i are colored i , all points in $\text{conv}(S_i)$ must be colored i . Upon termination, $X = \emptyset$, so the union of $\text{conv}(S_i)$ is the entire cube. Therefore, the color of all points is known, proving the correctness of the algorithm. We now turn to bound the expected number of queries.

⁴ The input m is not really needed and is given for clarity of exposition.

Consider some color i . We measure the progress made by the algorithm in color i by $\dim(\text{conv}(S_i))$ from the first time color i was hit, where $\dim(\text{conv}(S_i)) = 0$, to its final value $\dim(C_i)$. Suppose that at some step, the algorithm sampled the point x of color i . Let S, S_x denote the value of $\text{conv}(S_i)$ before and after updating for x , and let C denote C_i . Note that we have $S \subset S_x \subseteq C$. We claim that the following inequality holds:

$$E_x[\dim(C) - \dim(S_x)] \leq (\dim(C) - \dim(S))/2,$$

where the distribution of x is uniform on the set $C \setminus S$. Consider some coordinate j in $\text{supp}(C) \setminus \text{supp}(S)$. Then j is in $\text{supp}(S_x)$ iff $x_j \neq s_j$, where x_j is the j -th coordinate of x , and s_j is the unique value in $\pi_j(S)$. By linearity of expectation it suffices to prove that $\Pr[x_j \neq s_j] \geq 1/2$ for all such j . Indeed,

$$\Pr[x_j \neq s_j] = \frac{|C|/2}{|C \setminus S|} \geq \frac{|C|/2}{|C|} = \frac{1}{2},$$

as claimed.

We conclude the proof by observing that the above inequality implies that after $k+1$ hits to color i ,

$$E[\dim(C_i) - \dim(S_i)] \leq \dim(C_i)/2^k \leq \frac{n}{2^k}.$$

The dimension difference is an integer and therefore

$$P(\dim(C_i) - \dim(S_i) = 0) \leq \frac{n}{2^k}.$$

Therefore the expected time to get $C_i = S_i$ is

$$\begin{aligned} E\left(\min_{\dim(C_i)=\dim(S_i)} \{i\}\right) &= \sum_{i=1}^{\infty} P(\dim(C_i) = \dim(S_i)) \\ &\leq \lfloor \log_2 n \rfloor \cdot 1 + \frac{n}{2^{\lfloor \log_2 n \rfloor}} \cdot \sum_{i=1}^{\infty} 2^{-i} \leq \lfloor \log_2 n \rfloor + 2. \end{aligned}$$

So the expected number of hits required to exhaust color i is at most $2 + \log n$. The required result follows by linearity of expectation.

Proof (Proof of Theorem 2). It suffices to prove that checking if X is empty and random sampling from X can be efficiently implemented. To solve both problems we observe that for any cube C we can efficiently compute the cardinality of $X \cap C = C \setminus \bigcup_{i=1}^m \text{conv}(S_i)$. Indeed, the disjointness of $\text{conv}(S_i)$ implies that $|X \cap C| = 2^{\dim(C)} - \sum_i 2^{\dim(C \cap \text{conv}(S_i))}$, where the sum ranges over i such that $C \cap S_i$ is non-empty. For such i , the dimension of $C \cap \text{conv}(S_i)$ is just the number of coordinates in $\text{supp}(C) \cap \text{supp}(S_i)$.

For $C = \{0, 1\}^n$ the above observation solves the problem of checking whether X is empty. As for random sampling from X , we use the basic paradigm that counting and random sampling is equivalent. Specifically, we perform the procedure *Sample* described in Figure 2. One should note that for $j > 1$ line 3 can be performed in


```

Procedure Sample( $X$ )
1.  $C \leftarrow \{0, 1\}^n$ 
2. For  $j = 1$  to  $n$ 
3.    $p = |X \cap C \cap \pi_j^{-1}(0)| / |X \cap C|$ 
4.   Choose  $b \in \{0, 1\}$  randomly
       with  $\Pr[b = 0] = p$ 
5.    $C \leftarrow C \cap \pi_j^{-1}(b)$ 
6. Return the unique point  $x$  in  $C$ 

```

Fig. 2. Algorithm A, Procedure Sample

$O(1)$ time, by keeping the sets $\text{supp}(C) \cap \text{supp}(S_i)$ from the previous iteration, and performing the update only for coordinate j . It follows that the time needed to produce a uniformly random point from X (or prove no such point exists) is $O(mn)$, which yields the required bound.

Proof (Proof of Theorem 3). If $m = 2$, any partition discovery algorithm requires at least $\log n$ queries since there are n possible partitions in $\mathcal{P}$, and each query gives the algorithm at most one bit of information. Therefore, from now on we can restrict our attention to the representative discovery problem for $3 \leq m \leq 2^{n/2}$. Without loss of generality, $m = 3 \cdot 2^l$ for some non-negative integer l .

Let A' be some representative discovery algorithm. Acting as a deterministic adversary, we want to answer the color queries of the algorithm consistently, while ensuring the algorithm requires many color queries.⁵

When queried on the point $x \in \{0, 1\}^n$, we determine its color $c(x)$ as follows. The trailing l bits of $c(x)$ are just the trailing l bits of x , which we denote by y . The value of the remaining two bits, which has three possible values, is determined by performing a table lookup. The table $\{00, 01, 10\}$ is fed with the two input bits x_j and x_k for distinct indices $j, k \in \{1, \dots, n - l + 1\}$ that are determined based on the past queries of A' . Let $x^{(1)}, x^{(2)}, \dots, x^{(t)} = x$ be the sequence of past queries made by A' to points whose trailing bits are y . Then j, k are determined by the procedure in Figure 3.

We claim that this procedure produces a valid coloring for which the algorithm must make many color queries to find a representative for each color.

To prove the validity of the coloring, we exhibit a partition matching all answers made to the algorithm. The partition is defined by first partitioning the n -cube into $2^l = m/3$ subcubes $\{C_y\}$ according to the trailing l bits, y . The partition is further refined by partitioning each subcube into three subcubes according to the output of the lookup table for the two coordinates j_y, k_y . It remains to show that, for each y , one can exhibit values for j_y, k_y that are consistent with all answers made by the adversary. This follows from the fact that the sets calculated by above procedure satisfy $S_i \supseteq S_{i+1}$, and that as long as $|S_t| > 2$, all indices in S_t are equivalent for the first t queries to points in C_y .

To see that many queries are needed, observe that as long as $|S_t| > 2$, the answer to the query $c(x)$ is $00y$ or $10y$. Therefore, since A' must hit also $01y$ in order determine

⁵ Proving the lower bound for any possible algorithm A' proves that it also holds for any randomized algorithm as well.

```

1.  $S_0 \leftarrow \{1, \dots, n - l + 1\}$ .
2. For  $i = 1, 2, \dots, t$ 
3.    $S_{i,z} \leftarrow \{j \in S_{i-1} : x_j^{(i)} = z\}$  for  $z \in \{0, 1\}$ 
4.    $z_i \leftarrow \operatorname{argmax}_z \{|S_{i,z}|\}$ 
5.   If  $|S_{i,z_i}| \geq 2$  then
6.      $S_i \leftarrow S_{i,z_i}$ 
7.   Else
8.     Return  $\{j, k\} = S_{i-1}$ 
9. Return  $\{j, k\}$  as any two elements of  $S_t$ .

```

Fig. 3. The adversary's procedure

the partition of C_y , it must ask sufficiently many queries in C_y to ensure that $|S_t| = 2$. Since $|S_{i+1}| \geq |S_i|/2$ for all i , this requires at least $\log(n - l)$ queries to points in C_y . Therefore, discovering the partition requires at least $(m/3) \cdot \log(n - l)$, which is $\Omega(m \log n)$ as claimed.

3 Learning Partitions to p -Cubes

A subset of the cube is a p -cube if it can be expressed as the union of at most p cubes, not necessarily disjoint. We generalize the discussion of the preceding section on the $p = 1$ case to arbitrary integers $p \geq 1$. We denote the concept class of partitions into p -cubes by $\mathcal{P}_p$, and ask the following question:

Give an efficient partition/representative discovery algorithm with respect to $\mathcal{P}_p$.

We start by generalizing the definition of conv :⁶

Definition 1

$$\operatorname{conv}_p(S) = \bigcap_{\substack{S_1, \dots, S_p \\ \text{partition of } S}} \bigcup_{i=1}^p \operatorname{conv}(S_i).$$

Consider Algorithm A_p , obtained from Algorithm A by replacing conv with conv_p . Then:

Theorem 4. *Algorithm A_p discovers any partition from the concept class $\mathcal{P}_p$ within at most $O(mp^2 2^p \log n)$ expected number of queries.*

As for the case $p = 1$, discussed in the previous section, algorithm A_p is almost tight with respect to the number of color queries.

⁶ In the definition of conv_p we consider only proper partitions of S where no S_i is empty. It is easy to verify that allowing partitions with empty S_i results in the same definition, where by definition or notation, $\operatorname{conv}(\emptyset) = \emptyset$.

Theorem 5. Any representative discovery algorithm for $\mathcal{P}_p$, where $2 \leq m \leq 2^{n/2}$ and $p > 1$, requires at least $\Omega(m2^p \log n)$ color queries.

Before proving the theorems, we give three lemmas that shed some light on the behavior of conv_p . Note that while the computation of conv can be done efficiently, we are not aware of a method for the efficient computation of conv_p , even for the case $p = 2$. This problem shall be dealt with in Section 4.

Lemma 1. If the union of p cubes $C_1, \dots, C_p \subset \{0, 1\}^n$ is not the entire n -cube, then there is set J of at most p coordinates such that $|\pi_J(\cup_{i=1}^p C_i)| < 2^{|J|}$.

Corollary 1. For any cubes $C, C_1, \dots, C_p \subset \{0, 1\}^n$, one of the following two statements hold: (1) $C = \cup_{i=1}^p C_i$, (2) $|\cup_{i=1}^p C_i|/|C| \leq 1 - 2^{-p}$.

Proof (Proof of Lemma 1). We assume that $p \leq n$ since otherwise the statement is trivial. The existence of the required J is proved by induction on p . The claim holds for $p = 1$ since a cube C_1 that is not the entire n -cube must have a coordinate j with $|\pi_j(C_1)| < 2$.

Suppose that for some $p > 1$, the union of cubes $X = \cup_{i=1}^p C_i$ is not the entire n -cube. Then there must be a coordinate j with $|\pi_j(C_1)| < 2$. Without loss of generality $j = 1$ and $\pi_1(C_1) = \{0\}$. Let H denote the half cube $\pi_1^{-1}(1)$, then $C_1 \cap H = \emptyset$. Let I be the set of i such that $C_i \cap H$ is non empty. We distinguish two cases:

If $I = \emptyset$, then the set $J = \{1\}$ satisfies the requirements of the lemma.

Otherwise, $I \neq \emptyset$. By induction on the cubes $C_i \cap H$ for $i \in I$, there is a coordinate set $J' \subset \{2, \dots, n\}$ of size at most $|I| \leq p - 1$ such that $|\pi_{J'}(X \cap H)| < 2^{|J'|}$. The set $J = \{1\} \cup J'$ satisfies the requirements of lemma since J has the right cardinality and since the projection $\pi_J(X \cap H)$ does not cover all of $\pi_J(H)$.

Lemma 2

- For any two subsets S, T of the n -dimensional cube, $\text{conv}_p(S \cup T) \supseteq \text{conv}_p(S) \cup \text{conv}_p(T)$.
- For any subset S and point $x \in \text{conv}_p(S)$, we have $\text{conv}_p(S \cup \{x\}) = \text{conv}_p(S)$

Proof. Easy inspection.

Proof (Proof of Theorem 4). As in the analysis of Algorithm A, by linearity of expectation it suffices to prove an upper bound on the expected number of points sampled from color i by $O(p^2 2^p \log n)$. Let the p -cube C_i be the union of the cubes $\cup_{j=1}^{l_i} C_{ij}$, where $l_i \leq p$. By Lemma 2a it suffices to bound the expected number of points in $S_i \cap C_{ij}$ needed to achieve $\text{conv}_p(S_i \cap C_{ij}) = C_{ij}$ for all j . The claimed bound would follow by proving that the expected size of $S_i \cap C_{ij}$ is at most $O(p2^p \log n)$.

From now on, we would like to consider the situation from the view point of an arbitrary subcube C_{ij} . However, the subtle issue of sampling must be addressed first. Suppose that at some time in the run of algorithm A_p , the current value of S_i is S'_i . Then given that the sampled point x is in C_{ij} , its distribution is uniform on $C_{ij} \setminus \text{conv}_p(S'_i)$. However, since by Lemma 2b, adding a point $x \in \text{conv}_p(S'_i)$ to S'_i does not increase $\text{conv}_p(S'_i)$, we conclude that:

The expected size of $S_i \cap C_{ij}$ at the end of the algorithm is upper bounded by the expected number of uniform independent points one needs to sample from C_{ij} so that their conv_p is equal to C_{ij} .

By Lemma 1, the probability that a set S of k uniform independent random points from some d dimensional cube C doesn't satisfy $\text{conv}_p(S) = C$ can be upper bounded by:

$$\begin{aligned} \sum_{|J| \leq p} \sum_{y \in \{0,1\}^{|J|}} \Pr[y \notin \pi_J(S)] &\leq d^p 2^p (1 - 2^{-p})^k \\ &\leq d^p 2^p e^{-k/2^p}. \end{aligned}$$

Therefore, the expected number of random points one needs to sample from C until their conv_p equals C is bounded by $O(p2^p(1 + \log d))$. Therefore, since $d \leq n$, the expected number of points one needs for each of the p cubes of a color, and for each of the m colors is at most $O(p2^p \log n)$. This yields the required bound of the theorem.

Proof (Proof of Theorem 5). The proof has a similar flavor to the proof of Theorem 3. Let A'_p be some representative discovery algorithm for $\mathcal{P}_p$. Let n, m, p be three integers satisfying the requirements of the Theorem. Then, acting as an adversary we want to answer the color queries of the algorithm consistently, while ensuring the algorithm requires many color queries. Without loss of generality, it suffices to prove when $m = 2^l$ for some $l \geq 1$.

We build the partition in two stages. First, we partition the n -cube into 2^{l-1} subcubes according to the trailing $l - 1$ bits, $\{C_y : y \in \{0, 1\}^{l-1}\}$. Then, for each y , we choose a set J_y of p coordinates from $\{1, \dots, n - l\}$, and a length p bit string α_y . We color the cube C_y by two colors so that $x \in C_y$ is colored $y0$ if $\pi_{J_y}(x) = \alpha_y$, and is colored $y1$ otherwise. We claim that the we can adjust the parameters J_y and α_y according to the answers of the algorithm, so that A'_p will require many queries.

Our response to a color query $c(x)$ is computed as follows: Let Q be the set of past queries made by A'_p to points in C_y , including the last query to x . Then, as long as we have some $J \subset \{1, \dots, n - l\}$ of size p and $\alpha \in \{0, 1\}^p$ satisfying $\alpha \notin \pi_J(Q)$, we answer $y0$. This is obviously in agreement with the above partition for any such (J, α) . The first time when all (J, α) pairs have been eliminated, we set (J_y, α_y) to be one of the last (J, α) pair to survive. All subsequent queries to the C_y cube are answered by the above partition. By definition, this algorithm yields a valid coloring.

It remains to see that A'_p needs many queries to the C_y cube until it hits the color $y1$. Indeed, this follows from a theorem by Kleitman and Spencer [KS73], stating that the minimal number of points in C_y covering all such (J, α) pairs is $\Omega(2^p \log(n - l + 1))$. Therefore, since there are $2^{l-1} = m/2$ possible values for y , we obtain that the total number of queries A'_p needs is $\Omega(2^p m \log n)$, as claimed.

4 Efficient Learning Partitions into p -Cubes

Although Algorithm A_p uses an almost optimal number of queries, its running time may be exponential in n as it disregards the computational complexity of computing conv_p ,

which we do not know how to do faster than $\Omega(2^n)$. This not only makes Algorithm A_p inefficient, but also render its output less useful. In this section we propose Algorithm B, which is a computational efficient partition discovery algorithm for the concept class $\mathcal{P}_p$. However, the computational efficiency of Algorithm B is bought at the expense of a somewhat degraded query complexity, and at the expense of allowing the algorithm to err with an arbitrarily small probability.

Theorem 6. *Given some $\epsilon > 0$, Algorithm B described in Figure 4, is a partition discovery algorithm for $\mathcal{P}_p$, with error probability at most ϵ . Let $k = \lceil 2^p \log(m2^p n / \epsilon) \rceil$. Then its expected running time is $O(mn2^{2p} \lceil m2^{2p} + k \rceil)$, while its expected number of queries is $O(kmn2^p)$.*

Algorithm B is described in detail by the pseudo code of Figure 4 and subsequent figures. The algorithm uses several basic ideas to cover the n -cube by monochromatic cubes.

- The main loop, lines 2–8 in Figure 4 Finds a maximal monochromatic subcube C containing x , as long as there exists an uncovered point x , and add C to the cover.
- Lines 3–7 in Figure 4 find a maximal monochromatic subcube C containing x . This is done by starting with $C = \{x\}$, and scanning the coordinates from 1 to n . For each coordinate i , if the cube D obtained by turning the i -th coordinate of C into a star ($*$) is still monochromatic, replace C by D . The maximal cube, is the cube C after the completion of the loop.
- Line 6 in Figure 4 checks if a cube D is monochromatic. This task is performed (Figure 6) by sampling sufficiently many random points in D and verifying that all have the same color.
- Line 2 in Figure 4 finds a point x that is uncovered by the cubes found so far. The implementation (Figure 5) uses a similar paradigm to the one used in procedure *Sample* in Figure 2. Namely, that it suffices to know how to calculate the size of the covered part within a cube D . However, calculating this size is more complicated here since cubes covering the same color may intersect. This problem is resolved by applying the inclusion-exclusion formula.

Some implementation and notational notes:

- Throughout, it is convenient to represent a subcube by a string in $\{0, 1, *\}^n$, in the obvious manner.
- In line 5 of Figure 4, $C \oplus e_i$ denotes the exclusive or of C with the i -th unit vector. The resulting D is calculated from C by changing coordinate i into a star.
- Line 9 of *FindUncovered* does not require any actual computation, since the set $\mathcal{C}$ can be represented as m sets $\mathcal{C}_1, \dots, \mathcal{C}_m$ in the first place.
- Line 11 of *FindUncovered* can be implemented to run in time $O(1)$. This follows by observing that for $i > 1$, the intersection $D \cap \bigcap_{C \in \mathcal{C}_j} C$ is already known from the previous iteration, and that the required update consists of a local update for coordinate i .

```

Input: Integers  $n, m, p$ 
      Real  $\epsilon > 0$ 
      A coloring oracle  $c$ 
Output: A cover  $\mathcal{C}$  of  $\{0, 1\}^n$ 
       by monochromatic cubes
1.  $\mathcal{C} \leftarrow \emptyset$ 
2. While FindUncovered( $\mathcal{C}, 0, \{0, 1\}^n$ )
   finds a point  $x$ 
3.    $C \leftarrow \{x\}$ 
4.   For  $i = 1$  to  $n$ 
5.      $D \leftarrow C \cup (C \oplus e_i)$ 
6.     If Mono( $D, 2^p \log(m2^p n / \epsilon)$ ,  $x$ ) then
7.        $C \leftarrow D$ 
8.   Add  $C$  to  $\mathcal{C}$ 
9. Return  $\mathcal{C}$ 

```

Fig. 4. Algorithm B

The proof of Theorem 6 is immediate corollary of the following sequence of claims:

Lemma 3. *The main while loop will discover at most 2^p cubes of each color. Consequently, it will perform at most $m2^p$ iterations.*

Lemma 4. *It suffices to scan the coordinates once in order to find a maximal monochromatic cube containing x in lines 4-7 of algorithm B.*

Lemma 5. *Procedure Mono errs with probability at most $(1 - 2^{-p})^k$ in one invocation.*

Lemma 6. *The probability that procedure Mono makes an error throughout the execution of algorithm B is at most ϵ .*

Lemma 7. *Assuming that Mono did not err through the run of the algorithm, Procedure FindUncovered finds an uncovered point x iff there is such a point. Its total run time is $O(nm2^{2p})$.*

We prove above claims in order.

Proof (Proof of Lemma 3). The lemma immediately follows from Theorem 1.3 of Chandra and Markowsky [CM78]: Every k -term DNF has at most 2^k prime implicants.⁷ (Prime implicant is the same as maximal monochromatic cube in our terminology).

Proof (Proof of Lemma 4). Once we discover that coordinate i cannot be added to C since $C \cup (C \oplus e_i)$ is not monochromatic, then coordinate i cannot be added to any C' containing C . Therefore, it suffices to check each coordinate i once in any order. In particular in ascending order 1 through n .

⁷ The Theorem also states that there are examples where $3^{k/3}$ prime implicants are required for covering the DNF. Therefore, one cannot hope for a sub-exponential bound in the Lemma.

```

Procedure FindUncovered( $\mathcal{C}, i, D$ )
1. If  $i = n + 1$  then
2.   Let  $x$  be the single point in  $D$ 
3.   Return ``Found  $x$ ``
4. Else
5.   For  $b = 0$  to  $1$ 
6.      $D' \leftarrow D \cap \pi_i^{-1}(b)$ 
7.     sum  $\leftarrow 0$ 
8.     For  $j = 1$  to  $m$ 
9.        $\mathcal{C}_j \leftarrow$  the color  $j$  cubes of  $\mathcal{C}$ 
10.      For  $\mathcal{C}'_j \subseteq \mathcal{C}_j$ 
11.        sum  $\leftarrow$  sum  $- (-1)^{|\mathcal{C}'_j|} |D' \cap \bigcap_{C \in \mathcal{C}'_j} C|$ 
12.      If sum  $< |D'|$  then
13.        Return FindUncovered( $\mathcal{C}, i + 1, D'$ )
14. Return ``Not found``

```

Fig. 5. Algorithm B, Procedure FindUncovered

```

Procedure Mono( $D, k, x$ )
1. For  $i = 1$  to  $k$ 
2.    $y \leftarrow$  a random point in  $D$ 
3.   If  $c(y) \neq c(x)$  then
4.     Return false
5. Return true

```

Fig. 6. Algorithm B, Procedure Mono

Proof (Proof of Lemma 5). By definition, Procedure Mono does not err if its input cube D is monochromatic. So assume that D is not monochromatic. Since the points of D that are colored $c(x)$ can be represented as the union of at most p cubes, by Corollary 1, the probability that a random point y in D has the same color as x is bounded by $1 - 2^{-p}$. Therefore, the probability that all the k queries yield the same color $c(x)$ is bounded by $(1 - 2^{-p})^k$.

Proof (Proof of Lemma 6). For our choice of k in line 6 of algorithm B, the probability that a single call to Mono errs is at most $\epsilon / (mn2^p)$. Since by Lemma 3 the number of calls to the procedure Mono is at most $mn2^p$, the overall probability or error for Mono is at most ϵ , as claimed.

Proof (Proof of Lemma 7). Consider some call to Procedure FindUncovered during the run of Algorithm B. Let us denote $X_j = \bigcup_{C \in \mathcal{C}_j} C$, and $X = \bigcup_{C \in \mathcal{C}} C$. By assumption, Procedure Mono did not err before calling FindUncovered, so that X_j contains only points of color j , implying that the sets X_j are disjoint. Next, we claim that in line 12, sum = $|D' \cap X|$. Since by disjointness, $|D' \cap X| = \sum_{j=1}^m |D' \cap X_j|$, it suffices to prove that each iteration on j increments sum by $|D' \cap X_j|$, which follows from the

fact that lines 10–11 are a straight forward implementation of the inclusion exclusion formula.

It follows that the effect of lines 5–14 is to perform a recursive call with $D \cap \pi_i^{-1}(b)$ for the smallest value of b such that $D \cap \pi_i^{-1}(b) \setminus X$ is not empty, or return “Not Found” if not such value exist, which means that $D \setminus X$ is empty.

Therefore, if for the call to FindUncovered on line 2 of Algorithm B, no uncovered point x exists, “Not Found” is returned without performing any recursive call. Otherwise, $D \setminus X$ is not empty. By induction on i , all the calls to FindUncovered satisfy $\dim(D) = n + 1 - i$, and that $D \setminus X$ is not empty. In particular, for $i = n + 1$ the cube D consists of a single point x not covered by X .

5 Some Questions and Notes

- Our problem originally arose from the hardware verification domain. Can the algorithms proposed in this work be used for practical applications?
- A learner that can uniformly sample points from a cube $C \subseteq \{0, 1\}^n$, can efficiently learn C by checking its marginal distributions on the n coordinates. Namely, by checking which of the coordinates are constant (0 or 1), and which are not (equal probability for 0 and 1). Learning a p -cube D from its marginal distributions is a more complex task. In fact, there are examples, such as the 2-cube $\{00**\} \cup \{11**\}$, where D cannot be determined by its marginals. Can one exploit information on the marginal distributions to produce a better algorithm, if not in general, then at least in practice.
- In the inclusion-exclusion in FindUncovered procedure of Algorithm B (line 10 in Figure 5) takes 2^{2^p} time. Can one improve this bound or suggest a different more efficient procedure?
- It is natural to consider using decision trees learning algorithms to solve our cube partition problem and vice versa. However, one should be aware that there are examples where one representation is clearly better than the other:
 1. Using our algorithm for learning general decision trees whose leaves are colored by m colors, is not very attractive unless one can given a good bound on p . The obvious bound is the maximal number of leaves colored by the same color.
 2. When using a decision tree algorithm for learning an m -partition of the $\{0, 1\}^n$ into p -cubes, one should take into account the fact that the number of leaves in the resulting tree may be huge. An example for such a function F can be easily obtained from the iterated nand function considered in [JRSW97]. While F is representable by a partition with $m = 2^{O(\sqrt{n})}$, $p = 1$, the smallest decision tree describing F has an exponential size, $2^{\Omega(n)}$.
- Algorithm A is randomized. Can one devise an efficient deterministic algorithm for the same problem?
- Our lower bound (Theorem 5) directly applies only for error-free algorithms. We believe that it should be easy to generalise it but as it is, it does not immediately applies to algorithms that are allowed to err like Algorithm B.

Acknowledgments

We would like to thank Shai Fine for introducing the problem; to Ariel Birnbaum for numerous illuminating discussions; and to the anonymous referees who pointed us to [CM78], [JRSW97] and other useful remarks.

References

- [BBB⁺00] Beimel, A., Bergadano, F., Bshouty, N.H., Kushilevitz, E., Varricchio, S.: Learning functions represented as multiplicity automata. *J. ACM* 47(3), 506–530 (2000)
- [BGHM96] Bshouty, N.H., Goldman, S.A., Hancock, T.R., Matar, S.: Asking questions to minimize errors. *J. Comput. Syst. Sci.* 52(2), 268–286 (1996)
- [BR92] Blum, A., Rudich, S.: Fast learning of k -term dnf formulas with queries. In: *STOC 1992: Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pp. 382–389. ACM, New York (1992)
- [Bsh95] Bshouty, N.H.: Exact learning boolean functions via the monotone theory. *Inf. Comput.* 123(1), 146–153 (1995)
- [Bsh97] Bshouty, N.H.: Simple learning algorithms using divide and conquer. *Comput. Complex* 6(2), 174–194 (1997)
- [CM78] Chandra, A.K., Markowsky, G.: On the number of prime implicants. *Discrete Mathematics* 24(1), 7–11 (1978)
- [DF99] Downey, R.G., Fellows, M.R.: *Parameterized Complexity*. Springer, Heidelberg (1999)
- [Fel07] Feldman, V.: Attribute-efficient and non-adaptive learning of parities and dnf expressions. *J. Mach. Learn. Res.* 8, 1431–1460 (2007)
- [FM06] Fine, S., Mansour, Y.: Active sampling for multiple output identification. In: Lugosi, G., Simon, H.U. (eds.) *COLT 2006. LNCS (LNAI)*, vol. 4005, pp. 620–634. Springer, Heidelberg (2006)
- [HR05] Hellerstein, L., Raghavan, V.: Exact learning of dnf formulas using dnf hypotheses. *J. Comput. Syst. Sci.* 70(4), 435–470 (2005)
- [Jac94] Jackson, J.C.: An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. In: *IEEE Symposium on Foundations of Computer Science*, pp. 42–53 (1994)
- [JRSW97] Jukna, S., Razborov, A., Savicky, P., Wegener, I.: On p versus $NP \cap co-NP$ for decision trees and read-once branching programs. *Electronic Colloquium on Computational Complexity (ECCC)* 4(023) (1997)
- [Kal07] Kalai, A.T.: Learning nested halfspaces and uphill decision trees. In: Bshouty, N.H., Gentile, C. (eds.) *COLT. LNCS (LNAI)*, vol. 4539, pp. 378–392. Springer, Heidelberg (2007)
- [KM91] Kushilevitz, E., Mansour, Y.: Learning decision trees using the fourier spectrum. In: *STOC 1991: Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pp. 455–464. ACM, New York (1991)
- [KS73] Kleitman, D., Spencer, J.: Families of k -independent sets. *Discrete Math.* 6, 255–262 (1973)
- [Kus97] Kushilevitz, E.: A simple algorithm for learning $o(\log n)$ -term dnf. *Inf. Process. Lett.* 61(6), 289–292 (1997)
- [KV94] Kearns, M.J., Vazirani, U.V.: *An Introduction to Computational Learning Theory*. MIT Press, Cambridge (1994)

- [Mur98] Murthy, S.K.: Automatic construction of decision trees from data: A multi-disciplinary survey. *Data Mining and Knowledge Discovery* 2(4), 345–389 (1998)
- [SL91] Safavin, S.R., Langrebe, D.: A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man and Cybernetics* 21, 660–674 (1991)
- [Val84] Valiant, L.G.: A theory of the learnable. *Commun. ACM* 27(11), 1134–1142 (1984)
- [WGR05] Wile, B., Goss, J., Roesner, W.: *Comprehensive Functional Verification: The Complete Industry Cycle (Systems on Silicon)*. Morgan Kaufmann Publishers Inc., San Francisco (2005)

Iterative Learning of Simple External Contextual Languages

Leonor Becerra-Bonache^{1,*}, John Case², Sanjay Jain^{3,**},
and Frank Stephan^{4,**}

¹ Department of Computer Science, Yale University, New Haven,
CT 06520-8285, USA

`leonor.becerra-bonache@yale.edu`

² Department of Computer and Information Sciences, University of Delaware,
Newark, DE 19716-2586, USA

`case@cis.udel.edu`

³ Department of Computer Science, National University of Singapore,
Singapore 117590, Republic of Singapore

`sanjay@comp.nus.edu.sg`

⁴ Department of Computer Science and Department of Mathematics, National
University of Singapore, Singapore 117543, Republic of Singapore

`fstephan@comp.nus.edu.sg`

Abstract. It is investigated for which choice of a parameter q , denoting the number of contexts, the class of simple external contextual languages is iteratively learnable. On one hand, the class admits, for all values of q , polynomial time learnability provided an adequate choice of the hypothesis space is given. On the other hand, additional constraints like consistency and conservativeness or the use of a one-one hypothesis space changes the picture — iterative learning limits the long term memory of the learner to the current hypothesis and these constraints further hinder storage of information via padding of this hypothesis. It is shown that if $q > 3$, then simple external contextual languages are not iteratively learnable using a class preserving one-one hypothesis space, while for $q = 1$ it is iteratively learnable, even in polynomial time. For the intermediate levels, there is some indication that iterative learnability using a class preserving one-one hypothesis space might depend on the size of the alphabet. It is also investigated for which choice of the parameters, the simple external contextual languages can be learnt by a consistent and conservative iterative learner.

1 Introduction

Consider the following model of learning a class of languages: a learner M is shown any listing of the strings of a language L in the class and M outputs

* Supported by a Marie Curie International Fellowship within the 6th European Community Framework Programme.

** Supported in part by NUS grant number R252-000-212-112 and R252-000-308-112.

a sequence of hypotheses as it sees successively more and more strings of L . M , eventually, stops changing its mind and the final output is a grammar that correctly generates L . This model of learning is called “explanatory learning” as the final grammar can be seen as an “explanation for the language to be learnt”. Explanatory learning from positive data and its variants are frequently used to model important scenarios such as language acquisition of children [25].

Within the scenario of natural language acquisition, the formal study of this phenomenon requires answering the following question: Which kind of formal languages is adequate to model natural languages? This question has been a subject of debate for a long time. This debate started soon after the publication of [8] and it was focused on determining whether natural languages are context-free (CF) or not. Nevertheless, in the late 80’s, linguists seemed to finally agree that natural languages are not CF; there were discovered in many natural languages convincing examples of non-CF constructions [5, 28], such as so-called *multiple agreements*, *crossed agreements* and *duplication structures*. Besides, these works suggested that more generative capacity than CF is necessary to describe natural languages.

The difficulty of working more generally with context-sensitive languages has forced researchers to find other ways to generate CF and non-CF constructions, but keeping under control the generative power. This idea has led to the notion of *mildly context-sensitive* (MCS) family of languages, introduced by Joshi [12].

In the literature, different definitions of MCS have been presented. In this paper, by a *mildly context-sensitive* family of languages we mean a family $\mathcal{L}$ of languages that satisfies the following conditions:

- (1) each language in $\mathcal{L}$ is semilinear [21];
- (2) for each language in $\mathcal{L}$ the membership problem is solvable in deterministic polynomial time;
- (3) $\mathcal{L}$ contains the following non-context-free languages:
 - *multiple agreements*: $L_1 = \{a^n b^n c^n \mid n \geq 0\}$;
 - *crossed agreements*: $L_2 = \{a^n b^m c^n d^m \mid n, m \geq 0\}$;
 - *duplication*: $L_3 = \{ww \mid w \in \{a, b\}^*\}$.

Some authors [13, 26, 30] consider that such a family contains all CF languages and present mechanisms that fabricate mildly context-sensitive families which fully cover the CF but not the CS level of the Chomsky Hierarchy. However, taking into account the linguistic motivation of the concept of MCS, the following question arises: is it necessary that such a formalism generates all CF languages? As some authors [2, 3, 14, 18] pointed out, natural languages could occupy an orthogonal position in the Chomsky Hierarchy. In fact, we can find some examples of natural language constructions that are neither REG nor CF and also some REG or CF constructions that do not appear naturally in sentences. Therefore, it is justified to give up the requirement of generating all CF languages and we strive for formalisms which generate MCS languages in the above sense and occupy an orthogonal position in the Chomsky Hierarchy (furthermore, Gold [11] showed that any language class containing an infinite language and all its finite sublanguages (such as CF) is not explanatory learnable from positive data).

One example of a mechanism with these desirable linguistic properties is the *Simple External Contextual grammars* ($\text{SEC}_{p,q}$ grammars, where p, q are parameters discussed below). Note that, on the one hand, the corresponding class $\text{SEC}_{p,q}$ is, for $p, q > 1$, a mildly context-sensitive class, so the context-sensitive structures that led to the non-context-freeness of natural languages (multiple agreement, crossed agreement and duplication) can be covered by such grammars [2]. On the other hand, such classes $\text{SEC}_{p,q}$ are incomparable with the families REG and CF, but included in CS [2] (that is, they occupy an orthogonal position in the Chomsky Hierarchy). So, due to their properties, the $\text{SEC}_{p,q}$'s may be appropriate candidates to model natural language syntax.¹ Moreover, the $\text{SEC}_{p,q}$ grammar mechanism is (technically) quite simple and intuitively could be explained as follows: In the sentence “Anton learns English” one could add more objects and obtain “Anton learns English, French, German and Spanish”. Similarly the sentence “Gerd goes to France, Spain and Holland on Thursday, Saturday and Sunday, respectively” can be extended by expanding the list of countries and corresponding days, but for each new country also a new day has to be added. So, the idea is to start with an easy base sentence and then adding new parts at several places in a consistent manner. One can think of the parameter p as the number of positions in a base where additions can be inserted and the parameter q as the number of various contexts which can be inserted.

Becerra-Bonache and Yokomori [3] made the first attempt to learn these $\text{SEC}_{p,q}$ grammars from only positive data; they show that for each choice of parameters $p, q \geq 1$, $\text{SEC}_{p,q}$ is explanatorily learnable from positive data. They employ Shinohara's results [29]. However, the learning algorithm derived from their main result was not time-efficient. In [23], efficient learning of $\text{SEC}_{p,q}$ for some small values of the parameters p, q is considered.

The $\text{SEC}_{p,q}$ classes have their roots in contextual grammars [19], which were introduced with linguistic motivations (to model some natural aspects, like for example the acceptance of a word only in certain contexts). For an overview on contextual grammars the reader is referred to [22]. Fernau and Holzer [9] investigated learnability of classes of external contextual languages different from those of the present paper.

Human memory for past data seen seems to have limitations [17, 25, 31]. The present paper is about a nicely *memory-limited* form of explanatory learning from positive data called *iterative learning*. Each output grammar/hypothesis of an *iterative learner* depends only on the just prior, if any, hypothesis it output and on the string currently seen from an input listing of a language.

Our main positive results (Theorems 6 and 16 below in Sections 3 and 6, respectively) actually feature polynomial time learnability. This roughly means that the update time of the associated learner M is polynomial in the size of

¹ We do not claim that an SEC grammar is the best model for the syntax of natural languages or that we can describe all the constructions in natural languages by means of these grammars. But we believe that $\text{SEC}_{p,q}$ grammars have relevant properties from a linguistic point of view and are an attractive approximate model for natural language syntax that deserves further investigation.

its previous hypothesis and the latest datum. Here the size of the hypotheses themselves is bounded by a polynomial in the size of all the input data seen so far (thus the learner is *fair* and runs in time polynomial in all the data seen so far). In the prior literature on polynomial time explanatory learning (e.g., [16, 24, 33]), there are a number of suggestions on how to rule out unfair delaying tricks such as waiting for a long datum to have enough time to output a hypothesis actually based on a much shorter earlier datum. Fortunately, iterative learning (as in the present paper) is one of the best ways to rule out such delaying tricks. Intuitively, this is because the learner M does not have available for employment its whole history of past data. Theorem 6 says that, for each $p, q \geq 1$, the class $\text{SEC}_{p,q}$ is iteratively learnable in polynomial time using a class-preserving hypothesis space.

Of course, an iterative M can pad up its conjectured hypotheses to store a limited amount of past data seen. For example, some dummy information not affecting the semantics of a hypothesis can be added to it to, in effect, code in some bounded information about past data. In fact the proof of the positive result Theorem 6, depends on such a padding trick. It is thus interesting to see if such a result can still hold if we outlaw padding tricks in some natural ways.

One way to outlaw padding is to require the hypothesis space to be one-one, that is, to require that there is exactly one hypothesis available per relevant language. Another main result (Theorem 11 below in Section 4) says that *for class-preserving one-one hypothesis spaces*, for $p \geq 1, q \geq 4$, $\text{SEC}_{p,q}$ is not iteratively learnable. By contrast, Theorem 16 provides, for each p , for a class preserving one-one hypothesis space, polynomial time iterative learnability of $\text{SEC}_{p,1}$.

For a *consistent learner*, every hypothesis conjectured by the learner must generate all the data seen to that point. A *conservative learner* revises its hypothesis only if a current datum is inconsistent with it. Iterative learners which are both consistent and conservative are restricted in how much padding they can use. Another main result is that, for $p \geq 1, q \geq 2$, $\text{SEC}_{p,q}$ is not learnable by consistent and conservative iterative learners using a class-preserving hypothesis space.

In the remainder of the present paper, for the values of q not covered in each of the just above two paragraphs, we provide some partial results.

2 Notation and Preliminaries

For any unexplained recursion theoretic notation, the reader is referred to the textbook of Rogers [27]. The symbol $\mathbb{N}$ denotes the set of natural numbers, $\{0, 1, 2, 3, \dots\}$. For S a finite, non-empty subset of $\mathbb{N}$, $\text{gcd}(S)$ denotes the greatest common divisor of the elements in S . Σ denotes a finite alphabet set. Subsets of Σ^* are referred to as languages. The symbols $\emptyset, \subseteq, \subset, \supseteq$ and $\supset$ denote empty set, subset, proper subset, superset and proper superset, respectively. The cardinality of a set S is denoted by $|S|$. Let $|x|$ denote the length of the string x , where we take, then, $x = x(0)x(1)\dots x(|x| - 1)$. For $n < |x|$ let $x[n]$ denote the string

formed from the first n characters of x . For i, j with $i \leq j < |x|$ let $x[i, j]$ denote the substring $x(i)x(i+1) \dots x(j)$; if $j < i$ or $i \geq |x|$ or $j \geq |x|$ then $x[i, j] = \epsilon$, the empty string. Furthermore $x \cdot y$ or just xy denotes the concatenation of the strings x and y .

We often use regular expressions to define languages: For example, $A + B$ denotes the union $A \cup B$, x denotes $\{x\}$, $A - x$ denotes the set $A - \{x\}$, $A \cdot B = \{x \cdot y : x \in A, y \in B\}$. For example, $aa(bb + cc)^* = \{a \cdot a \cdot x : x \in \{b \cdot b, c \cdot c\}^*\}$ and $a^3 \cdot a^* = \{a^n : n \geq 3\}$.

We now present concepts from language learning theory. Sets of the form $\{x : x < n\}$, for some n , are called initial segments of $\mathbb{N}$. A (*finite*) *sequence* σ is a mapping from an initial segment of $\mathbb{N}$ into $(\Sigma^* \cup \{\#\})$. The empty sequence is denoted by λ . The *content* of a sequence σ , denoted $\text{cnt}(\sigma)$, is the set of elements occurring in σ that are different from $\#$. The *length* of σ , denoted by $|\sigma|$, is the number of elements in σ . So, $|\lambda| = 0$. For $n \leq |\sigma|$, the initial sequence of σ of length n is denoted by $\sigma[n]$. So, $\sigma[0]$ is λ . Intuitively, $\#$'s represent pauses in the presentation of data. We let σ, τ and γ range over finite sequences.

There are two types of concatenation: $\diamond$ is the symbol for concatenation of sequences (including those consisting of one string only) while $\cdot$ denotes the concatenation of strings or sets of strings. So $a^3 \diamond a^5 \diamond a^8$ is the sequence a^3, a^5, a^8 ; while $a^3 \cdot a^5 \cdot a^8$ is the string a^{16} .

A *text* [11] for a language L is a mapping T from $\mathbb{N}$ into $(\Sigma^* \cup \{\#\})$ such that L is the set of all strings occurring in the range of T . $T(i)$ represents the $(i+1)$ -th element in the text. The *content* of a text T , denoted by $\text{cnt}(T)$, is the set of elements occurring in T that are different from $\#$; that is, the language which T is a text for. $T[n]$ denotes the finite initial sequence of T with length n .

Definition 1 (Case, Gold, Lange, Lynes, Wiehagen and Zeugmann [6, 11, 17, 32]). Let $H_0, H_1, H_2, \dots$ be the underlying hypothesis space.

(a) An *iterative learner* or, in this paper, just *learner*, is a total recursive mapping M from $(\mathbb{N} \cup \{?\}) \times (\Sigma^* \cup \{\#\})$ to $\mathbb{N} \cup \{?\}$. The definition of M is extended to finite sequences by

$$\forall n \forall x_0, x_1, x_2, \dots, x_n \in \Sigma^* \cup \{?\} [M(x_0 \diamond x_1 \diamond x_2 \diamond \dots \diamond x_n) = M(M(\dots M(M(M(?), x_0), x_1), x_2), \dots), x_n)]$$

and every expression $M(\sigma)$ for σ a finite sequence, refers to this extension.

(b) M learns a language L from text T iff there is an index e with $H_e = L$ and $M(T[n]) = e$ for almost all n .

(c) A class $\mathcal{L}$ of languages is *iteratively learnable* iff there is an iterative learner M such that M learns every language $L \in \mathcal{L}$ from every text T for L .²

² In (a) above in this definition, we required M to be *total* recursive instead of just partial recursive, where, for successful iterative learning of a language L , M would have to be defined on the initial segments on any text for L . It is a folk result that this makes a difference as to what can be iteratively learned (see [7] for a proof). Our positive results happen to hold for total iterative learners M and our negative results would also hold if we removed the totality restriction. It is for expository convenience that we consider herein, for our negative results, total iterative learners.

Intuitively, an iterative learner [17, 32] is a learner whose hypothesis depends only on its last conjecture and current input. That is, for $n \geq 0$, $M(T[n+1])$ can be computed algorithmically from $M(T[n])$ and $T(n)$. Here, note that $M(T[0]) = ?$.

Definition 2 (L. and M. Blum and Fulk [4, 10]). σ is said to be a *stabilizing sequence* for M on L iff (a) $\text{cntnt}(\sigma) \subseteq L$ and (b) $M(\sigma \diamond \tau) = M(\sigma)$ for all τ with $\text{cntnt}(\tau) \subseteq L$. Furthermore, σ is said to be a *locking sequence* for M on L iff both σ is a stabilizing sequence for M on L and $M(\sigma)$ is an index of L .

If M learns L , then every stabilizing sequence for M on L is a locking sequence for M on L . Furthermore, one can show that if M learns L , then for every σ such that $\text{cntnt}(\sigma) \subseteq L$, there exists a locking sequence for M on L which extends σ , see [4, 10].

Definition 3 (Angluin and L. and M. Blum [1, 4]). Let $H_0, H_1, H_2, \dots$ be the hypothesis space used by M .

(a) M is said to be *consistent* iff, for all texts T and all $n \in \mathbb{N}$, $\text{cntnt}(T[n]) \subseteq H_{M(T[n])}$. (b) M is said to be *conservative* iff, for all texts T and all $n \in \mathbb{N}$, $\text{cntnt}(T[n+1]) \subseteq H_{M(T[n])}$ implies $M(T[n+1]) = M(T[n])$.

Kudlek, Martín-Vide, Mateescu and Mitrana [14] introduced and studied a mechanism to fabricate MCS families of languages called p -dimensional external contextual grammars.

Let $p \geq 1$ be a fixed integer. A p -word is a p -tuple $(w_1, w_2, \dots, w_p)$ of strings. A p -context is a $2p$ -word. An $\text{SEC}_{p,q}$ language can be represented by an $\text{SEC}_{p,q}$ grammar defined as follows.

Definition 4 (Becerra-Bonache and Yokomori [3]). Fix Σ . A *simple external contextual grammar with parameters p and q* (an $\text{SEC}_{p,q}$ grammar) is a pair $G = (\text{base}, C)$, where base is a p -word over Σ , and C is a set of p -contexts of cardinality at most q .

Given a p -word $w = (w_1, w_2, \dots, w_p)$ and a p -context $u = (u_1, u_2, \dots, u_{2p-1}, u_{2p})$, $\text{gen}(w, u)$ is the p -word $(u_1 w_1 u_2, u_3 w_2 u_4, u_5 w_3 u_6, \dots, u_{2p-1} w_p u_{2p})$. We generalize the definition of gen to multiple contexts by saying $\text{gen}(w, C) = \{\text{gen}(w, u) : u \in C\}$.

Suppose that a p -word base and a set C of p -contexts are given. Then $\text{Lang}(\text{base}, C)$ is obtained by considering the smallest set S satisfying the following two conditions:

- $\text{base} \in S$;
- If p -word $w \in S$ and p -context $u \in C$, then $\text{gen}(w, u) \in S$.

By Kleene's Minimal Fixed-Point Theorem [27], such a set S uniquely exists and is recursively enumerable. Now

$$\text{Lang}(\text{base}, C) = \{w_1 w_2 \dots w_p : (w_1, w_2, \dots, w_p) \in S\}.$$

We refer to (base, C) as grammar for $\text{Lang}(\text{base}, C)$ and let $\mathcal{G}_{p,q} = \{(\text{base}, C) : \text{base} \text{ is a } p\text{-word and } C \text{ is a set of at most } q \text{ } p\text{-contexts}\}$. Let $\text{SEC}_{p,q} =$

$\{Lang(base, C) : base \text{ is a } p\text{-word and } C \text{ is a set of at most } q \text{ } p\text{-contexts}\}$. Furthermore, let $SEC_{p,*} = \bigcup_{j \in \{1,2,\dots\}} SEC_{p,j}$ and $SEC_{*,q} = \bigcup_{j \in \{1,2,\dots\}} SEC_{j,q}$.

For example, $\{a^n b^n c^n \mid n > 0\}$ is generated by the following $SEC_{2,1}$ grammar: $G_{2,1} = (base = (\lambda, \lambda), context_1 = (a, b, c, \lambda))$.

We define the *size* of various objects of importance. The size of a string w is the length of the string w . The size of a p -word is the sum of p and the sizes of the strings in it. The size of a context set C is the sum of the cardinality of C and the sizes of the contexts in it. The size of a grammar G is the size of its base plus the size of its context set. The size of a finite sequence $x_0 \diamond x_1 \diamond x_2 \diamond \dots \diamond x_n$ is the sum of $n + 1$ and the size of all strings $x_0, x_1, x_2, \dots, x_n$ in this finite sequence.

For us, an iterative learner M runs in polynomial time iff both its update-function $M(e, x)$ runs in time polynomial in the size of e and x and the size of $M(\sigma)$ is polynomially bounded in the size of σ for every finite sequence σ .

We say a learner which learns a class $\mathcal{L}$ is *class-preserving* iff its underlying hypothesis space does not generate any languages outside $\mathcal{L}$ [15].

3 $SEC_{p,q}$ Is Consistently Iteratively Learnable

Kudlek, Martín-Vide, Mateescu and Mitrana [14] noted that the membership question for languages in $SEC_{p,q}$ is decidable in polynomial time.

Lemma 5. *Fix p, q and let $(base, C)$ be a member of $\mathcal{G}_{p,q}$. Given a string x , it is decidable in polynomial time (in size of $x, base, C$) whether $x \in Lang(base, C)$. The degree of the polynomial is linear in p and independent of q .*

Theorem 6. *For each $p, q \in \{1, 2, 3, \dots\}$, $SEC_{p,q}$ has a polynomial time iterative consistent learner M which uses a class-preserving hypothesis space. The runtime of M (measured in terms of the size of the previous hypothesis and current input-datum) and the size of M 's conjecture (measured in terms of the size of all input data seen so far) are bounded by a polynomial of degree linear in pq .*

Proof. A pair of base/context set $(base, C)$ is said to *minimally generate* a set Z iff both $Z \subseteq Lang(base, C)$ and for no $C' \subset C$, $Z \subseteq Lang(base, C')$.

For a fixed p, q , let X_Z denote the set of elements of $\mathcal{G}_{p,q}$ which minimally generate Z . For any string x , one can determine $X_{\{x\}}$, by considering all possible $(base, C)$, such that

- $base = (w_1, w_2, \dots, w_p)$ where each w_i is a substring of x ,
- for each member $(u_1, u_2, \dots, u_{2p-1}, u_{2p})$ of C , each u_i is a substring of x ,
- C contains at most q contexts,
- $(base, C)$ minimally generates $\{x\}$.

Note that this can be done in polynomial time in length of x , as the number of possible substrings of x is at most $(|x| + 1)^2$, and thus the number of possible grammars $(base, C)$ is at most $((|x| + 1)^2)^{p+2pq}$ (note that if the number of contexts is less than q , one could consider the rest of the contexts as “empty”).

Furthermore, for a nonempty set Z , given X_Z and $X_{\{x\}}$, note that $X_{Z \cup \{x\}}$ consists of grammars $(base, C) \in \mathcal{G}_{p,q}$ such that

- for some C', C'' satisfying $C = C' \cup C''$ it holds that $(base, C') \in X_Z$ and $(base, C'') \in X_{\{x\}}$,
- no $C''' \subset C$ satisfies the property above (for the given $base$ with C''' in place of C).

Thus, one can determine $X_{Z \cup \{x\}}$ from X_Z and x in time polynomial in $|x|$ and size of X_Z . Furthermore, size of X_Z is polynomial in size of Z (as each string in the base/contexts is a substring of one of the strings in Z and there are at most q contexts in each $(base, C) \in X_Z$). Thus, one can compute $X_{Z \cup \{x\}}$ in time polynomial in size of $Z \cup \{x\}$, given X_Z and x .

Now consider an arbitrary text T . Then we claim that $\lim_{n \rightarrow \infty} X_{\text{ctnt}(T[n])}$ converges. This can be seen as follows. One may assume without loss of generality that T does not contain any $\#$ (as input $\#$ does not lead to modification of $X_{\text{ctnt}(T[n])}$). Now consider a forest formed as follows. F_1 consists of $|X_{\text{ctnt}(T[1])}|$ roots corresponding to each member of $X_{\text{ctnt}(T[1])}$ (labeled using the corresponding member). By induction we will have that $X_{\text{ctnt}(T[n])}$ would be a subset of leaves of the forest F_n . F_{n+1} is constructed by possibly adding some children to leaves of F_n as follows. If $(base, C) \in X_{\text{ctnt}(T[n+1])} - X_{\text{ctnt}(T[n])}$, then pick a $(base, C') \in X_{\text{ctnt}(T[n])}$ such that $C' \subset C$ (there exists such a $(base, C')$ by construction). Add $(base, C)$ as a child of $(base, C')$. As the depth of the forest is at most q and the number of roots and the branching factor at any node is finite, the sequence $F_1, F_2, \dots$ converges.

Now one considers iterative learning of $\text{SEC}_{p,q}$. Let $g(\cdot)$ be a 1–1 polynomial time computable and polynomial time invertible coding of all finite sets of grammars over $\mathcal{G}_{p,q}$. The learner uses a class-preserving hypothesis space H such that $H_{g(X)}$ is for a minimal language in $\{\text{Lang}(G) : G \in X\}$ (for $X = \emptyset$, we let $g(X)$ to be grammar for $\{\epsilon\}$). Note that such a class-preserving hypothesis space H can be constructed by letting, for $X \neq \emptyset$, $x \in H_{g(X)}$ iff $x \in \text{Lang}(base, C)$ for all $(base, C) \in X$ such that $(\forall y \text{ length lexicographically smaller than } x)[y \in \text{Lang}(base, C) \Leftrightarrow y \in H_{g(X)}]$. In the construction of $H_{g(X)}$, minimal language instead of intersection of languages is used to have a class-preserving hypothesis space rather than class-comprising one (a *class-comprising* hypothesis space can also have hypotheses for languages not in the class of languages under consideration [15]).

The learner, on input $T[n]$, outputs the hypothesis $g(X_{\text{ctnt}(T[n])})$, if $\text{ctnt}(T[n]) \neq \emptyset$. Otherwise, the learner outputs $?$. Note that $X_{\text{ctnt}(T[n+1])}$ can be iteratively computed using $T(n)$ and $X_{\text{ctnt}(T[n])}$ (which can be obtained from $g(X_{\text{ctnt}(T[n])})$). Here note that, if the input language belongs to $\text{SEC}_{p,q}$, then (a) every grammar in $\lim_{n \rightarrow \infty} X_{\text{ctnt}(T[n])}$ contains the input language, (b) there is a grammar for the input language in $\lim_{n \rightarrow \infty} X_{\text{ctnt}(T[n])}$. Thus, for large enough n , $H_{g(X_{T[n]})}$ is the input language. $\square$

Corollary 7. *Suppose $\mathcal{G}$ is a recursive subset of $\mathcal{G}_{p,q}$ such that*

- *for all $C' \subseteq C$, $(base, C) \in \mathcal{G} \Rightarrow (base, C') \in \mathcal{G}$,*
- *for all p -words $base$, $(base, \emptyset) \in \mathcal{G}$,*
- *for all $G, G' \in \mathcal{G}$, one can effectively check whether $\text{Lang}(G) \subseteq \text{Lang}(G')$.*

Then,

- (a) $\mathcal{L} = \{Lang(G) : G \in \mathcal{G}\}$ is conservatively iteratively learnable using a class-preserving hypothesis space (this learner however may not be consistent);
 (b) for each $G \in \mathcal{G}$, one can effectively find a $D(G)$ such that, for all $G' \in \mathcal{G}$, if $D(G) \subseteq Lang(G')$, then $Lang(G) \subseteq Lang(G')$.

Proof. This follows using slight modification of the proof of Theorem 6 above. To define X_Z , as in Theorem 6 proof, we only use grammars from $\mathcal{G}$. Also, g is a coding for all finite sets of grammars from $\mathcal{G}$. The hypothesis space H' used by the learner is defined by using $H'_{2g(X_{\text{ctnt}(T[n])})}$ to be $H_{g(X_{\text{ctnt}(T[n])})}$, where H is as defined in proof of Theorem 6 (for the modified g). We let $H'_{1+2g(X_{\text{ctnt}(T[n])})}$ to contain just the shortest element generated by all the grammars in $X_{\text{ctnt}(T[n])}$. On input $T[n]$, the learner outputs $2g(X_{\text{ctnt}(T[n])})$ if $X_{\text{ctnt}(T[n])}$ contains a grammar G such that, for all $G' \in X_{\text{ctnt}(T[n])}$, $Lang(G) \subseteq Lang(G')$. Otherwise, the learner outputs $1 + 2g(X_{\text{ctnt}(T[n])})$.

The above learner is conservative: if the previous hypothesis output was $2g(X_{\text{ctnt}(T[n])})$ or $1 + 2g(X_{\text{ctnt}(T[n])})$, and the new input $T(n)$ belongs to the corresponding language, then $T(n)$ belongs to all $Lang(G)$, $G \in X_{\text{ctnt}(T[n])}$, and thus $X_{\text{ctnt}(T[n+1])} = X_{\text{ctnt}(T[n])}$.

Define $D(G)$ as follows. Consider a text T for $Lang(G)$. Then, one could iteratively construct $X_{\text{ctnt}(T[n])}$ until an n is found such that, for all $G' \in X_{\text{ctnt}(T[n])}$, $Lang(G) \subseteq Lang(G')$. Then $D(G) = \text{ctnt}(T[n])$ satisfies (b). $\square$

A tell-tale set [1] of a language $L \in \mathcal{L}$, is a finite set $S \subseteq L$ such that, for all $L' \in \mathcal{L}$, if $S \subseteq L'$, then $L' \not\subseteq L$. Note that $D(G)$ as defined in Corollary 7 is a tell-tale set.

Suppose $\mathcal{G} \subseteq \mathcal{G}_{p,q}$ satisfies the preconditions as in Corollary 7. Suppose further that, for all $G, G' \in \mathcal{G}$, $Lang(G) \not\subseteq Lang(G')$ implies there exists an x of length at most polynomial in the size of G, G' such that $x \in Lang(G) - Lang(G')$. Then the proof of Corollary 7 can be used to give a tell-tale set of polynomial size, as the branching factor of the forest formed in the proof of Theorem 6 would then be polynomially bounded in the size of G , when one considers an increasing text T for the input language.

For $|\Sigma| = 1$, $\text{SEC}_{p,q} = \text{SEC}_{1,q}$, as the order of words in the base/contexts does not matter. Furthermore, $Lang(G)$ is regular for each $G \in \mathcal{G}_{1,q}$ (where the automata for accepting $Lang(G)$ can be effectively obtained from G). Thus, for $|\Sigma| = 1$, $p, q \in \mathbb{N}$, $\text{SEC}_{p,q}$ is conservatively iteratively learnable. The following proposition generalizes this to $\text{SEC}_{p,*}$.

Proposition 8. Fix $\Sigma = \{a\}$. Then, $\text{SEC}_{1,*}$ is iteratively learnable using a class preserving hypothesis space. The learner can be made consistent or conservative (but not both simultaneously).

Proof. We first define $D(G)$, effectively obtainable from G , such that, for all $G' \in \mathcal{G}_{1,*}$, if $D(G) \subseteq Lang(G')$ then $Lang(G) \subseteq Lang(G')$. For ease of notation, we consider $G \in \mathcal{G}_{1,*}$ to be of the form (a^n, S) , with $Lang(G) = a^n S^*$, where S is a finite set of strings.

Consider $G = (a^n, S)$. If $S = \emptyset$, then $D(G) = \{a^n\}$. If $S = \{a^{i_1}, a^{i_2}, \dots, a^{i_r}\}$ is not empty, then let $m = \gcd(\{i_1, i_2, \dots, i_r\})$. Thus, $\text{Lang}(G)$ is a finite variant of $a^n(a^m)^*$. Let i be minimal such that $a^n a^{i*m}$ and $a^n a^{i*m+m} \in \text{Lang}(G)$. Now, for any set S' ,

- If $a^n \in a^s(S')^*$, then $s \leq n$;
- if $\{a^{n+im}, a^{n+im+m}\} \subseteq a^s(S')^*$, then $a^s\{a^{n+im-s}, a^{n+im+m-s}\}^* \subseteq a^s(S')^*$;
- for $s \leq n$, $\text{Lang}(G) - a^s\{a^{n+im-s}, a^{n+im+m-s}\}^*$ is finite and one can effectively (from s) find this set.

Let $D(G) = \{a^n, a^{n+im}, a^{n+im+m}\} \cup \bigcup_{s \leq n} [\text{Lang}(G) - a^s\{a^{n+im-s}, a^{n+im+m-s}\}^*]$. It follows that any language in $\text{SEC}_{1,q}$ containing $D(G)$ also contains $\text{Lang}(G)$.

Now we can use the methods of Theorem 6 and Corollary 7 to obtain iterative consistent or iterative conservative learner. Note that existence of $D(G)$ as above is enough to guarantee the convergence of $X_{\text{ctnt}(T[n])}$ for texts T for $\text{Lang}(G)$, as needed for learnability. $\square$

Theorem 9. *Suppose that $\{a, b\} \subseteq \Sigma$. Then $\text{SEC}_{1,*}$ is not explanatorily learnable.*

Proof. Let $L_i = \{ab^j : j \leq i\}^*$. Let $H = \{ax : x \in \{a, b\}^*\}$. Note that $L_0 \subset L_1 \subset L_2 \subset \dots \subset H$ and $H = \bigcup_{i \in \mathbb{N}} L_i$. Furthermore, $L_0, L_1, L_2, \dots, H \in \text{SEC}_{1,*}$. Thus, $\text{SEC}_{1,*}$ is not explanatorily learnable, by a result of Gold [11]. $\square$

4 Padding Is Necessary

Padding naturally needs that there are several hypotheses for at least some of the languages involved. Therefore it is natural to ask how learnability is affected in the case that there is only one grammar for each language in the class to be learnt. In such a situation, it is of course also needed to consider class-preserving hypothesis spaces as otherwise hypotheses for languages outside the class could be used to store information intermediately. For the following, since we are considering one-one hypothesis spaces, we often identify the language with its grammar.

Remark 10. Let a class-preserving one-one hypothesis space $H_0, H_1, H_2, \dots$ of some class and an iterative learner M for this class be given. Then M is conservative. One can even show the following more strict variant:

$$(\forall \sigma)(\forall x \in H_{M(\sigma)})[M(\sigma \diamond x) = M(\sigma)].$$

If this condition would fail for some σ and $x \in H_{M(\sigma)}$, the learner M would not learn $H_{M(\sigma)}$ from any text T containing infinitely many x . This holds as for every n with $T(n) = x$, either $M(T[n]) \neq M(\sigma)$ or $M(T[n+1]) \neq M(\sigma)$, although $M(\sigma)$ is the only index of that language.

Theorem 11. *Suppose that $q \in \{4, 5, 6, \dots, *\}$ and $p \in \{1, 2, 3, \dots, *\}$. Then $\text{SEC}_{p,q}$ is not iteratively learnable using a class preserving one-one hypothesis space.*

Proof. Suppose by way of contradiction that some iterative learner M learns $\text{SEC}_{1,q}$ using a class preserving one-one hypothesis space. Let H be the set described by the hypothesis $M(\sigma_1)$ where $\sigma_1 = a^4 \diamond a^5$ (if $M(\sigma_1) = ?$, then clearly M cannot distinguish between texts for $a^4 a^*$ and $a^6 a^*$). The set H is not empty and has thus a shortest element, let n be its length. Now consider the following cases where Case i is only taken if no Case j with $j < i$ applies.

Case 1: $H \not\subseteq a^*$. Let $x \in H - a^*$ and let T be a text for $a^* x^* - \{x\}$. This language is in $\text{SEC}_{1,4}$ as one can generate it with base ϵ and the four contexts (a, ϵ) , (a, x) , (ϵ, x^2) and (ϵ, x^3) . As M is conservative, $M(\sigma_1 \diamond x) = M(\sigma_1)$; so M converges on the text $\sigma_1 \diamond x \diamond T$ and the text $\sigma_1 \diamond T$ to the same hypotheses although these are texts for the different languages $a^* x^*$ and $a^* x^* - \{x\}$, respectively.

Case 2: $n < 4$. Let T be a text for $a^{n+1} \cdot a^*$. As M is conservative, it converges to the same hypothesis on the texts $\sigma_1 \diamond T$ and $\sigma_1 \diamond a^n \diamond T$, which are texts for different languages in $\text{SEC}_{1,4}$.

Case 3: $n > 4$. In this case let σ_2 be a stabilizing sequence for M on H . Let T be a text for $a^5 a^*$. Then, M on $\sigma_2 \diamond T$ converges to the same hypothesis as on $\sigma_1 \diamond T$, though they are texts for different languages in $\text{SEC}_{1,3}$.

Case 4: $n = 4$ and $a^5 \notin H$. In this case, let T be a text for $a^4(a^2 + a^3)^*$ and let σ_3 be locking sequence for M on H . Now the learner converges to the same grammar on $\sigma_3 \diamond T$ as on $\sigma_1 \diamond T$, though these are texts for $a^4(a^2 + a^3)^*$ and $a^4 a^*$ respectively.

Case 5: $H = a^4 a^*$. As M is conservative, $M(\sigma_1 \diamond a^6) = M(\sigma_1)$. Now let T be a text for $a(a^3 + a^4)^*$. The learner M converges on $\sigma_1 \diamond a^6 \diamond T$ and on $\sigma_1 \diamond T$ to the same grammar, though these are respectively the texts for the languages $a(a^3 + a^4 + a^5)^*$ and $a(a^3 + a^4)^*$.

Hence, in all five cases, the learner M fails to infer some language it should infer. It is easy to see that the case-distinction is exhaustive. So, the theorem follows. $\square$

Another method to hinder padding is to require that a learner is consistent and conservative. Consistency enforces that the learner has to incorporate new data in a reasonable way so that no padding can be done by choosing a bogus hypothesis, conservativeness rules out updating done for data-storage purposes only.

Proposition 12. *Suppose that $q \in \{2, 3, 4, \dots, *\}$. Then $\text{SEC}_{p,q}$ has no consistent and conservative iterative learner using a class-preserving hypothesis space.*

5 Learnability and the Unary Alphabet

The previous section leaves open whether $\text{SEC}_{p,1}$, $\text{SEC}_{p,2}$ and $\text{SEC}_{p,3}$ can be iteratively learnt using a class-preserving one-one hypothesis space. While this question will be answered positively for $\text{SEC}_{p,1}$ by Theorem 16 below, it remains open for $\text{SEC}_{p,2}$ and $\text{SEC}_{p,3}$. The main purpose of this section is to close this gap for the case that the alphabet has size 1 and hence the situation is easier to clarify. Note that the proof of Theorem 11 needs $q \geq 4$ only in Case 1.

In the other cases, the languages considered $(a^n a^*, a^4(a^2 + a^3)^*, a(a^3 + a^4)^*, a(a^3 + a^4 + a^5)^*)$ are all in $\text{SEC}_{1,3}$. Hence, one has for $\Sigma = \{a\}$ that $\text{SEC}_{p,3}$ is also not iteratively learnable using a class-preserving one-one hypothesis space.

Corollary 13. *Suppose that $|\Sigma| = 1$. Then $\text{SEC}_{p,3}$ is not iteratively learnable using a class preserving one-one hypothesis space.*

Theorem 14. *Suppose $\Sigma = \{a\}$. Then $\text{SEC}_{1,2}$ is not iteratively learnable using a class preserving one-one hypothesis space.*

Remark 15. Note that for $|\Sigma| = 1$, $\text{SEC}_{p,1} = \text{SEC}_{1,1}$ and $\text{SEC}_{p,1}$ has a consistent and conservative iterative learner which uses a class preserving one-one hypothesis space. If the input language is a singleton a^r , then the learner conjectures $\{a^r\}$. Otherwise, the learner conjectures $a^r(a^s)^*$, where a^r is the shortest string seen so far and $s = \gcd(\{i : i \neq 0, a^{r+i} \text{ is seen in the input so far}\})$. Note that, for nonempty S with $0 \notin S$, $\gcd(\{j\} \cup S) = \gcd(\{j, \gcd(S)\})$. Also, $\gcd(\{j\} \cup \{i + j : i \in S\}) = \gcd(\{j, \gcd(S)\})$. Thus, $s = \gcd(\{i : i \neq 0, a^{r+i} \text{ is seen in the input so far}\})$, can always be computed using the new datum and the previous hypothesis.

6 Classes with One Context Only

For arbitrary alphabet size, we do not yet know if $\text{SEC}_{p,1}$ can be consistently iteratively learnt using a class preserving one-one hypothesis space. However, we show in this section that one can do so if the consistency requirement is dropped.

In the theorem below, the degree of the polynomial bounding the runtime of M depends linearly on p . The size of the hypothesis (measured in terms of the size of all input data seen so far) is linear.

Theorem 16. *Suppose $p \in \{1, 2, 3, \dots, *\}$. Then $\text{SEC}_{p,1}$ is iteratively learnable in polynomial time using the hypothesis space $\mathcal{G}_{p,1}$. Moreover, one can even use a class preserving one-one hypothesis space in place of $\mathcal{G}_{p,1}$.*

7 A Special Case

In this section, a subclass of $\text{SEC}_{p,q}$ consisting of regular sets only will be considered which is defined as follows: $\mathcal{R}_{p,q} = \{x_1 \cdot Y_1^* \cdot x_2 \cdot Y_2^* \cdot \dots \cdot x_p \cdot Y_p^* : x_1, x_2, \dots, x_p \in \Sigma^*, Y_1, Y_2, \dots, Y_p \subseteq \Sigma^*, |Y_1| + |Y_2| + \dots + |Y_p| \leq q\}$.

This subclass is obtained by permitting only right contexts and by requiring that the strings in every context are different from ϵ at only one place. Note that $\mathcal{R}_{q+1,q} = \mathcal{R}_{*,q}$ and for the unary alphabet $\text{SEC}_{p,q} = \mathcal{R}_{1,q}$.

Remark 17. As one can construct, for each language in $\mathcal{R}_{p,q}$, a finite automaton accepting it, the inclusion-problem for these languages is decidable. Thus, by Corollary 7

- $\mathcal{R}_{p,q}$ is conservatively learnable and
- there is a recursive function which computes for every $L \in \mathcal{R}_{p,q}$ (given in adequate form) a finite subset $D(L)$ of L such that, for all $H \in \mathcal{R}_{p,q}$, $D(L) \subseteq H \Rightarrow L \subseteq H$.

Remark 18. For $q \in \{6, 7, 8, \dots, *\}$, one cannot iteratively learn $\mathcal{R}_{p,q}$ using a one-one class-preserving hypothesis space. This holds as the learner on input a^4, a^5 either produces a hypothesis $L \subseteq a^*$ which then can be diagonalized as shown in Theorem 11 or it produces a hypothesis L which contains some $x \notin a^*$. In the latter case, the learner cannot distinguish the language $\{a, x\}^*$ from the language $\{a, xx, xxx, ax, xa, xax\}^* = \{a, x\}^* - \{x\}$. We are not able to use $q = 4$ or $q = 5$ as in the proof of Theorem 11, as the language given there for this case is not in $\mathcal{R}_{5,4}$.

If $\Sigma = \{a\}$ then one cannot iteratively learn $\mathcal{R}_{p,2}$ using a class-preserving one-one hypothesis space by Theorem 14. Our next result shows that this is possible in the case that $|\Sigma| \geq 3$ and hence there is a proven dependence on the alphabet size.

Suppose $\Sigma \supseteq \{a, b, c\}$. Let $S_L = \bigcup_{r \geq \max((s_2)^3 + 3s_2, 1001)} a^r b^r c \{a, b\}^*$, where s_2 is the length of the second shortest string in L .

Theorem 19. *For any $L, L' \in \mathcal{R}_{3,2}$, if $L - S_L \subseteq L'$, then $S_L \cap L \subseteq L'$ (and thus, $L \subseteq L'$).*

Now one can conclude the following result on the learnability of $\mathcal{R}_{3,2}$ when the alphabet size is at least 3.

Theorem 20. *If $\{a, b, c\} \subseteq \Sigma$ then $\mathcal{R}_{3,2}$ can be iteratively learnt using a class preserving one-one hypothesis space.*

Proof. One can use the following iterative algorithm for learning $\mathcal{R}_{3,2}$ using a class preserving one-one hypothesis space. On input $\#$ do not change the hypothesis. If the input is a string w then proceed according to that of the following cases which is applicable.

1. If the previous hypothesis was $?$, then output a grammar for $\{w\}$.
2. If the previous hypothesis contains w , then repeat the hypothesis.
3. If the previous hypothesis H does not contain w and H is not of the form $a^h b^h c \{a, b\}^*$ for some $h > 0$, then let $S = D(H) \cup \{w\}$. Let k be a code for S . Let n be the length of the second smallest string in S . Output a hypothesis for $a^m b^m c \{a, b\}^*$, where $m = 2^k(2n^3 + 4n + 1001)$.
4. If the previous hypothesis H of the learner does not contain w and is of the form $a^m b^m c \{a, b\}^*$ for some $m > 0$, then let S' be the set coded by $k' = \max\{k'' : 2^{k''} \text{ divides } m\}$ and let $S = S' \cup \{w\}$.
 - If there is an $L \in \mathcal{R}_{3,2}$ such that for all $L' \in \mathcal{R}_{3,2}$ the implication $S \subseteq L' \Rightarrow L \subseteq L'$ holds, then output the hypothesis for L . Note that the above condition on S and L can be checked by just constructing X_S as in proof of Theorem 6, by checking if a grammar for L is in X_S and by checking whether each hypothesis in X_S , which belongs to $\mathcal{R}_{3,2}$, contains L .

- Otherwise, let k be code for finite set S and let n be the length of the second smallest string in S . Output a hypothesis for $a^m b^m c\{a, b\}^*$, where $m = 2^k(2n^3 + 4n + 1001)$.

Note that each set of the form $a^m b^m c\{a, b\}^*$, does not have a proper superset in $\mathcal{R}_{3,2}$. Thus, in step 4, when we see a data outside the hypothesis, we know that coding must have been used. Note furthermore that $2^k(2n^3 + 4n + 1001) > n^3 + 3n^2$ and $2^k(2n^3 + 4n + 1001) > 1000$. Thus the condition of Theorem 19 on the choice of m is satisfied. It follows easily from Theorem 19 that the above algorithm learns $\mathcal{R}_{3,2}$ using a class preserving one-one hypothesis space. $\square$

Acknowledgments. We thank Samuel E. Moelius III and the anonymous referees of ALT 2008 for useful comments.

References

- [1] Angluin, D.: Inductive inference of formal languages from positive data. *Information and Control* 45, 117–135 (1980)
- [2] Becerra-Bonache, L.: On the Learnability of Mildly Context-Sensitive Languages Using Positive Data and Correction Queries. PhD thesis, Rovira i Virgili University (2006)
- [3] Becerra-Bonache, L., Yokomori, T.: Learning mild context-sensitiveness: toward understanding children's language learning. In: Paliouras, G., Sakakibara, Y. (eds.) *ICGI 2004. LNCS (LNAI)*, vol. 3264, pp. 53–64. Springer, Heidelberg (2004)
- [4] Blum, L., Blum, M.: Toward a mathematical theory of inductive inference. *Information and Control* 28, 125–155 (1975)
- [5] Bresnan, J., Kaplan, R.M., Peters, S., Zaenen, A.: Cross-serial dependencies in Dutch. *Linguistic Inquiry* 13, 613–635 (1982)
- [6] Case, J., Lynes, C.: Inductive inference and language identification. In: *Ninth International Colloquium on Automata, Languages and Programming (ICALP 1982)*. LNCS, vol. 140, pp. 107–115. Springer, Heidelberg (1982)
- [7] Case, J., Moelius, S.E.: Parallelism increases iterative learning power. In: Hutter, M., Servadio, R.A., Takimoto, E. (eds.) *ALT 2007. LNCS (LNAI)*, vol. 4754, pp. 41–55. Springer, Heidelberg (2007)
- [8] Chomsky, N.: Three models for the description of language. *IRE Transactions on Information Theory* 3, 113–124 (1956)
- [9] Fernau, H., Holzer, M.: External contextual and conditional languages. In: *Recent Topics in Mathematical and Computational Linguistics, Papers in Honor of Solomon Marcus on the Occasion of his 75th Birthday*, pp. 104–120. Editura Academiei Române, Bucuresti (2000)
- [10] Fulk, M.: Prudence and other conditions on formal language learning. *Information and Computation* 85, 1–11 (1990)
- [11] Gold, E.M.: Language identification in the limit. *Information and Control* 10, 447–474 (1967)
- [12] Joshi, A.K.: Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions? In: *Natural Language Parsing*, pp. 206–250. Cambridge University Press, Cambridge (1985)
- [13] Joshi, A.K., Schabes, Y.: Tree-adjoining grammars. *Handbook of Formal Languages*, vol. 3, pp. 69–123. Springer, Heidelberg (1997)

- [14] Kudlek, M., Martín-Vide, C., Mateescu, A., Mitrana, V.: Contexts and the concept of mild context-sensitivity. *Linguistics and Philosophy* 26, 703–725 (2003)
- [15] Lange, S., Zeugmann, T.: Language learning in dependence on the space of hypotheses. In: *Proceedings of the Sixth Annual Conference on Computational Learning Theory, COLT 1993*, pp. 127–136. ACM Press, New York (1993)
- [16] Lange, S., Wiehagen, R.: Polynomial time inference of arbitrary pattern languages. *New Generation Computing* 8, 361–370 (1991)
- [17] Lange, S., Zeugmann, T.: Incremental learning from positive data. *Journal of Computer and System Sciences* 53, 88–103 (1996)
- [18] Manaster-Ramer, A.: Some uses and abuses of mathematics in linguistics. In: *Issues in Mathematical Linguistics*, pp. 73–130. John Benjamins, Amsterdam (1999)
- [19] Marcus, S.: Contextual grammars. *Revue Roumaine des Mathématiques Pures et Appliquées* 14, 1525–1534 (1969)
- [20] Marcus, S., Paun, G., Martín-Vide, C.: Contextual grammars as generative models of natural languages. *Computational Linguistics* 24(2), 245–274 (1998)
- [21] Parikh, R.J.: On context-free languages. *Journal of the ACM* 13, 570–581 (1966)
- [22] Paun, G.: *Marcus Contextual Grammar*. Kluwer Academic Publishers, Netherlands (1997)
- [23] Oates, T., Armstrong, T., Becerra-Bonache, L., Atamas, M.: Inferring grammars for mildly context-sensitive languages in polynomial-time. In: Sakakibara, Y., Kobayashi, S., Sato, K., Nishino, T., Tomita, E. (eds.) *ICGI 2006. LNCS (LNAI)*, vol. 4201, pp. 137–147. Springer, Heidelberg (2006)
- [24] Pitt, L.: Inductive inference, DFAs, and computational complexity. In: Jantke, K.P. (ed.) *AI 1989. LNCS*, vol. 397, pp. 18–44. Springer, Heidelberg (1989)
- [25] Osherson, D.N., Stob, M., Weinstein, S.: *Systems That Learn, An Introduction to Learning Theory for Cognitive and Computer Scientists*. MIT Press, Cambridge (1986)
- [26] Roach, K.: Formal properties of head grammars. In: Manaster-Ramer, A. (ed.) *Mathematics of Language*, pp. 293–348. John Benjamins, Amsterdam (1987)
- [27] Rogers, H.: *Theory of Recursive Functions and Effective Computability*. McGraw-Hill, New York (1967)
- [28] Shieber, S.M.: Evidence against the context-freeness of natural language. *Linguistics and Philosophy* 8, 333–343 (1985)
- [29] Shinohara, T.: Rich classes inferable from positive data: Length-bounded elementary formal systems. *Information and Computation* 108, 175–186 (1994)
- [30] Steedman, M.: Dependency and coordination in the grammar of Dutch and English. *Language* 61, 523–568 (1985)
- [31] Wexler, K., Culicover, P.W.: *Formal Principles of Language Acquisition*. MIT Press, Cambridge (1980)
- [32] Wiehagen, R.: Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. *Elektronische Informationsverarbeitung und Kybernetik* 12, 93–99 (1976)
- [33] Yoshinaka, R.: Learning efficiency of very simple grammars from positive data. In: Hutter, M., Servidio, R.A., Takimoto, E. (eds.) *ALT 2007. LNCS (LNAI)*, vol. 4754, pp. 227–241. Springer, Heidelberg (2007)

Topological Properties of Concept Spaces

Matthew de Brecht and Akihiro Yamamoto

Graduate School of Informatics, Kyoto University
Yoshida Honmachi, Sakyo-ku, Kyoto, Japan 606-8501
`matthew@iip.ist.i.kyoto-u.ac.jp`
`akihiro@i.kyoto-u.ac.jp`

Abstract. Based on the observation that the category of concept spaces with the positive information topology is equivalent to the category of countably based T_0 topological spaces, we investigate further connections between the learning in the limit model of inductive inference and topology. In particular, we show that the “texts” or “positive presentations” of concepts in inductive inference can be viewed as special cases of the “admissible representations” of computable analysis. We also show that several structural properties of concept spaces have well known topological equivalents. In addition to topological methods, we use algebraic closure operators to analyze the structure of concept spaces, and we show the connection between these two approaches. The goal of this paper is not only to introduce new perspectives to learning theorists, but also to present the field of inductive inference in a way more accessible to domain theorists and topologists.

1 Introduction

It is well known among learning theorists that there are many connections between topology and inductive inference [16, 21], and that some important notions in inductive inference can be easily formalized by introducing the positive information topology on to a concept space [18]. Here we take “concept space” to mean a subset of the powerset of natural numbers. What is less well known is that *every* countably based T_0 topological space is homeomorphic to some concept space. Therefore, it is important to clarify the connection between properties of concept spaces proposed by learning theorists and properties of topological spaces, which is one of the goals of this paper. For example, we show that the property that every concept has a finite tell-tale [1] is equivalent to a separation axiom that has been around since the early 1960’s [3], and also that “texts” or “positive presentations” are a special kind of the admissible representations [25] used in computable analysis [27].

In addition to topological methods, we also use algebraic closure operators to analyze structural properties of concept spaces. This should not be confused with the *topological* closure operator defined on the concept space, instead we are referring to an operator defined on the set of natural numbers that arises when one embeds the concept space into a minimal algebraic closed set system. This

approach has the benefit that it retains some information about the structure of a concept space that is lost when we take a purely topological view.

Another one of our goals is to explain some of the basic concepts of inductive inference in a way more accessible to domain theorists and topologists. It is our hope that this will promote a more fluid exchange of ideas between these fields.

We do not consider computability issues in this paper, and so the learners we consider will be non-computable in general. In the next section we introduce concept spaces and learning in the limit with an emphasis on the topological characteristics of these notions. We introduce algebraic closure operators in the third section. In the fourth section, we analyze some well known structural properties of concept spaces using topology and algebraic closure operators. We conclude in the fifth section.

2 Concept Spaces and Learning in the Limit

We assume the reader has a basic knowledge of topology [15]. In this paper, $X \subseteq Y$ will mean that X is a subset of Y , and $X \subset Y$ will mean that X is a proper subset of Y . We denote set difference by $X \setminus Y$. Given a function f from a set X to a set Y , X will be called the “domain” of f , and Y the “codomain” of f . The range of f will be denoted $\text{rng}(f)$. The image of a subset $S \subseteq X$ under f will be denoted $f(S)$. Given functions f and g such that the domain of g equals the codomain of f , their composition will be denoted $g \circ f$. The set of natural numbers will be denoted by $\mathbf{N}$. A *concept space* is a collection of subsets of $\mathbf{N}$. An element of a concept space is called a *concept*. Given a concept space $\mathcal{L}$ and any subset S of $\mathbf{N}$, we define $\uparrow_{\mathcal{L}} S = \{L \in \mathcal{L} \mid S \subseteq L\}$ and $\downarrow_{\mathcal{L}} S = \{L \in \mathcal{L} \mid L \subseteq S\}$.

Definition 1 (Luo and Schulte [18]). *Let $\mathcal{L}$ be a concept space. A subset of $\mathcal{L}$ is called a Π -basic open set if and only if it is equal to $\uparrow_{\mathcal{L}} F$ for some finite subset F of $\mathbf{N}$. An arbitrary union (including the empty union) of Π -basic open sets is called a Π -open set. Π -closed sets and Π -clopen sets are defined as usual. The resulting topology is called the positive information topology (Π -topology) on the concept space $\mathcal{L}$. A mapping between concept spaces that is continuous with respect to the Π -topologies is said to be Π -continuous. $\square$*

The following proposition should probably be attributed to Dana Scott.

Proposition 1. *Every countably based T_0 space is homeomorphic to some concept space with the Π -topology.*

Proof. Let X be a countably based T_0 space and let $\{\beta_i\}_{i \in I}$ be a countable base for X , where $I \subseteq \mathbf{N}$. For $x \in X$, define $\eta(x) = \{i \in I \mid x \in \beta_i\}$. Define $P(X) = \{\eta(x) \mid x \in X\}$. Then $P(X)$ is a concept space, and it is easy to see that $\eta: X \rightarrow P(X)$ is a homeomorphism. $\square$

Since every concept space is a countably based T_0 space, Proposition 1 implies that the category of concept spaces and Π -continuous maps is equivalent to the

category of countably based T_0 spaces and continuous maps (see Mac Lane [19] for more on category theory).

Let $\mathcal{T}(\mathcal{L}) = \{T \in (\mathbf{N} \cup \{\#\})^{\mathbf{N}} \mid \text{rng}(T) \setminus \{\#\} \in \mathcal{L}\}$. $\mathcal{T}(\mathcal{L})$ will be called the set of *texts* for concepts in $\mathcal{L}$. Here, $(\mathbf{N} \cup \{\#\})^{\mathbf{N}}$ is the set of all functions from $\mathbf{N}$ to $\mathbf{N} \cup \{\#\}$, and we can alternatively think of it as the set of all countably infinite sequences of elements of $\mathbf{N} \cup \{\#\}$. The element $\#$ is a special symbol not in $\mathbf{N}$ that is necessary for defining texts for empty concepts. For formal purposes, we will view $\mathcal{T}(\mathcal{L})$ as a concept space, where each $T \in \mathcal{T}(\mathcal{L})$ is the concept of all finite initial prefixes of T properly encoded as natural numbers.

Define the mapping $\tau_{\mathcal{L}}: \mathcal{T}(\mathcal{L}) \rightarrow \mathcal{L}$ so that $\tau_{\mathcal{L}}(T) = \text{rng}(T) \setminus \{\#\}$ for each $T \in \mathcal{T}(\mathcal{L})$. We will say that T is a text for $L \in \mathcal{L}$ if and only if $\tau_{\mathcal{L}}(T) = L$. Intuitively, T is a text for L if and only if it is an infinite enumeration of the elements of L , with occasional pauses denoted by $\#$. It is easily seen that $\tau_{\mathcal{L}}: \mathcal{T}(\mathcal{L}) \rightarrow \mathcal{L}$ is a Π -continuous quotient map.

Recall that a topological space is *zero-dimensional* if and only if it has a basis of clopen sets. Clearly $\mathcal{T}(\mathcal{L})$ is a zero-dimensional concept space. The mapping $\tau_{\mathcal{L}}$ has a kind of “universal” property as seen in the following theorem. The proof can be given by adapting Matthias Schröder’s proof of his Theorem 12 characterizing admissible representations of topological spaces [25].

Theorem 1. *Let $\mathcal{K}$ be a zero-dimensional concept space, and $f: \mathcal{K} \rightarrow \mathcal{L}$ be a Π -continuous map. Then there exists a Π -continuous map $g: \mathcal{K} \rightarrow \mathcal{T}(\mathcal{L})$ such that $f = \tau_{\mathcal{L}} \circ g$. $\square$*

Theorem 1 essentially means that the pair $\langle \mathcal{T}(\mathcal{L}), \tau_{\mathcal{L}} \rangle$ is an *admissible representation* for $\mathcal{L}$ (for more on admissible representations and their applications in computable analysis, see [25, 27]). In light of Theorem 1, admissible representations provide a notion of a “text” to a topological space, without having to first find a homeomorphic concept space (we should note that admissible representations are not unique to a space, although they can be continuously reduced to one another). We get a nice corollary that follows from Schröder’s Generalized Main Theorem¹ [25] and our Theorem 1.

Corollary 1. *A function $f: \mathcal{K} \rightarrow \mathcal{L}$ between concept spaces is Π -continuous if and only if there is a Π -continuous function $g: \mathcal{T}(\mathcal{K}) \rightarrow \mathcal{T}(\mathcal{L})$ such that $f \circ \tau_{\mathcal{K}} = \tau_{\mathcal{L}} \circ g$. $\square$*

The Π -continuous function g in the corollary can be thought of as an operator that gradually reads in a text for some $K \in \mathcal{K}$ and outputs a text for $f(K) \in \mathcal{L}$. Since a finite portion of the output of g depends only on a finite portion of the input, we can see that this is essentially the same as the enumeration operators used in analyzing the reducibility of one concept space to another (see [12] for the definition of strong and weak reductions). Luo and Schulte [18] were the first to notice that a strong reduction between concept spaces induces an injective

¹ Note that the sequential continuity mentioned in [25] is equivalent to the usual notion of continuity for countably based spaces.

continuous function between them, and we can easily see from the above corollary that the converse also holds if we allow non-effective learners and enumeration operators.

A *hypothesis space* for a concept space $\mathcal{L}$ is a pair $\langle \mathcal{H}, h \rangle$, where $\mathcal{H}$ is a discrete concept space (i.e. the Π -topology is the discrete topology) and $h: \mathcal{H} \rightarrow \mathcal{L}$ is a Π -continuous surjective function. Every function from a discrete space is continuous, but we include the Π -continuity requirement to allow the definition to be easily generalized to non-discrete hypothesis spaces. We will give an example of a non-discrete hypothesis space towards the end of this paper. We should also point out that every discrete concept space is countable, but since non-countable concept spaces would not be learnable with an uncountable discrete hypothesis space anyways (see [20]), this does not pose a problem for us.

Given a hypothesis space $\langle \mathcal{H}, h \rangle$ for $\mathcal{L}$, we let $\mathcal{H}^{\mathbf{N}}$ denote the *concept space* of countably infinite sequences of concepts in $\mathcal{H}$. If we view $\mathbf{N}$ as a concept space homeomorphic to the natural numbers with the discrete topology, then $\mathcal{H}^{\mathbf{N}}$ is an exponential object [19] in the category of concept spaces and Π -continuous functions. But for our purposes, we can just think of the concepts in $\mathcal{H}^{\mathbf{N}}$ as being the set of all finite prefixes of some infinite sequence of concepts of $\mathcal{H}$ (properly encoded as natural numbers). Thus the Π -topology on $\mathcal{H}^{\mathbf{N}}$ is similar to the Π -topology on $\mathcal{T}(\mathcal{L})$, and both can be thought of as subspaces of the Baire space [14]. We first give a topological definition of learning in the limit.

Definition 2. A learner for $\mathcal{L}$ with respect to the hypothesis space $\langle \mathcal{H}, h \rangle$ is a Π -continuous function $\psi: \mathcal{T}(\mathcal{L}) \rightarrow \mathcal{H}^{\mathbf{N}}$. We say that ψ learns $\mathcal{L}$ in the limit from positive data if and only if for every $T \in \mathcal{T}(\mathcal{L})$, the sequence $\psi(T)$ of elements of $\mathcal{H}$ converges to some $H \in \mathcal{H}$ such that $\tau_{\mathcal{L}}(T) = h(H)$. $\mathcal{L}$ is said to be learnable in the limit from positive data if and only if there exists a learner that learns $\mathcal{L}$ with respect to some hypothesis space. $\square$

The word “converges” in the above definition is to be taken in the topological sense [15]. We next give what we will call the “standard” definition of learning in the limit. Let SEQ be the set of all finite sequences of elements of $\mathbf{N} \cup \{\#\}$. Given a text T , we let $T[n]$ denote the initial segment of T of length n .

Definition 3 (Gold [10]). A learner for $\mathcal{L}$ with respect to the hypothesis space $\langle \mathcal{H}, h \rangle$ is a function $\psi: SEQ \rightarrow \mathcal{H} \cup \{?\}$, where $?$ is a special symbol not in $\mathcal{H}$. We say that ψ learns $\mathcal{L}$ in the limit from positive data if and only if for every $T \in \mathcal{T}(\mathcal{L})$, there is some $H \in \mathcal{H}$ and some $m \in \mathbf{N}$ such that $\psi(T[n]) = H$ for all $n \geq m$ and $\tau_{\mathcal{L}}(T) = h(H)$. $\mathcal{L}$ is said to be learnable in the limit from positive data if and only if there exists a learner that learns $\mathcal{L}$ with respect to some hypothesis space. $\square$

Note that we do not require ψ to be computable. Since we have required that hypothesis spaces be discrete, it is not difficult to see that both definitions give the same notion of learnability to concept spaces. Although the definitions of a learner are technically different, it is easy enough to convert one type of learner to the other type that we will not be overly concerned with the differences.

3 Algebraic Closure Operators

We next give a way of analyzing the structure of a concept space using algebraic closure operators. Several terms such as *closed set* and *compact* will clash with their topological meanings, but it should be clear from context what we mean, because the Π -topology is on a concept space (i.e., a set of subsets of $\mathbf{N}$), whereas the algebraic closure operators we discuss are defined on $\mathbf{N}$ itself. Thus if we talk about the closure of a set of natural numbers, it is clear that we are not referring to any Π -closed set.

Definition 4 (see [5]). *Let U be a set, and let $C: 2^U \rightarrow 2^U$ be a mapping on the powerset of U . C is called an algebraic closure operator on U if the following conditions hold for all subsets X and Y of U :*

1. $X \subseteq C(X)$.
2. $C(X) = C(C(X))$.
3. $X \subseteq Y$ implies $C(X) \subseteq C(Y)$.
4. $C(X) = \bigcup \{C(F) \mid F \text{ is a finite subset of } X\}$.

An operator that only fulfills the first three conditions above is simply called a closure operator. A closed set or a fixed point of C is a set X such that $X = C(X)$. A finitely generated closed set is a set X such that $X = C(F)$ for some finite subset F of U . $\square$

The set of all fixed points of an algebraic closure operator is called an *algebraic closure system*. An algebraic closure system $\mathcal{C}$ forms a complete lattice (ordered by subset inclusion) where:

$$\bigvee_{i \in I} C(X_i) = C\left(\bigcup_{i \in I} X_i\right) \quad \text{and} \quad \bigwedge_{i \in I} C(X_i) = \bigcap_{i \in I} C(X_i).$$

Given a poset P , a non-empty subset D of P is called a *directed set* if for all $x, y \in D$, there exists $z \in D$ such that $x \leq z$ and $y \leq z$.

Definition 5. *Let L be a lattice and let x be an element of L . We say that x is compact if and only if for every directed set $D \subseteq L$ such that $x \leq \bigvee D$, there exists an element $d \in D$ such that $x \leq d$.* $\square$

The compact elements of an algebraic closure system are precisely the finitely generated closed sets [5]. Compact elements are sometimes called *finite* or *finitary* elements. We now introduce the following closure operator on $\mathbf{N}$ with respect to a given concept space.

Definition 6. *Let $\mathcal{L}$ be a concept space. For any subset S of $\mathbf{N}$, let*

$$C_{\mathcal{L}}(S) = \bigcup \left\{ \bigcap \{L \in \mathcal{L} \mid F \subseteq L\} \mid F \text{ is a finite subset of } S \right\}.$$

Define $\mathcal{A}(\mathcal{L})$ to be the set of fixed points of $C_{\mathcal{L}}(\cdot)$. $\square$

It can be shown that $\mathcal{A}(\mathcal{L})$ is the smallest algebraic closure system that contains $\mathcal{L}$ (we should note that we follow the convention that $\bigcap \emptyset = \mathbf{N}$). Intuitively, given a set $S \subseteq \mathbf{N}$, $C_{\mathcal{L}}(S)$ is the largest subset of $\mathbf{N}$ that we can be sure is included in each concept of $\mathcal{L}$ that contains S , given that we can only inspect finite subsets of S at a given time. Thus, given two finite subsets F and G of some unknown concept $L \in \mathcal{L}$, we can think of F as containing “more information” about L than G if $C_{\mathcal{L}}(G) \subset C_{\mathcal{L}}(F)$.

The Π -topology on $\mathcal{A}(\mathcal{L})$ is precisely the Scott-topology on $\mathcal{A}(\mathcal{L})$ when viewed as a lattice (see [9] for more on the Scott-topology). It follows that a function $f: \mathcal{A}(\mathcal{K}) \rightarrow \mathcal{A}(\mathcal{L})$ is Π -continuous (equivalently, Scott-continuous) if and only if $f(\bigvee D) = \bigvee f(D)$ for every directed subset D of $\mathcal{A}(\mathcal{K})$.

Definition 7. Let $\mathcal{K}$ and $\mathcal{L}$ be concept spaces, and let $f: \mathcal{K} \rightarrow \mathcal{L}$ be a Π -continuous function. Define $\mathcal{A}(f): \mathcal{A}(\mathcal{K}) \rightarrow \mathcal{A}(\mathcal{L})$ so that for each $X \in \mathcal{A}(\mathcal{K})$, $\mathcal{A}(f)(X) = \bigcup \{ \bigcap \{ f(K) \mid F \subseteq K \in \mathcal{K} \} \mid F \text{ is a finite subset of } X \}$. $\square$

The following can be proven as a special case of proposition II-3.9 in [9].

Theorem 2. $\mathcal{A}(f): \mathcal{A}(\mathcal{K}) \rightarrow \mathcal{A}(\mathcal{L})$ is Scott-continuous and $\mathcal{A}(f)(L) = f(L)$ for all $L \in \mathcal{L}$. Furthermore, $\mathcal{A}(f)$ is the supremum (ordered pointwise, i.e., $g \leq h$ iff $g(X) \subseteq h(X)$ for all X) of all such Scott-continuous extensions of f . $\square$

Thus, given a text T for some $L \in \mathcal{L}$, we can let $D = \{X_0, X_1, X_2, \dots\}$ represent the ascending chain of the closed sets produced by applying $C_{\mathcal{L}}$ to the set of natural numbers appearing in each initial finite segment of T . Since $L = \bigvee_{i \in \mathbf{N}} X_i = \bigvee D$ and D is directed, in fact a chain, $f(L) = \mathcal{A}(f)(\bigvee D) = \bigvee \mathcal{A}(f)(D)$. Thus, we can produce a text for $f(L)$ by enumerating in parallel the elements of each $\mathcal{A}(f)(X_i)$, while we can obtain each X_i by seeing more and more of T . More abstractly, we can think of $\mathcal{A}(f): \mathcal{A}(\mathcal{K}) \rightarrow \mathcal{A}(\mathcal{L})$ as a mapping from partial information about some $K \in \mathcal{K}$ to partial information about $f(K) \in \mathcal{L}$. The fact that it is the supremum of all continuous extensions means that it is the “best” such mapping of partial information.

We mention to those familiar with category theory that, despite our notation, $\mathcal{A}$ is *not* a functor because although it preserves identities it does not preserve composition.

4 Structural Properties of Concept Spaces

We now introduce a handful of structural properties that have been introduced by learning theorists to characterize the learnability of concept spaces. We can only give a brief explanation of their importance, and the reader should consult the references for more details. In the following subsections, we will assume that $\mathcal{K}$ and $\mathcal{L}$ are arbitrary concept spaces.

4.1 Finite Thickness and Finite Elasticity

The first two structural properties of concept spaces that we will consider are *finite thickness* and *finite elasticity*. Both are sufficient conditions for a concept space to be learnable in the limit.

Definition 8 (Angluin [1]). We say that $\mathcal{L}$ has finite thickness if and only if for each $x \in \mathbf{N}$, $\uparrow_{\mathcal{L}} \{x\}$ is finite. $\square$

Finite thickness was introduced by Angluin, and is well known to be a property held by pattern languages [1].

Definition 9 (Wright [28], Motoki et al. [22]). We say that $\mathcal{L}$ has infinite elasticity if and only if there exists an infinite sequence of concepts $L_1, L_2, L_3, \dots$ in $\mathcal{L}$ and elements $x_0, x_1, x_2, \dots$ such that $\{x_0, \dots, x_{n-1}\} \subseteq L_n$ but $x_n \notin L_n$. $\mathcal{L}$ has finite elasticity if and only if it does not have infinite elasticity. $\square$

Finite elasticity is strictly weaker than finite thickness and was introduced by Wright [28] because it is preserved under “unions” of concept spaces (we use quotations and the symbol $\dot{\cup}$ to distinguish from the set theoretical definition of union):

Theorem 3 (Wright [28]). If $\mathcal{K}$ and $\mathcal{L}$ have finite elasticity, then their “union” $\mathcal{K} \dot{\cup} \mathcal{L} = \{K \cup L \mid K \in \mathcal{K} \text{ and } L \in \mathcal{L}\}$ has finite elasticity. $\square$

Neither finite thickness nor finite elasticity are topological properties in the sense that they are not *topologically invariant* [15]. A topologically invariant property is a property that if held by one space is held by every other homeomorphic space. As an example showing that finite thickness and finite elasticity are not topologically invariant, we let $\mathcal{SINGLE} = \{\{n\} \mid n \in \mathbf{N}\}$ and let $\mathcal{L}_1 = \{L_i \mid i \geq 0\} \cup \{J_i \mid i \geq 0\}$ where:

$$\begin{aligned} L_0 &= \{\langle n, 0 \rangle \mid n \in \mathbf{N}\} \cup \{\langle 0, 1 \rangle\}; & L_i &= \{\langle n, 0 \rangle \mid 0 \leq n \leq 2i \text{ \& } n \neq i\} \cup \{\langle i, 1 \rangle\} \\ J_0 &= \{\langle n, 1 \rangle \mid n \in \mathbf{N}\} \cup \{\langle 0, 0 \rangle\}; & J_i &= \{\langle n, 1 \rangle \mid 0 \leq n \leq 2i \text{ \& } n \neq i\} \cup \{\langle i, 0 \rangle\} \end{aligned}$$

It is easy to see that $\mathcal{SINGLE}$ has finite thickness. However, since $L_1 \cup J_1 \subset L_2 \cup J_2 \subset \dots$ and $L_0 \cup J_0 = \bigcup_{i \geq 1} L_i \cup J_i$, it can be proven that $\mathcal{L}_1 \dot{\cup} \mathcal{L}_1$ is not learnable from positive data. Therefore, $\mathcal{L}_1$ does not have finite elasticity. But it can easily be shown that $\mathcal{L}_1$ is homeomorphic to $\mathcal{SINGLE}$, because L_0 is the only concept that contains the subset $\{\langle 0, 0 \rangle, \langle 0, 1 \rangle, \langle 1, 0 \rangle\}$, J_0 is the only one that contains $\{\langle 0, 0 \rangle, \langle 0, 1 \rangle, \langle 1, 1 \rangle\}$ and all of the other concepts are finite and not contained in any other concepts.

Since we have shown that finite thickness and finite elasticity are not topologically invariant, we can not hope to have a nice topological characterization of these properties. However, there is a nice characterization of finite elasticity using algebraic closure operators.

Theorem 4. The following statements are equivalent for any concept space $\mathcal{L}$.

1. $\mathcal{L}$ has finite elasticity.
2. $\mathcal{A}(\mathcal{L})$ has finite elasticity.
3. $\mathcal{A}(\mathcal{L})$ is Noetherian (every strictly increasing chain of closed sets is finite).
4. Every closed set in $\mathcal{A}(\mathcal{L})$ is compact.
5. $\mathcal{A}(\mathcal{L})$ is learnable in the limit from positive data. $\square$

The main parts of the proofs can be found in [6] and [7]. The equivalence of Noetherian and learnable algebraic closure systems is a generalization of the same result in [26] for the class of ideals of a ring.

4.2 Scattered Concept Spaces

Scattered concept spaces are *defined* topologically, and therefore the property is easily seen to be topologically invariant. The notion was introduced to the learning community by Luo and Schulte [18] as a means of characterizing mind-change complexity. Mind-change complexity is a means of measuring the “difficulty” of a learning problem using ordinals. We will use the standard definition of a learner in the following definition, with the convention that for every learner ψ , $\psi(\varepsilon) = ?$, where ε is the empty sequence.

Definition 10 (Freivalds and Smith [8]). *Given an ordinal α , an α -mind-change counter for a learner ψ is a function $\rho_\psi: SEQ \rightarrow (\alpha + 1)$ such that $\rho_\psi(\varepsilon) = \alpha$ and $\psi(\sigma) \neq \psi(\sigma \diamond x)$ implies $\rho_\psi(\sigma) > \rho_\psi(\sigma \diamond x)$. We say that a concept space $\mathcal{L}$ is learnable with mind-change bound α if and only if there exists some ψ that learns $\mathcal{L}$ with some α -mind-change counter ρ_ψ . We say that $\mathcal{L}$ has mind-change complexity α if and only if $\mathcal{L}$ is learnable with mind-change bound α but is not learnable with mind-change bound β for any $\beta < \alpha$. $\square$*

Intuitively, when the learner changes its hypothesis, it must decrement its ordinal counter without going below zero. Some learning theorists (such as Luo and Schulte [18]) do not require the learner to decrement its counter when its previous hypothesis was “?”. We *do* require that the learner decrement its counter after every change of hypothesis, because we feel that the theory becomes more mathematically natural this way (see [21] for further support for this approach to mind-change complexity). For example, using our definition, for every countable ordinal α , the concept space α^{op} (a concept space order-isomorphic to α with the reversed ordering) has mind-change complexity α , whereas according to Luo and Schulte’s definition, ω^{op} and $(\omega + 1)^{op}$ would both have mind-change complexity ω . Thus, our definition of *accumulation order* is slightly different than [18], but has the advantage that it is equivalent to the standard definition of the Cantor-Bendixon rank of a topological space (see [14]). The differences are minor, and it is easily seen that we can still use Luo and Schulte’s results by slightly adjusting them to meet our definitions.

Definition 11. *A concept $L \in X \subseteq \mathcal{L}$ is an isolated point of X if and only if there is a Π -open subset U of $\mathcal{L}$ such that $L \in U$ and $X \cap U = \{L\}$. If $L \in X \subseteq \mathcal{L}$ and L is not an isolated point of X , then L is an accumulation point of X . $\square$*

Definition 12 (see [14, 18]). *Let $\mathcal{L}$ be a concept space. For each ordinal α , the α -th derived set of $\mathcal{L}$, denoted $\mathcal{L}^{(\alpha)}$, is defined inductively as follows:*

1. $\mathcal{L}^{(0)}$ is defined to be $\mathcal{L}$.
2. If α is a successor ordinal, then $\mathcal{L}^{(\alpha)}$ is defined to be the set of all accumulation points of $\mathcal{L}^{(\alpha-1)}$.
3. If α is a limit ordinal, then $\mathcal{L}^{(\alpha)}$ is defined to be $\bigcap_{\beta < \alpha} \mathcal{L}^{(\beta)}$.

The accumulation order of a concept $L \in \mathcal{L}$, denoted $\text{acc}_{\mathcal{L}}(L)$, is defined as the maximal ordinal α such that $L \in \mathcal{L}^{(\alpha)}$. The accumulation order of a concept space $\mathcal{L}$, denoted $\text{acc}(\mathcal{L})$, is the least ordinal α such that $\mathcal{L}^{(\alpha)} = \mathcal{L}^{(\alpha+1)}$. If $\text{acc}(\mathcal{L}) = \alpha$ and $\mathcal{L}^{(\alpha)}$ is empty, then $\mathcal{L}$ is said to be scattered. $\square$

Since all concept spaces are countably-based, it can be shown that $\text{acc}(\mathcal{L})$ is defined for *every* concept space $\mathcal{L}$ and is always strictly less than ω_1 , the least uncountable ordinal (see Theorem I.6.9 in [14] for the key part of the proof of this claim). We can also see that every scattered concept space $\mathcal{L}$ contains at most a countable number of concepts, since every $L \in \mathcal{L}$ has a finite subset F such that $\uparrow_{\mathcal{L}} F$ contains L and no other concept with accumulation order greater than or equal to $\text{acc}_{\mathcal{L}}(L)$, which gives an injection from $\mathcal{L}$ to finite subsets of $\mathbf{N}$.

Scattered concept spaces also have an important role in *weak reductions* (see [12] for definition and details). A *weak-complete* [12] concept space is one that is learnable, and in which every other learnable concept space is weakly reducible to it. The following property is used in the characterization of weak-complete concept spaces.

Definition 13 (Jain et al. [11]). *A concept space $\mathcal{L}$ is quasi-dense if $\mathcal{L}$ is non-empty and for any finite $F \subseteq \mathbf{N}$, either there exists no concept in $\mathcal{L}$ containing F or else there exist infinitely many distinct concepts in $\mathcal{L}$ containing F . $\square$*

Jain et al. [11] characterized weak-complete concept spaces as precisely those that are learnable and contain a quasi-dense subspace².

Theorem 5. *The following statements are equivalent for any concept space $\mathcal{L}$.*

1. $\mathcal{L}$ is a scattered concept space.
2. $\mathcal{L}$ is inferable from positive data with a mind change bound.
3. $\mathcal{L}$ does not contain a quasi-dense subspace.
4. There exists a concept space $\mathcal{K}$ with finite elasticity and an injective Π -continuous function $f: \mathcal{L} \rightarrow \mathcal{K}$.

Proof. The equivalence of 1 and 2 is the main result of [18]. To show that 1 implies 3, assume $\mathcal{S}$ is a quasi-dense subspace of $\mathcal{L}$. Since any Π -open subset of $\mathcal{L}$ that intersects $\mathcal{S}$ intersects an infinite subset of $\mathcal{S}$, it can be shown using transfinite induction that $\mathcal{S} \subseteq \mathcal{L}^{(\alpha)}$ for all α , and therefore $\mathcal{L}$ is not scattered. To see that 3 implies 1, note that if $\mathcal{L}$ is not scattered then there is a subset $\mathcal{L}^{(\alpha)} = \mathcal{L}^{(\alpha+1)}$ of $\mathcal{L}$ (for some ordinal α) that is infinite and easily seen to be quasi-dense.

That 4 implies 1 is due to Luo and Schulte [18]. They showed that every concept space with finite elasticity is scattered, and also that if a concept space $\mathcal{L}_1$ is weakly reducible to a concept space $\mathcal{L}_2$, then if $\mathcal{L}_2$ is learnable with a mind-change bound α then so is $\mathcal{L}_1$. Since a Π -continuous injection from $\mathcal{L}$ to $\mathcal{K}$ is essentially a strong reduction (a special kind of weak reduction), the claim follows.

To show 2 implies 4, assume that $\mathcal{L}$ is learnable with mind-change bound α . Let ι be an indexing of $\mathcal{L}$ (i.e. $\iota: \mathcal{L} \rightarrow \mathbf{N}$ is an injective function). For $L \in \mathcal{L}$, define $K_{\iota(L)} = \{\iota(L') \mid L = L' \text{ or } \text{acc}_{\mathcal{L}}(L) < \text{acc}_{\mathcal{L}}(L')\}$, and let $\mathcal{K} = \{K_{\iota(L)} \mid L \in \mathcal{L}\}$. $\mathcal{K}$

² To be more precise, Jain et al. [11] included an effectiveness condition since they were concerned with effective learners and reductions. However, the proof can easily be generalized to this form when allowing non-effective learners and reductions.

has finite elasticity, because a proof that $\mathcal{K}$ has infinite elasticity could be used to construct an infinitely decreasing sequence of ordinals (the reader should also note that $\text{acc}(\mathcal{L}) = \text{acc}(\mathcal{K})$). Finally, the function $f: \mathcal{L} \rightarrow \mathcal{K}$ such that $f(L) = K_{\iota(L)}$ is clearly an injection and can be shown to be Π -continuous in the usual way. $\square$

The equivalence of 1 and 4 is interesting because it is an example of reducing a “complicated” concept space to a less “complicated” one (in the sense that each concept in $\mathcal{K}$ is the closure of a single natural number, and $\mathcal{A}(\mathcal{K})$ is Noetherian). The equivalence of 2 and 3 is particularly interesting because, combined with Luo and Schulte’s work on the relationship between weak-reductions and mind-change complexity [18], it shows that without any restrictions concerning computability, the ordering relation between learnable concept spaces induced by weak reductions is strictly weaker than the ordering relation induced by mind-change complexity.

4.3 Alexandrov Concept Spaces

Alexandrov spaces are topological spaces that are useful because of their close relationship to partial orders (see [13]).

Definition 14. *An Alexandrov concept space is a concept space in which every concept has a smallest open neighborhood.* $\square$

Note that if U is a smallest open neighborhood of $L \in \mathcal{L}$, then U must be equal to $\uparrow_{\mathcal{L}} L$. The following property was proposed as part of a sufficient condition for computable learners to learn an indexed family of recursive sets.

Definition 15 (Angluin [2], Kobayashi [17]). *Let L be a concept in $\mathcal{L}$. A characteristic set for L is a finite set F such that $F \subseteq L$ and for all $L' \in \mathcal{L}$, if $F \subseteq L'$ then $L \subseteq L'$.* $\square$

The following theorem shows that these properties are equivalent, and furthermore that they are topologically invariant properties. It is also clear that every Alexandrov concept space is countable.

Theorem 6. *The following statements are equivalent for any concept space $\mathcal{L}$.*

1. $\mathcal{L}$ is an Alexandrov concept space.
2. Every L in $\mathcal{L}$ has a characteristic set.
3. Every L in $\mathcal{L}$ is compact in $\mathcal{A}(\mathcal{L})$.

Proof. Easily follows from the definitions. $\square$

As is mentioned in [23], the second statement in the above theorem is also equivalent to the *finite cross property* proposed by Sato and Umayahara [24]³.

The topological properties of an Alexandrov concept space are determined by the subset relation among its concepts. Π -continuous mappings to and from Alexandrov concept spaces are particularly easy to work with because of the following two lemmas.

³ We thank the anonymous referee for introducing us to [23].

Lemma 1. *For any Alexandrov concept space $\mathcal{L}$ and any concept space $\mathcal{K}$, a map $f: \mathcal{L} \rightarrow \mathcal{K}$ is Π -continuous if and only if it is monotonic with respect to set inclusion.* $\square$

Note, however, that *all* Π -continuous functions are monotonic. The above lemma shows that monotonicity is sufficient for Alexandrov concept spaces.

Lemma 2. *Assume that $\mathcal{L}$ is an Alexandrov concept space, and let $f: \mathcal{K} \rightarrow \mathcal{L}$ be a Π -continuous map from any concept space $\mathcal{K}$ to $\mathcal{L}$. Then for every $K \in \mathcal{K}$ there is a finite $F \subseteq K$ such that $A(f)(C_{\mathcal{K}}(F)) = f(K)$.* $\square$

It is easy to see that the class of scattered Alexandrov concept spaces properly contains the class of concept spaces with finite elasticity. In addition, there exists non-scattered Alexandrov spaces, such as $\mathcal{FIN}$ (the set of all finite subsets of $\mathbb{N}$), and also non-Alexandrov scattered spaces, such as $\mathcal{L}_2 = \{L_\omega\} \cup \{L_i \mid i \geq 0\}$, where $L_i = \{\langle i, 1 \rangle\} \cup \{\langle j, 0 \rangle \mid j \leq i\}$ for $i \geq 0$, and $L_\omega = \{\langle j, 0 \rangle \mid j \geq 0\}$.

The accumulation order of scattered Alexandrov concept spaces has the following simple characterization (which is a transfinite generalization of Luo and Schulte's notion of *inclusion depth* [18]). For any scattered Alexandrov concept space $\mathcal{L}$, inductively define $\mu_{\mathcal{L}}(L) = \sup\{\mu_{\mathcal{L}}(L') + 1 \mid L \subset L' \in \mathcal{L}\}$ for $L \in \mathcal{L}$.

Theorem 7. $\text{acc}(\mathcal{L}) = \sup\{\mu_{\mathcal{L}}(L) + 1 \mid L \in \mathcal{L}\}$ for any scattered Alexandrov concept space $\mathcal{L}$. $\square$

4.4 Finite Tell-Tales

The last major property we will investigate is the notion of a *finite tell-tale*. This was given by Angluin as part of a necessary and sufficient condition for the learnability of a concept space.

Definition 16 (Angluin [1]). *Let L be a concept in $\mathcal{L}$. A finite tell-tale for L is a finite set F such that $F \subseteq L$ and for all $L' \in \mathcal{L}$, if $F \subseteq L'$ then $L' \not\subseteq L$.* $\square$

This is related to the following topological notion (this is the only place in this paper where “closed” is meant in the topological sense).

Definition 17. *A subset S of topological space X is locally closed in X if and only if there exists an open set U and a closed set V such that $S = U \cap V$.* $\square$

The following is a separation axiom proposed by Aull and Thron [3] that is strictly between the T_0 and T_1 axioms⁴.

Definition 18 (Aull and Thron [3]). *A T_D -space is a topological space X such that $\{x\}$ is locally closed in X for every $x \in X$.* $\square$

⁴ The authors learned of this separation axiom from [13], and at the time of writing this paper have not yet been able to obtain a copy of [3].

Theorem 8. *The following statements are equivalent for any concept space $\mathcal{L}$.*

1. $\mathcal{L}$ is countable and every L in $\mathcal{L}$ has a finite tell-tale.
2. $\mathcal{L}$ is learnable in the limit from positive data.
3. $\mathcal{L}$ is a countable T_D -space.
4. There exists an Alexandrov concept space $\mathcal{K}$ and an injective Π -continuous function $f: \mathcal{L} \rightarrow \mathcal{K}$.

Proof. The equivalence of 1 and 2 is essentially due to Angluin [1] (she proved an effective version, but the generalized version is an easy corollary [20]). The equivalence of 1 and 3 is easily seen by noting that F is a finite tell-tale of L if and only if $(\uparrow_{\mathcal{L}} F) \cap (\downarrow_{\mathcal{L}} L) = \{L\}$, and that $\downarrow_{\mathcal{L}} L$ is Π -closed for every $L \in \mathcal{L}$.

The proof that 2 implies 4 is due to Jain et al. [11], where they showed that every learnable concept space can be strongly reduced to $\mathcal{RLNIT}_{0,1}$, which is defined as all subsets of the rationals $\mathbf{Q}$ of the form $\{q \in \mathbf{Q} \mid 0 \leq q \leq r\}$ for each rational r between 0 and 1 (inclusive). $\mathcal{RLNIT}_{0,1}$ is easily seen to be an Alexandrov concept space, and the essence of their proof (although it is done for effective learners) can easily be generalized. This induces, as we mentioned earlier, an injective Π -continuous function f from $\mathcal{L}$ to $\mathcal{RLNIT}_{0,1}$.

To see that 4 implies 1, let $L \in \mathcal{L}$. By Lemma 2, there is a finite $F \subseteq L$ such that $\mathcal{A}(f)(C_{\mathcal{L}}(F)) = f(L)$. For any $L' \in \mathcal{L}$ such that $L' \subseteq L$ and $F \subseteq L'$, the monotonicity of $\mathcal{A}(f)$ implies that $\mathcal{A}(f)(C_{\mathcal{L}}(F)) \subseteq \mathcal{A}(f)(L') \subseteq \mathcal{A}(f)(L)$. Thus $f(L') = f(L)$. Since f is injective, $L = L'$. $\square$

It is clear from the above proof that the T_D axiom states that every concept has a finite tell-tale. Being a topologically invariant property, it is then meaningful to say that every real number in the Euclidean real line has a finite tell-tale. Since the space of reals $\mathbf{R}$ are uncountable, they are not learnable by the standard definition. However, if we let $\mathcal{H}_{\mathbf{R}} = \{\eta(x) \mid x \in \mathbf{R}\}$, where $\eta(x) = \{\langle c, r \rangle \in \mathbf{Q} \times \mathbf{Q} \mid r > |x - c|\}$ if $x \in \mathbf{R} \setminus \mathbf{Q}$ and $\eta(x) = \{\langle x, 0 \rangle\}$ if $x \in \mathbf{Q}$ ($\mathbf{Q}$ being the set of rationals), and let $h_{\mathbf{R}}: \mathcal{H}_{\mathbf{R}} \rightarrow \mathbf{R}$ be defined as $h_{\mathbf{R}}(\eta(x)) = x$, then we can see that the Euclidean real line is learnable according to Definition 2 using the non-discrete hypothesis space $\langle \mathcal{H}_{\mathbf{R}}, h_{\mathbf{R}} \rangle$. Intuitively, a learner in this case would be required to converge exactly for rational numbers, but only gradually for irrational numbers.

Given a hypothesis space $\langle \mathcal{H}, h \rangle$ for $\mathcal{L}$ and a Π -continuous injection $f: \mathcal{L} \rightarrow \mathcal{K}$ to an Alexandrov concept space $\mathcal{K}$, we can define a learner ψ_f for $\mathcal{L}$ in the following way. For $\sigma \in SEQ$, let $rng(\sigma)$ be the set of elements in σ other than “#”. Let $\psi_f(\varepsilon) = ?$. Assume $\psi_f(\sigma)$ is defined, and for $x \in \mathbf{N} \cup \{\#\}$, let:

$$\psi_f(\sigma \diamond x) = \begin{cases} H_L & \text{if } \mathcal{A}(f)(C_{\mathcal{L}}(rng(\sigma \diamond x))) = f(L), \\ \psi_f(\sigma) & \text{otherwise;} \end{cases}$$

where $H_L \in \mathcal{H}$ is some pre-defined hypothesis such that $h(H_L) = L$. From the above theorem, it is easy to see that ψ_f learns $\mathcal{L}$ in the limit from positive data. Although not all learners for $\mathcal{L}$ can be defined in this way, many “intuitive” learning strategies are of this form. For example, consider $\mathcal{L}_2$, which we gave

earlier as an example of a scattered concept space that is not Alexandrov. We can define a concept space $\mathcal{N}_\perp = \{N_\perp\} \cup \{N_i \mid i \geq 0\}$, where $N_\perp = \{0\}$ and $N_i = \{0, i + 1\}$. Define the Π -continuous function $f: \mathcal{L}_2 \rightarrow \mathcal{N}_\perp$ so that $f(L_\omega) = N_\perp$ and $f(L_i) = N_i$ for $i \geq 0$. Intuitively, ψ_f chooses a hypothesis for L_ω until it sees evidence that confirms otherwise. It is easy to see that ψ_f learns $\mathcal{L}_2$ with an optimal mind-change bound.

Finally, we mention that the class of countable T_D concept spaces properly includes both the class of Alexandrov concept spaces and the class of scattered concept spaces. An example of a learnable concept space that is neither Alexandrov nor scattered is $\mathcal{COSINGLE} = \{\mathbf{N} \setminus \{n\} \mid n \in \mathbf{N}\}$. The “intuitive” learning strategy for this space is given by a Π -continuous injection into the ordinal ω (viewed as a concept space, i.e., for each $n \in \mathbf{N}$ there is a concept $\bar{n} \in \omega$ containing all natural numbers less than n) which maps $\mathbf{N} \setminus \{n\}$ to $\bar{n}$.

5 Concluding Remarks

We have made several connections between topology and learning in the limit, and have analyzed some of the structural properties of concept spaces.

By interpreting both texts and hypothesis spaces topologically, we can now apply the information theoretic intuitions from topology and domain theory to various aspects of the learning process. For example, open sets are often thought of as “observable” properties by domain theorists, and since the goal of learning is to identify a particular point in the space, smaller open sets can be viewed as more informative observations. It is important to note that the topology on the concept space is determined by the way it is presented, thus informants and queries will result in different topologies on the concept space. The hypothesis space $\mathcal{H}$ generally induces an even finer topology (by taking the quotient topology) on the concept space. Thus, $\mathcal{H}$ is a more “informative” representation of the concept space than texts are, since it makes more properties observable. This gives us an information theoretic interpretation of learning as the process of converting a given representation of a concept into a more “informative” representation. However, since the limiting process is non-continuous, the open sets induced by $\mathcal{H}$ are only observable in the limit, via the learner, given the initial representation of the concept space by texts.

Cartesian closed categories are important in modeling the semantics of computation, but unfortunately the category of countably based T_0 spaces is not cartesian closed. However, by replacing $T(\mathcal{L})$ with an admissible representation for $\mathcal{L}$ in Definition 2, we can naturally extend the learning in the limit paradigm to the cartesian closed category of spaces with admissible representations [25]. Such a generalization would give an approach to investigating the semantic connections between learning and computation, which has not yet been thoroughly investigated. We expect that this would lead to a more “goal oriented” approach to learning, where the interest becomes in determining whether or not a concept space can be learned “well enough” to compute some function between concept spaces to an arbitrarily high precision. This approach would probably require

us to allow non-discrete hypothesis spaces (or take some similar tactic) to relax our definition of convergence.

The “problem of induction” occurs when generalizing a finite number of observations to an infinite hypothesis, or in other words, when the learner conjectures a hypothesis that is not completely entailed by the observations. By saying that a set of observations X entails a set Y , we mean that $Y \subseteq C_{\mathcal{L}}(X)$. The use of the word “entails” is justified, and can be more thoroughly formalized, by the fact that $\langle \mathcal{A}(\mathcal{L}), C_{\mathcal{L}} \rangle$ is essentially an abstract algebraic logic (see [4]), where we view $\mathbf{N}$ as a set of formulas, and logical consequence between a set of formulas X and a formula ϕ is given by $X \vdash \phi \iff \phi \in C_{\mathcal{L}}(X)$. The role of closure operators in learning is that they give an upper bound of what deductive reasoning (meaning true assumptions lead to true conclusions) can tell us about which elements of $\mathbf{N}$ are “true” in the sense that they belong to the unknown concept being presented. These observations are important so that we can better distinguish between the deductive and inductive reasoning that is used in learning.

Acknowledgements

This work was partly supported by Grant-in-Aid for Scientific Research (B) 19300046 from JSPS, and by Kyoto University Global COE “Information Education and Research Center for Knowledge-Circulating Society”. We also thank the anonymous referees for their helpful comments.

References

- [1] Angluin, D.: Inductive inference of formal languages from positive data. *Information and Control* 45, 117–135 (1980)
- [2] Angluin, D.: Inference of reversible languages. *Journal of the ACM* 29, 741–765 (1982)
- [3] Aull, C.E., Thron, W.J.: Separation axioms between T_0 and T_1 . *Indag. Math.* 24, 26–37 (1963)
- [4] Brown, D.J., Suszko, R.: Abstract logics. *Dissertationes Mathematicae* 102, 9–41 (1973)
- [5] Burris, S., Sankappanavar, H.P.: *A Course in Universal Algebra*. Springer, Heidelberg (1981)
- [6] de Brecht, M., Kobayashi, M., Tokunaga, H., Yamamoto, A.: Inferability of closed set systems from positive data. In: Washio, T., Satoh, K., Takeda, H., Inokuchi, A. (eds.) *JSAI 2006. LNCS (LNAI)*, vol. 4384, pp. 265–275. Springer, Heidelberg (2007)
- [7] de Brecht, M., Yamamoto, A.: Mind change complexity of inferring unbounded unions of pattern languages from positive data. In: Balcázar, J.L., Long, P.M., Stephan, F. (eds.) *ALT 2006. LNCS (LNAI)*, vol. 4264, pp. 124–138. Springer, Heidelberg (2006)
- [8] Freivalds, R., Smith, C.H.: On the role of procrastination for machine learning. *Information and Computation* 107, 237–271 (1993)
- [9] Gierz, G., Hofmann, K.H., Keimel, K., Lawson, J.D., Mislove, M.W., Scott, D.W.: *Continuous Lattices and Domains*. Cambridge University Press, Cambridge (2003)

- [10] Gold, E.M.: Language identification in the limit. *Information and Control* 10, 447–474 (1967)
- [11] Jain, S., Kinber, E., Wiehagen, R.: Language learning from texts: Degrees of intrinsic complexity and their characterizations. *Journal of Computer and System Sciences* 63, 305–354 (2001)
- [12] Jain, S., Sharma, A.: The intrinsic complexity of language identification. *Journal of Computer and System Sciences* 52, 393–402 (1996)
- [13] Johnstone, P.T.: *Stone Spaces*. Cambridge University Press, Cambridge (1982)
- [14] Kechris, A.: *Classical Descriptive Set Theory*. Springer, Heidelberg (1995)
- [15] Kelley, J.: *General Topology*. Springer, Heidelberg (1975)
- [16] Kelly, K.: *The Logic of Reliable Inquiry*. Oxford University Press, Oxford (1996)
- [17] Kobayashi, S.: Approximate identification, finite elasticity and lattice structure of hypothesis space. Technical Report, CSIM 96-04, Dept. of Compt. Sci. and Inform. Math., Univ. of Electro- Communications (1996)
- [18] Luo, W., Schulte, O.: Mind change efficient learning. *Information and Computation* 204, 989–1011 (2006)
- [19] Mac Lane, S.: *Categories for the Working Mathematician*. Springer, Heidelberg (1971)
- [20] Martin, E., Osherson, D.: *Elements of Scientific Inquiry*. MIT Press, Cambridge (1998)
- [21] Martin, E., Sharma, A., Stephan, F.: Unifying logic, topology and learning in parametric logic. *Theoretical Computer Science* 350, 103–124 (2006)
- [22] Motoki, T., Shinohara, T., Wright, K.: The correct definition of finite elasticity: Corrigendum to identification of unions. In: *Proc. COLT 1991*, p. 375 (1991)
- [23] Sato, M., Mukouchi, Y., Zheng, D.: Characteristic sets for unions of regular pattern languages and compactness. In: Richter, M.M., Smith, C.H., Wiehagen, R., Zeugmann, T. (eds.) *ALT 1998. LNCS (LNAI)*, vol. 1501, pp. 220–233. Springer, Heidelberg (1998)
- [24] Sato, M., Umayahara, K.: Inductive inferability for formal languages from positive data. *IEICE Trans. Inf. and Syst.* E75-D(4), 415–419 (1992)
- [25] Schröder, M.: Extended admissibility. *Theoretical Computer Science* 284, 519–538 (2002)
- [26] Stephan, F., Ventsov, Y.: Learning algebraic structures from text. *Theoretical Computer Science* 268, 221–273 (2001)
- [27] Weihrauch, K.: *Computable Analysis*. Springer, Heidelberg (2000)
- [28] Wright, K.: Identification of unions of languages drawn from an identifiable class. In: *Proc. COLT 1989*, pp. 328–333 (1989)

Dynamically Delayed Postdictive Completeness and Consistency in Learning

John Case and Timo Kötzing

Department of Computer and Information Sciences,
University of Delaware, Newark, DE 19716-2586, USA
{case,koetzing}@cis.udel.edu

Abstract. In computational function learning in the limit, an algorithmic *learner* tries to find a program for a computable function g given successively more values of g , each time outputting a conjectured program for g . A learner is called *postdictively complete* iff all available data is correctly postdicted by each conjecture.

Akama and Zeugmann presented, for each choice of natural number δ , a relaxation to postdictive completeness: each conjecture is required to postdict only all *except the last δ seen* data points.

This paper extends this notion of delayed postdictive completeness from *constant* delays to *dynamically* computed delays. On the one hand, the delays can be different for different data points. On the other hand, delays no longer need to be by a fixed finite number, but any type of computable countdown is allowed, including, for example, countdown in a system of ordinal notations and in other graphs disallowing *computable* infinitely descending counts.

We extend many of the theorems of Akama and Zeugmann and provide some feasible learnability results. Regarding *fairness* in feasible learning, one needs to limit use of tricks that postpone output hypotheses until there is enough time to “think” about them. We see, for polytime learning, postdictive completeness (and delayed variants): 1. allows *some* but *not* all postponement tricks, and 2. there is a surprisingly tight boundary, for polytime learning, between what postponement is allowed and what is not. For *example*: 1. the set of polytime computable functions *is* polytime postdictively completely learnable employing some postponement, *but* 2. the set of exptime computable functions, while polytime learnable with a little more postponement, is *not* polytime postdictively completely learnable! We have that, for w a notation for ω , the set of exptime functions *is* polytime learnable with *w-delayed* postdictive completeness. Also provided are generalizations to further, small constructive limit ordinals.

1 Introduction

“Explanatory learning”, or Ex-learning for short, is a standard model of limit learning of computable functions. In this model a learner is given successively longer initial segments of a target function. For each initial segment of the target

function, the learner gives an hypothesis. The learner is said to successfully *Ex-learn* the target function iff the infinite sequence of hypotheses generated by the learner on the initial segments of the target functions converges in the limit to a (single) correct program for the target function [JORS99].

In some literature on limit learning this intuitively simple success criterion is used as a minimal requirement for success, and additional requirements are defined and examined. We call two such extra requirements *postdictive completeness* (the hypotheses correctly postdict the data seen so far) and *postdictive consistency* (the hypotheses do not explicitly contradict the given data) [Bär74, BB75, Wie76, Wie78].¹ There are Ex-learnable sets of functions that cannot be learned with the additional requirement of postdictive completeness or consistency.

Akama and Zeugmann [AZ07] presented success criteria that are a little less restrictive than postdictively complete Ex-learning. Their criteria delay the requirement to postdict a given datum by a fixed natural number δ of (not necessarily distinct) hypotheses output. For ordinary postdictive completeness, if a learner h has seen so far, on a computable g input, $g(0), \dots, g(n-1)$, then h 's corresponding hypothesis, p_n , must correctly compute $g(0), \dots, g(n-1)$.² For delay δ , Akama and Zeugmann, require only that, on $g(0), \dots, g(n-1)$, h 's *later* hypothesis, $p_{n+\delta}$, must correctly compute $g(0), \dots, g(n-1)$. Essentially, the delay δ learner could, after seeing $g(0), \dots, g(n-1)$, run a counter down from δ to 0 to see which future hypothesis must correctly compute $g(0), \dots, g(n-1)$.

In the present paper we extend this notion of delayed postdictive completeness from *constant* delays δ to *dynamically computed* delays. One of the ways we consider herein to do this involves counting down from notations for constructive ordinals. We explain. Everyone knows how to use the natural numbers for counting, including for counting *down*. Freivalds and Smith [FS93], as well as [ACJS04], employed in learning theory *notations for constructive ordinals* [Rog67, § 11.7] as devices for algorithmic counting down. Theorems 4 and 5 in Section 3 provide strong justification for studying the herein ordinal countdown variants of Postdictive Completeness.

[SSV04] gives a further generalized notion of counting down. They consider certain partial orders with no *computable* infinitely descending chains. In the present paper we consider arbitrary and also computable graphs with no infinite, computable paths, and we algorithmically count “down” along their paths. Theorem 11, in Section 4.2 below, gives a nice example of a linearly ordered, com-

¹ We use the terminology *postdictive completeness* because the the hypotheses must *completely* postdict the data seen to that point. We use the terminology *postdictive consistency* because the the hypotheses need only *avoid explicit inconsistencies* with the data seen to that point. Such an hypothesis may, then, on some input for which the data seen to that point tells the answer, go undefined (i.e., go into an infinite loop) and, thereby, not explicitly contradict the known data. In the literature on these requirements, except for [Ful88], what we call postdictively complete is called *consistent*, and what we call postdictively consistent is called *conformal*.

² Note that, for $n = 0$, the data seen is empty and the output hypothesis, p_0 , is unconstrained.

putable such graph which nonetheless has infinite paths (just not computable ones). We call our graphs in the present paper, *countdown graphs*.

We make use of countdown graphs for delaying the requirement of postdictive completeness (respectively, consistency) by requiring a learner to start an *independent* countdown for each datum $g(i)$ seen and to be postdictively complete (respectively, consistent) regarding $g(i)$ as soon as the countdown for $g(i)$ terminates.³

Section 2 will introduce the notation and concepts used in this paper.

In the prior literature we also see further variants of postdictive completeness and consistency not based on delay. For example, [CJSW04] surveys with references these variants. Roughly, below, when we attach $\mathcal{R}$ to the front of a name of a criterion requiring postdictive completeness or consistency, this means that the associated learnability must be witnessed by a (total) computable learner as opposed to just a partial computable learner (defined at least where it minimally needs to be); when we attach a $\mathcal{T}$ to the front of a name of a criterion requiring postdictive completeness (respectively, consistency), this means that the associated learnability must be witnessed by a (total) computable learner which is postdictively complete (respectively, consistent) on *all* input functions regardless of whether the learner actually learns them.

Sections 3 and 4 present our results. All of our results in Section 3 provide information about polynomial time learners. Furthermore, some of our results in Section 4.1 entail learnability with linear time learners. These time bounds are uniform bounds on how much time it takes the learner to conjecture each hypothesis in terms of the total size of the input data it can use for making this conjecture. Suppose for discussion p is a polynomial time bound. Pitt [Pit89] notes that Ex-learning allows unfair postponement tricks, i.e., a learner h can put off outputting significant conjectures based on data σ until it has seen a much larger sequence of data τ so that $p(|\tau|)$ is enough time for h to think about σ as long as it needs.⁴ In this way the polytime restriction on each output does not, by itself, have the desirable effect of constraining the total learning time. Pitt [Pit89] then lays out some additional constraints toward avoiding “cheating” by such postponement tricks. He discusses in this regard what we are calling postdictive completeness. He also considers further constraints since he wants to forbid enumeration techniques [JORS99]. For our complexity-bounded results in Section 4.1 we get by with an extremely fair, restricted kind of *linear-time* learner, we call *transductive*. A transductive learner has access only to its current datum.

In Section 3 we see, from Theorems 4 and 5 and the proof of the first, that, for polytime learning, postdictive completeness (and delayed variants): 1. allows *some* but *not* all postponement tricks, and 2. there is a surprisingly tight boundary, for polytime learning, between what postponement is allowed and what is not. For *example*: 1. the set of polytime computable functions *is* polytime

³ Below we refer to a vector of such individual counts as a *multicount*.

⁴ Pitt talks in this context of *delaying tricks*. We changed this terminology due to the clash with Akama and Zeugmann’s terminology for *delayed* postdictive completeness.

postdictively completely Ex-learnable (by a complexity-bounded enumeration technique) employing some postponement, *but* 2. the set of exptime computable functions, while polytime Ex-learnable with a little more postponement, is *not* polytime postdictively completely Ex-learnable! From Theorem 4, we see that, for w a notation for ω , the set of exptime functions *is* polytime Ex-learnable with w -*delayed* postdictive completeness. Theorems 4 and 5 also provide generalizations to further, small constructive limit ordinals.

Section 4.1 shows how the different variants of our criteria relate in learning power. Our main theorem in this section is Theorem 6. For *example*, it entails that there is a set of computable functions which is postdictively *consistently* learnable (with no delays) by a transductive, linear time learner *but* is *not* postdictively *completely* learnable with delays employing *any* countdown graph.

In Section 4.2, our main result, Theorem 13, entails (including with Corollaries) *complete characterizations* of learning power *in dependence on* associated (computable) *countdown graphs*. Corollary 16 extends the finite hierarchy given in [AZ07] into the constructive transfinite.

We omit most proofs due to space constraints. A more complete version of the present paper can be found online [CK08]. Many of our omitted proofs use Kleene's Recursion Theorem [Rog67, page 214, problem 11-4] or the Case's *Operator Recursion Theorem* [Cas74] (and are a bit combinatorially difficult).

2 Mathematical Preliminaries

Any unexplained complexity-theoretic notions are from [RC94]. All unexplained general computability-theoretic notions are from [Rog67].

Strings herein are finite and over the alphabet $\{0, 1\}$. $\{0, 1\}^*$ denotes the set of all such strings; ε denotes the empty string.

$\mathbb{N}$ denotes the set of natural numbers, $\{0, 1, 2, \dots\}$. We do not distinguish between natural numbers and their *dyadic* representations as strings.⁵

For each $w \in \{0, 1\}^*$ and $n \in \mathbb{N}$, w^n denotes n copies of w concatenated end to end. For each string w , we define $\text{size}(w)$ to be the length of w . As we identify each natural number x with its dyadic representation, for all $n \in \mathbb{N}$, $\text{size}(n)$ denotes the length of the dyadic representation of n . For all strings w , we define $|w|$ to be $\max\{1, \text{size}(w)\}$.⁶

The symbols $\subseteq$, $\subset$, $\supseteq$, $\supset$ respectively denote the subset, proper subset, superset and proper superset relation between sets.

For sets A, B , we let $A \setminus B := \{a \in A \mid a \notin B\}$, $\overline{A} := \mathbb{N} \setminus A$.

We sometimes denote a function f of $n > 0$ arguments $x_1, \dots, x_n$ in lambda notation (as in Lisp) as $\lambda x_1, \dots, x_n. f(x_1, \dots, x_n)$. For example, with $c \in \mathbb{N}$, $\lambda x. c$ is the constantly c function of one argument.

⁵ The *dyadic* representation of a natural number x := the x -th finite string over $\{0, 1\}$ in *lexicographical order*, where the counting of strings starts with zero [RC94]. Hence, unlike with binary representation, lead zeros matter.

⁶ This convention about $|\varepsilon| = 1$ helps with runtime considerations.

A function ψ is *partial computable* iff there is a Turing machine computing ψ . $\mathcal{R}$ and $\mathcal{P}$ denote the set of all (total) computable and partial computable functions $\mathbb{N} \rightarrow \mathbb{N}$, respectively. If ψ is not defined for some argument x , then we denote this fact by $\psi(x)\uparrow$, and we say that ψ on x *diverges*. The opposite is denoted by $\psi(x)\downarrow$, and we say that ψ on x *converges*.

We say that a partial function ψ *converges to* p iff $\forall^\infty x : \psi(x)\downarrow = p$.

[RC94, §3] describes an *efficiently* numerically named or coded⁷ programming system for multi-tape Turing machines (TMs) which compute the partial computable functions $\mathbb{N} \rightarrow \mathbb{N}$. Herein we name this system φ . φ_p denotes the partial computable function computed by the TM-program with code number p in the φ -system, and Φ_p denotes the partial computable *runtime* function of the TM-program with code number p in the φ -system. In the present paper, we employ a number of complexity bound results from [RC94, §§ 3 & 4] regarding (φ, Φ) . These results will be clearly referenced as we use them.

We fix the 1-1 and onto pairing function $\langle \cdot, \cdot \rangle : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ from [RC94], which is based on dyadic bit-interleaving. Pairing and unpairing is computable in linear time. π_1 and π_2 , respectively, denote the unpairing into the left and right component of a given coded pair, respectively.

For all $f, g \in \mathcal{R}$ we let $\langle f, g \rangle$ denote $\lambda i. \langle f(i), g(i) \rangle$.

Whenever we consider tuples of natural numbers as input to TMs, it is understood that the general coding function $\langle \cdot, \cdot \rangle$ is used to (left-associatively) code the tuples into appropriate TM-input.

A *finite sequence* is a mapping with a finite initial segment of $\mathbb{N}$ as domain (and range, $\mathbb{N}$). $\emptyset$ denotes the empty sequence (and, also, the empty set). The set of all finite sequences is denoted by Seq . For each finite sequence σ , we will denote the first element, if any, of that sequence by $\sigma(0)$, the second, if any, with $\sigma(1)$ and so on. $\text{last}(\sigma)$ denotes the last element of σ , if any. $\# \text{elems}(\sigma)$ denotes the number of elements in a finite sequence σ , that is, the cardinality of its domain.

We use $\diamond$ (with infix notation) to denote concatenation on sequences. For any natural number x , we let $\overline{x}$ denote the sequence of length one with only element x , and we let $\overline{x}^n$ be the code of the sequence of length n , each element being x .

From now on, by convention, f, g and h with or without decoration range over (partial) functions $\mathbb{N} \rightarrow \mathbb{N}$, x, y with or without decorations range over $\mathbb{N}$ and σ, τ with or without decorations range over finite sequences of natural numbers.

Following [LV97], we fix a coding $\langle \cdot \rangle_{\text{Seq}}$ of all sequences into $\mathbb{N}$ ($= \{0, 1\}^*$) – with the following properties.

The set of all codes of sequences is decidable in linear time. The time to encode a sequence, that is, to compute $\lambda k, v_1, \dots, v_k. \langle v_1, \dots, v_k \rangle_{\text{Seq}}$ is $\mathcal{O}(\lambda k, v_1, \dots, v_k. \sum_{i=1}^k |v_i|)$. Therefore, the size of the codeword is also linear in the size of the elements: $\lambda k, v_1, \dots, v_k. |\langle v_1, \dots, v_k \rangle_{\text{Seq}}|$ is

⁷ This numerical coding guarantees that many simple operations involving the coding run in linear time. This is by contrast with historically more typical codings featuring prime powers and corresponding at least exponential costs to do simple things.

$\mathcal{O}(\lambda k, v_1, \dots, v_k \cdot \sum_{i=1}^k |v_i|)$.⁸ We also have $\lambda\langle\sigma\rangle_{\text{Seq}} \cdot \#\text{elets}(\sigma)$ is linear time computable; $\lambda\langle\sigma\rangle_{\text{Seq}}, i \cdot \begin{cases} \sigma(i), & \text{if } i < \#\text{elets}(\sigma); \\ 0, & \text{otherwise,} \end{cases}$ is linear time computable; and

$$\forall \sigma : \#\text{elets}(\sigma) \leq |\langle\sigma\rangle_{\text{Seq}}|. \quad (1)$$

Henceforth, we will many times identify a finite sequence σ with its code number $\langle\sigma\rangle_{\text{Seq}}$. However, when we employ expressions such as $\sigma(x)$, $\sigma = f$ and $\sigma \subset f$, we consider σ as a *sequence*, not as a number.

For a partial function g and $i \in \mathbb{N}$, if $\forall j < i : g(j) \downarrow$, then $g[i]$ is defined to be the finite sequence $g(0), \dots, g(i-1)$.

A *pre-order* is a pair $(A, \leq_A)$ such that $\leq_A$ is a transitive and reflexive binary relation on A .

Church and Kleene introduced systems of ordinal notations. See Rogers [Rog67, § 11.7]. For us, a *system of ordinal notations* is a pair $(\mathcal{N}, \leq_{\mathcal{N}})$ and associated functions $k_{\mathcal{N}}, p_{\mathcal{N}}, q_{\mathcal{N}} \in \mathcal{P}$ and $\nu_{\mathcal{N}}$ mapping $\mathcal{N}$ into the set of all constructive ordinals, such that $\mathcal{N} \subseteq \mathbb{N}$, and, for all $u, v \in \mathcal{N}$, we have: $u \leq_{\mathcal{N}} v$ iff $\nu_{\mathcal{N}}(u) \leq \nu_{\mathcal{N}}(v)$; if $\nu_{\mathcal{N}}(u) = 0$, then $k_{\mathcal{N}}(u) = 0$; if $\nu_{\mathcal{N}}(u)$ is successor ordinal, then $k_{\mathcal{N}}(u) = 1$ and $\nu_{\mathcal{N}}(p_{\mathcal{N}}(u)) + 1 = \nu_{\mathcal{N}}(u)$; if $\nu_{\mathcal{N}}(u)$ is limit ordinal, then $k_{\mathcal{N}}(u) = 2$ and $\varphi_{q_{\mathcal{N}}(u)}$ is a monotonic increasing computable function such that $\nu_{\mathcal{N}} \circ \varphi_{q_{\mathcal{N}}(u)}$ converges to $\nu_{\mathcal{N}}(u)$.

Note that $\leq_{\mathcal{N}}$ is not necessarily computable. If it is, then $(\mathcal{N}, \leq_{\mathcal{N}})$ is called *computably related*.

For countdown in polynomial time, we use *feasibly related feasible systems of ordinal notations* [CKP07]. In such systems, many predicates and operations on notations are feasibly computable. For example, one can use the (efficiently) coded tuple $\langle a_n, \dots, a_0 \rangle$ as a notation for the ordinal $\omega^n \cdot a_n + \dots \omega^0 \cdot a_0$. The resulting system of ordinal notations gives a notation to all ordinals $< \omega^\omega$ and allows for polytime comparing, adding and so on.

Note that, for any constructive ordinal α , there is a computably related system of ordinal notations which gives a notation to α [Rog67]; furthermore, there is also a feasibly related feasible system of ordinal notations giving a notation to α [CKP07].

In this paper we consider *several* indexed families of learning criteria. We proceed somewhat abstractly to avoid needless terminological repetitions.

For each $\mathcal{C} \subseteq \mathcal{P}$ and $\delta \subseteq \mathcal{R}^2$, we say that the pair $(\mathcal{C}, \delta)$ is a *learning criterion* (for short, *criterion*). The set $\mathcal{C}$ is called a *learner admissibility restriction*, and intuitively serves as a limitation on what functions will be considered as learners. Typical learner admissibility restrictions are $\mathcal{P}, \mathcal{R}$, as well as complexity classes. The predicate δ is called a *sequence acceptance criterion*, intuitively restricting what output-sequences by the learner are considered a successful learning of a given function. For $h \in \mathcal{P}, g \in \mathcal{R}$ we say that h $(\mathcal{C}, \delta)$ -learns g iff $h \in \mathcal{C}$ and $(\lambda x. h(g[x]), g) \in \delta$. For $h \in \mathcal{P}, g \in \mathcal{R}$, we call $\lambda x. h(g[x])$ the *learning-sequence* of h given g . Here's an *example* δ , herein called **Ex**. Let $\mathbf{Ex} = \{(\langle p, d \rangle, q) \in$

⁸ For these $\mathcal{O}$ -formulas, $|\varepsilon| = 1$ helps.

$\mathcal{R}^2 \mid p$ converges to some $e \wedge \varphi_e = q$. Intuitively, $((p, d), q) \in \mathbf{Ex}$ means that the learning-sequence $\langle p, d \rangle$ successfully learns the function q iff: for some i , $p(i)$ is a correct program number for q , and this hypothesized program number will never change after that point i . N.B. For *this* example, the learning-sequence is a sequence of coded pairs and $\mathbf{Ex}$ completely disregards the second component d . Some *other* sequence acceptance criteria below make use of d as an auxiliary output of the learner. In these cases, d will code countdowns until some events of interest must happen. For $h \in \mathcal{P}$ and $\mathcal{S} \subseteq \mathcal{R}$ we say that h $(\mathcal{C}, \delta)$ -learns $\mathcal{S}$ iff, for all $g \in \mathcal{S}$, h $(\mathcal{C}, \delta)$ -learns g . The set of $(\mathcal{C}, \delta)$ -learnable sets of computable functions is $\mathcal{C}\delta := \{\mathcal{S} \subseteq \mathcal{R} \mid \exists h \in \mathcal{C} : h \text{ } (\mathcal{C}, \delta)\text{-learns } \mathcal{S}\}$. Instead of writing the pair $(\mathcal{C}, \delta)$, we will ambiguously write $\mathcal{C}\delta$. We will omit $\mathcal{C}$ if $\mathcal{C} = \mathcal{P}$.⁹ One way to *combine* two sequence acceptance criteria δ and δ' is to intersect them as sets. We write $\delta\delta'$ for the intersection, and we present examples featuring countdowns in the next section.

We can turn a given sequence acceptance criterion δ into a learner admissibility restriction $\mathcal{T}\delta$ by admitting only those learners that obey δ *on all input*: $\mathcal{T}\delta := \{h \in \mathcal{P} \mid \forall g \in \mathcal{R} : (\lambda x. h(g[x]), g) \in \delta\}$.

The following two definitions formalize the intuitive discussion about countdown graphs as given above in Section 1.

A *graph* is a pair $(G, \rightarrow)$, where $G \subseteq \mathbb{N}$ and $\rightarrow$ is a binary relation on G . We will use infix notation for $\rightarrow$. For each graph $(G, \rightarrow)$, we say that τ is a G -*path* iff $\#elets(\tau) > 0$, $\forall i < \#elets(\tau) : \tau(i) \in G$ and $\forall i < \#elets(\tau) - 1 : \tau(i) \rightarrow \tau(i+1)$. For each graph G , let $\vec{G}$ denote the set of all G -paths. (S, R) is a *subgraph* of $(G, \rightarrow)$, iff $S \subseteq G$ and R is $\rightarrow$ restricted to $(S \times S)$.

For all $m, n \in \mathbb{N}$, we write $m \rightarrow^* n$ (respectively, $m \rightarrow^+ n$) iff there is a G -path τ such that $\tau(0) = m$, $\text{last}(\tau) = n$ (respectively, additionally $\#elets(\tau) > 1$). We sometimes write G for $(G, \rightarrow)$. A graph $(G, \rightarrow)$ is said to be *computable* iff G and $\rightarrow$ are computable. Note that $G \in \mathcal{G}$ is computable iff $\vec{G}$ is computable. For a graph $(G, \rightarrow)$ we sometimes identify $m \in G$ with $\{n \in G \mid m \rightarrow^+ n\}$. With every pre-order $(A, \leq_A)$ we associate the graph $(A, >_A)$, where, for all $a, b \in A$, $a >_A b$ iff $(b \leq_A a \text{ and } a \not\leq_A b)$.

A graph $(G, \rightarrow)$ is called a *countdown graph*, iff $\neg \exists r \in \mathcal{R} \forall i \in \mathbb{N} : r(i) \rightarrow r(i+1)$. Note that if G is a countdown graph, then so is every subgraph of G . Let $\mathcal{G}$, $\mathcal{G}_{\text{comp}}$, respectively, denote the set of all and all computable countdown-graphs, respectively.

Example countdown graphs can be obtained from systems of ordinal notations. Let $(\mathcal{N}, \leq_{\mathcal{N}})$ be a system of ordinal notations. Then, $(\mathcal{N}, \leq_{\mathcal{N}})$ is a pre-order without infinite descending chains, so the graph associated with $(\mathcal{N}, \leq_{\mathcal{N}})$ is a countdown graph. If $(\mathcal{N}, \leq_{\mathcal{N}})$ is computably related, then the associated graph will be computable.

⁹ Thus, every sequence acceptance criterion denotes at the same time a learning criterion and the set of learnable sets. It will be clear from context which meaning is intended. An example: $\mathbf{Ex}$, then, denotes sequence acceptance criterion $\mathbf{Ex}$, learning criterion $(\mathcal{P}, \mathbf{Ex})$ and set $\mathcal{PEx}$ of $(\mathcal{P}, \mathbf{Ex})$ -learnable sets.

In Theorem 11 below we give one example of a countdown graph not based on a system of ordinal notations.

Soon we define what postdictive completeness, respectively consistency, with respect to $G \in \mathcal{G}$ means. Intuitively, every learner is required to have two outputs: a hypothesis, and a countdown output. For any learner $g \in \mathcal{R}$, if the learner sees $g[i]$, the countdown output will need to encode one countdown for each $j < i$. As soon as the countdown for a given data item is over, the hypothesis has to be postdictively complete, respectively consistent, *for that data item*. We will refer to the countdown output of a learner as a *multicount* (as it represents more than one countdown). We refer to a learning-output of hypothesis and multicount as a *hypothesis-multicount*.

The set of all *multicountdown sequences* is defined as $\mathbb{M} := \{\sigma \in \text{Seq} \mid \forall i < \#elets(\sigma) : (\sigma(i) \in \text{Seq} \wedge \#elets(\sigma(i)) = i)\}$.¹⁰

An example multicountdown sequence is $\sigma_0 := \langle \rangle_{\text{Seq}}, \langle 3 \rangle_{\text{Seq}}, \langle 2, 3 \rangle_{\text{Seq}}, \langle 1, 2, 2 \rangle_{\text{Seq}}, \langle 0, 1, 2, 1 \rangle_{\text{Seq}}, \langle 0, 0, 2, 2 \rangle_{\text{Seq}}, \langle 2, 0, 2, 3, 1, 1 \rangle_{\text{Seq}}, \langle 0, 0, 2, 7, 0, 0, 5 \rangle_{\text{Seq}}$. σ_0 can be displayed as a matrix like this:

$$\sigma_0 = \begin{array}{c} \begin{array}{cc} & x \\ n & 0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \left(\begin{array}{ccccccc} & 3 & 2 & 1 & 0 & 0 & 2 & 0 \\ & & 3 & 2 & 1 & 0 & 0 & 0 \\ & & & 2 & 2 & 2 & 2 & 2 \\ & & & & 1 & 2 & 3 & 7 \\ & & & & & 2 & 1 & 0 \\ & & & & & & 1 & 0 \\ & & & & & & & 5 \end{array} \right). \end{array} \quad (2)$$

In (2) each column is a multicount. For example, column $x = 4$ represents the multicount $\sigma_0(4) = \langle 0, 1, 2, 1 \rangle$. Each row of (2) provides the successive values of a particular countdown. For example, the n -th row of (2) (without initial empty entries) is the n -th countdown of σ_0 . As we will see below, for an associated learner g , the n -th row will be relevant to $g(n)$.

For each $\sigma \in \mathbb{M}$ and $n < \#elets(\sigma) - 1$ we define $\text{row}(n, \sigma) := \langle \sigma(n + 1)(n), \dots, \sigma(\#elets(\sigma) - 1)(n) \rangle_{\text{Seq}}$. For σ_0 as presented above in (2), we have, for example, $\text{row}(4, \sigma_0) = \langle 2, 1, 0 \rangle_{\text{Seq}}$. Each $\text{row}(n, \sigma)$ is a countdown.

We will consider a given countdown sequence τ as *terminated* with respect to a given countdown graph $G \in \mathcal{G}$, iff $\tau \notin \vec{G}$. We then say that “ τ has terminated” or “ τ has bottomed out”. For a given multicountdown sequence we will define the set of all n such that the n -th countdown has (started and) bottomed out just below. For all σ and all $G \in \mathcal{G}$, define $\perp_G(\sigma) = \{n < \#elets(\sigma) - 1 \mid \sigma \notin \mathbb{M} \vee \text{row}(n, \sigma) \notin \vec{G}\}$. We omit the subscript G whenever no confusion can arise.

We pronounce $\perp$ as “bottom”. For $\sigma \in \mathbb{M}$, $\perp(\sigma)$ is the set of all countdown numbers where the countdown has terminated.

Let us, for example, consider the finite countdown graph G on $\{0, 1, 2, 3\}$ with the natural $>$ -order on $\mathbb{N}$. For σ_0 depicted above in (2), we have $\perp_G(\sigma_0) =$

¹⁰ Of course, $\sigma(i) \in \text{Seq}$ means that the number $\sigma(i)$ is the code of a sequence.

$\{0, 1, 2, 3, 6\}$. The example of rows $n = 4$ and $n = 5$ shows that reaching a minimal element (in this case 0) of G does not imply immediate termination of the countdown. The example of rows $n = 2$ and $n = 3$ shows how countdowns terminate when not obeying the graph relation. Note that the countdown for row $n = 6$ has terminated immediately when it started, as it started with 5, and $\langle 5 \rangle_{\text{Seq}}$ is not a G -path. From rows $n = 4$ and $n = 6$ we see that the different countdowns do not have to terminate in row order.

Next we define two families of sequence acceptance criteria, employing countdowns as described above. The rest of the paper will be concerned with studying these criteria in various settings.

Definition 1. For $G \in \mathcal{G}$ let, for all $p, d, q \in \mathcal{R}$,

- $\mathbf{Pcp}_G(\langle p, d \rangle, q) :\Leftrightarrow \forall x \forall n \in \perp_G(d[x]) : \varphi_{p(x)}(n) \downarrow = q(n)$; and
- $\mathbf{Pcs}_G(\langle p, d \rangle, q) :\Leftrightarrow \forall x \forall n \in \perp_G(d[x]) : \varphi_{p(x)}(n) \downarrow \Rightarrow \varphi_{p(x)}(n) = q(n)$.

For all $g \in \mathcal{R}$ and $h, f \in \mathcal{P}$, we say that $\langle h, f \rangle$ *works postdictively completely (respectively, consistently) on g with G -delay* iff $(\lambda i. (\langle h(g[i]), f(g[i]) \rangle), g) \in \mathbf{Pcp}_G$ (respectively, $\mathbf{Pcs}_G$). We omit “with G -delay”, if no confusion can arise.

3 Complexity Results

For this section only, let $\mathcal{N}$ be a feasibly related feasible system of ordinal notations for at least the ordinals $< \omega^2$. Let w be a notation for ω in $\mathcal{N}$. For each $n \in \mathbb{N}$, $\underline{n}$ denotes a notation for n in $\mathcal{N}$, such that $\lambda n. \underline{n}$ is computable in polytime. We will assume for all constructive ordinals α ,

$$\forall n \in \mathbb{N}, u \in \mathcal{N} : (u \text{ is notation in } \mathcal{N} \text{ for } \alpha + n) \Rightarrow n \leq u. \quad (3)$$

Definition 2. Let $\exp$ denote the function $\lambda x. 2^x$. Furthermore, for all n , we write $\exp^n$ for the n -times application of $\exp$. In particular, $\exp^0$ denotes the identity. For all k let $\text{Exp}_k \text{Programs} := \{e \mid e \in \mathbb{N} \wedge \exists p \text{ polynomial } \forall n \in \mathbb{N} : \Phi_e(n) \leq \exp^k(p(|n|))\}$ and $\mathbf{EXP}_k \mathbf{F} := \{\varphi_e \mid e \in \text{Exp}_k \text{Programs}\}$. Also, we let $\text{ExpPrograms} := \text{Exp}_1 \text{Programs}$, $\mathbf{EXPF} := \mathbf{EXP}_1 \mathbf{F}$, $\text{PolyPrograms} := \text{Exp}_0 \text{Programs}$ and $\mathbf{PF} := \mathbf{EXP}_0 \mathbf{F}$.

For $g \in \mathbf{PF}$ we say that g is *computable in polytime*, or also, *feasibly computable*. Recall that we have, by (1), $\forall \sigma : \# \text{elets}(\sigma) \leq |\sigma|$.

Definition 3. Let S, T be such that

$$\forall p, x, t : S(p, x, t) = \begin{cases} \varphi_p(x), & \text{if } \Phi_p(x) \leq |t|; \\ 0, & \text{otherwise;} \end{cases} \quad (4)$$

$$\forall p, x, t : T(p, x, t) = \begin{cases} 1, & \text{if } \Phi_p(x) \leq |t|; \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

¹¹ Specific systems of ordinal notations seen in the literature typically, perhaps always, satisfy (3).

We now use the notion introduced above for subscripting our criteria with ordinal notations instead of countdown graphs.

Theorem 4

- (a) $\mathbf{PF} \in \mathbf{PFP}_{\mathbf{cp}_0} \mathbf{Ex}$.
- (b) $\mathbf{EXPF} \in \mathbf{PFP}_{\mathbf{cp}_w} \mathbf{Ex}$.
- (c) $\forall n : \mathbf{EXP}_n \mathbf{F} \in \mathbf{PFP}_{\mathbf{cp}_{w.\underline{n}}} \mathbf{Ex}$.

Furthermore, each of (a), (b) and (c) is witnessed by a respective learner $\langle h, f \rangle$ such that $\text{range}(h) \subseteq \text{PolyPrograms}$, $\subseteq \text{ExpPrograms}$ and $\subseteq \text{Exp}_n \text{Programs}$, respectively.

Proof of (a). This proof employs a complexity-bounded enumeration technique [JORS99]. By [RC94, Theorem 3.13], there is a linear time computable patch_0 such that,

$$\forall \sigma \forall x : \varphi_{\text{patch}_0(\sigma)}(x) = \begin{cases} \sigma(x), & \text{if } x < \# \text{elets}(\sigma); \\ 0, & \text{otherwise;} \end{cases} \quad (6)$$

and all outputs of patch_0 are programs computable in linear time.

By [RC94, Theorems 4.13(b) & 4.17] there is a linear time computable e such that $\mathbf{PF} = \{\varphi_{e(j)} \mid j \in \mathbb{N}\}$ and $\forall j \in \mathbb{N} : e(j) \in \text{PolyPrograms}$. From [RC94, Corollary 3.7] S and T from (4) and (5) above are polytime computable.¹² Then, by [RC94, Lemmas 3.15 & 3.16], it is easy to see that there is $h \in \mathbf{PF}$ such that

$$\forall \sigma : h(\sigma) = \begin{cases} e(j), & \begin{array}{l} \text{if there is a minimal} \\ j \leq |\sigma| : \forall x < \# \text{elets}(\sigma) : \\ (T(e(j), x, \sigma) \wedge S(e(j), x, \sigma) = \sigma(x)); \end{array} \\ \text{patch}_0(\sigma), & \text{otherwise.} \end{cases} \quad (7)$$

To show that h converges on all $g \in \mathbf{PF}$: Let $g \in \mathbf{PF}$. Let j_0 be minimal such that $\varphi_{e(j_0)} = g$. Let p be a polynomial such that $\forall x : \Phi_{e(j_0)}(x) \leq p(|x|)$. We then have the following.

- $\forall^\infty n, j_0 \leq n \leq |g[n]|$ (by (1)).
- $\forall^\infty n \forall j < j_0 : g[n] \not\subseteq \varphi_{e(j)}$ (as j_0 minimal such that $\varphi_{e(j_0)} = g$).
- We have $\forall^\infty x : \Phi_{e(j_0)}(x) \leq x$.¹³ Hence, $\forall^\infty n \forall x \leq n : \Phi_{e(j_0)}(x) \leq n$.¹⁴ Therefore, using (1), $\forall^\infty n \forall x < n : T(e(j_0), x, g[n])$; hence, also $\forall^\infty n \forall x < n : S(e(j_0), x, g[n]) = \varphi_{e(j_0)}(x) = g(x)$.

By the three items above, we have $\forall^\infty n : h(g[n]) = e(j_0)$. Let $f = \lambda \sigma. \underline{0}$. Obviously, $\langle h, f \rangle$ witnesses $\mathbf{PF} \in \mathbf{Pcp}_0 \mathbf{Ex}$. The furthermore clause follows from the choice of e and patch_0 . $\square$ (OF (a))

¹² N.B. S and T above are variants of the S and T featured in [RC94, Corollary 3.7].

¹³ By [RC94, §2.5, (9)], there are $a, b \in \mathbb{N}$ such that $\forall x : 2^{|x|} \leq a \cdot x + b$; thus, there is $d > 0$ such that $\forall^\infty x : 2^{|x|} \leq d \cdot x$. Clearly, $\forall^\infty x : p(|x|) \leq \frac{1}{d} 2^{|x|}$. Thus, $\forall^\infty x : p(|x|) \leq x$.

¹⁴ Let n_0, n_1 be such that $\forall x \geq n_0 : \Phi_{e(j_0)}(x) \leq x$ and $\forall x < n_0 : \Phi_{e(j_0)}(x) \leq n_1$. Then, for all $n \geq \max\{n_0, n_1\}$ and for all $x \leq n$, we have (if $x < n_0$) $\Phi_{e(j_0)}(x) \leq n_1 \leq n$, and (otherwise) $\Phi_{e(j_0)}(x) \leq x \leq n$.

We will not give proofs for (b) and (c), as they are only slight modifications of the proof for (a). Note that (a) and (b) are *both* special cases of (c).

Theorem 5

- (a) $\forall n \in \mathbb{N} : \mathbf{EXPF} \notin \mathbf{PFPcp}_n \mathbf{Ex}$.
 (b) $\forall k, n \in \mathbb{N} : \mathbf{EXP}_{k+1} \mathbf{F} \notin \mathbf{PFPcp}_{w \cdot k + \underline{n}} \mathbf{Ex}$.

We will not prove (b), but only its simpler to prove special case (a).

Proof of (a). Suppose by way of contradiction otherwise as witnessed by n and $\langle h, f \rangle$. Note that $\{\sigma \mid \exists g \in \mathbf{EXPF}, \sigma \subset g\} = \text{Seq}$; thus, $\langle h, f \rangle \in \mathcal{TPcp}_{\underline{n}}$.

Define $g \in \mathcal{R}$ according to the following informal definition in stages. g_s denotes g as defined until before stage s .

```

 $g_0 = \varepsilon$ 
stage  $s = 0$  to  $\infty$ 
    if  $h(g_s \diamond \bar{0} \diamond \bar{0}^n) = h(g_s)$ 
        then  $g_{s+1} = g_s \diamond \bar{1} \diamond \bar{0}^n$ 
        else  $g_{s+1} = g_s \diamond \bar{0} \diamond \bar{0}^n$ 
    
```

Claim 1: h does not converge on g .

We show the claim by showing $\forall s : h(g_{s+1}) \neq h(g_s)$. As $\langle h, f \rangle \in \mathcal{TPcp}_{\underline{n}}$, we have for all $s \in \mathbb{N}$ and each $j \in \{0, 1\}$, $\lambda_i \leq n \cdot f(g_s \diamond \bar{j} \diamond \bar{0}^i)$ is not a $\underline{n}$ -path, as there is no path of length $n + 1$ in $\underline{n}$; hence, $\varphi_{h(g_s \diamond \bar{j} \diamond \bar{0}^n)}(\# \text{elets}(g_s)) = j$.

If now $h(g_s \diamond \bar{0} \diamond \bar{0}^n) = h(g_s)$, then $\varphi_{h(g_{s+1})}(\# \text{elets}(g_s)) = \varphi_{h(g_s \diamond \bar{1} \diamond \bar{0}^n)}(\# \text{elets}(g_s)) = 1 \neq \bar{0} = \varphi_{h(g_s \diamond \bar{0} \diamond \bar{0}^n)}(\# \text{elets}(g_s)) = \varphi_{h(g_s)}(\# \text{elets}(g_s))$; thus, $h(g_{s+1}) \neq h(g_s)$.

If $h(g_s \diamond \bar{0} \diamond \bar{0}^n) \neq h(g_s)$, then $h(g_{s+1}) = h(g_s \diamond \bar{0} \diamond \bar{0}^n) \neq h(g_s)$.

□ (OF CLAIM 1)

Claim 2: $g \in \mathbf{EXPF}$.

By the construction of g , we have $\forall s : g_s \in \{0, 1\}^{s \cdot (n+1)}$. Hence, to compute $g(x)$ for any given x , it suffices to execute stages 0 through x of the above algorithm to get g_{x+1} , from which $g(x)$ can then be extracted. Therefore, it suffices to show that, for all s , the stages 0 through s of the above algorithm can be done with an appropriate timebound.

Let p be a polynomial upper-bounding the runtime of h such that $\forall x : x \leq p(x)$. For any stage s , the time to execute stage s is in $\mathcal{O}(\lambda s \cdot p(|g_s \diamond \bar{0}^{n+1}|) + p(|g_s|)) = \mathcal{O}(\lambda s \cdot p(|g_s| + n + 1)) \stackrel{15}{=} \mathcal{O}(\lambda s \cdot p(s \cdot (n + 1) + n + 1)) = \mathcal{O}(\lambda s \cdot p(s))$. Therefore, for all s , the time to execute all stages 0 to s is bounded above by $\mathcal{O}(\lambda s \cdot (s + 1) \cdot p(s)) \subseteq \mathcal{O}(\lambda s \cdot 2^{p'(|s|)})$ for some polynomial p' .¹⁶

□ (OF CLAIM 2)

□ (OF (a))

¹⁵ $\mathcal{O}(|g_s|) = \mathcal{O}(\# \text{elets}(g_s))$.

¹⁶ Find k such that $\mathcal{O}(p) = \mathcal{O}(\lambda x \cdot x^k)$. By [RC94, §2.5, (9)], there are a, b such that $x \leq a \cdot 2^{|x|} + b$. Thus, there are c, d, c', d' such that $\forall x : p(x) \leq c \cdot x^k + d \leq c \cdot (a \cdot 2^{|x|} + b)^k + d \leq c' \cdot 2^{k \cdot |x|} + d'$.

4 General Results

4.1 Results Mostly Not Comparing Graphs

The following theorem shows the relationship between the different learning criteria as defined in this paper.

Theorem 6. We have the following.

$$\forall G \in \mathcal{G}_{\text{comp}} : \mathcal{TPcp}_G \mathbf{Ex} = \mathcal{TPcs}_G \mathbf{Ex}. \quad (8)$$

$$\mathcal{RPcs}_{\emptyset} \mathbf{Ex} \setminus \left(\bigcup_{G \in \mathcal{G}} \mathcal{Pcp}_G \mathbf{Ex} \right) \neq \emptyset. \quad (9)$$

$$\mathcal{RPcp}_{\emptyset} \mathbf{Ex} \setminus \left(\bigcup_{G \in \mathcal{G}} \mathcal{TPcp}_G \mathbf{Ex} \right) \neq \emptyset. \quad (10)$$

$$\mathcal{Pcp}_{\emptyset} \mathbf{Ex} \setminus \left(\bigcup_{G \in \mathcal{G}_{\text{comp}}} \mathcal{RPcs}_G \mathbf{Ex} \right) \neq \emptyset. \quad (11)$$

Furthermore, the separations (9) and (10) are witnessed by sets of functions such that the positive part of the separation is witnessed by a (fair) learner computable in linear time working transductively.

Our proof of (8) above is an extension of Fulk's proof of the $G = \emptyset$ case [Ful88].

*Proof of (10).*¹⁷ Let $\mathcal{S} := \{g \in \mathcal{R} \mid (\bar{0} \diamond (\pi_1 \circ g), g) \in \mathcal{Pcp}_{\emptyset} \mathbf{Ex}\}$. Obviously, $\mathcal{S} \in \mathbf{LinFTdPcp}_{\emptyset} \mathbf{Ex} \subseteq \mathcal{RPcp}_{\emptyset} \mathbf{Ex}$. Let $G \in \mathcal{G}$. Suppose, by way of contradiction, $\mathcal{S} \in \mathcal{TPcp}_G \mathbf{Ex}$ as witnessed by $\langle h_0, f_0 \rangle$. Define, for all $e \in \mathbb{N}$, $S_e := \{\sigma \mid \forall i < \#elets(\sigma) : \pi_1(\pi_1(\sigma(i))) = e\}$. Note that S_e is uniformly computable in e . By **KRT**, there is e such that φ_e is defined as the union over an infinite, with respect to sequence-extension strictly increasing, family of finite sequences $(\sigma_s)_{s \in \mathbb{N}}$ recursively specified as follows.

$$\sigma_0 := \emptyset; \quad (12)$$

$$\forall s : \sigma_{s+1} := \mu \sigma \in S_e \bullet \sigma_s \subset \sigma \wedge h_0(\sigma) \neq h_0(\sigma_s). \quad (13)$$

We reason by induction that, for all s , σ_s is defined. Clear for σ_0 . Let s be such that σ_s is defined. Let $\tau, \tau' \in S_e$ be two extensions of σ_s such that τ and τ' differ at position $\#elets(\sigma_s)$ (the first position not in σ_s), and $\#elets(\sigma_s)$ is in the bottomed-out set of f_0 after f_0 gets any one of the sequences τ or τ' as input. Then, as $\langle h_0, f_0 \rangle \in \mathcal{TPcp}_G$, $\varphi_{h_0(\tau)}(\#elets(\sigma_s)) = \tau(\#elets(\sigma_s)) \neq \tau'(\#elets(\sigma_s)) = \varphi_{h_0(\tau')}(\#elets(\sigma_s))$. Hence, for at least one $\sigma \in \{\tau, \tau'\}$, $h_0(\sigma) \neq h_0(\sigma_s)$.

We have a contradiction, as trivially $\varphi_e \in \mathcal{S}$ and $\langle h_0, f_0 \rangle$ does not $\mathcal{TPcp}_G \mathbf{Ex}$ -identify φ_e . $\square$ (FOR (10))

¹⁷ An anonymous referee pointed out that (10) can be proven by showing that all sets in $\mathcal{TPcp}_G \mathbf{Ex}$ can be reliably learned, as it is known that not all reliably learnable sets are $\mathcal{RPcp}_{\emptyset} \mathbf{Ex}$ -learnable [CJSW04]. We give this proof as a particularly short exemplar of many proofs omitted in Section 4.

Theorem 7. Let $G \in \mathcal{G}$. Then $\mathbf{TPcp}_G \mathbf{Ex}$ is closed under computably enumerable unions.

Our proof for Theorem 7 makes use of the notion of *reliability* [Min76, BB75].

Theorem 8. We have

$$\bigcup_{G \in \mathcal{G}} \mathbf{Pcs}_G \mathbf{Ex} \subset \mathbf{Ex}. \quad (14)$$

Furthermore, the separation is witnessed by a (fair) learner computable in linear time working transductively.

4.2 Dependencies on the Countdown Graphs

Next we define a pre-order, $\leq_{CD}$, on $\mathcal{G}$. We will see that $\leq_{CD}$ characterizes relative learning-power in dependence on countdown graphs.

Definition 9. For two graphs G, G' we write $G \leq_{CD} G'$ (read: G is countdown reducible to G') iff there is a $k \in \mathcal{R}$, such that

- (i) for all $y \in G$: $k(\vec{y}) \in G'$;
- (ii) for all $\tau \diamond \vec{y} \in \vec{G}$ such that $\#elets(\tau) > 0$, we have $k(\tau) \rightarrow_{G'} k(\tau \diamond \vec{y})$.

Intuitively, k maps any G -path into a vertex of G' .¹⁸ Clearly, $\leq_{CD}$ is a pre-order.

Proposition 10. Let $G, G' \in \mathcal{G}$. Let $k \in \mathcal{R}$. The following are equivalent.

- (a) $G \leq_{CD} G'$ as witnessed by k ;
- (b) $\forall \tau \in \vec{G} : (\lambda i < \#elets(\tau). k(\tau[i + 1])) \in \vec{G}'$.

Next we exhibit nice example countdown graphs and indicate how they compare by $\leq_{CD}$.

ω denotes the order-type of the natural numbers ordered by $\leq$, ω^{-1} denotes the order-type of the natural numbers ordered by $\geq$.

Theorem 11. There is a computable total ordering $\leq_R$ on $\mathbb{N}$ of order-type $\omega + \omega^{-1}$ such that there are no computable infinitely descending chains with respect to $\leq_R$; hence, $(\mathbb{N}, >_R)$ is a countdown graph.

For the rest of this section, let $\leq_R$ be as in Theorem 11, and let R denote the countdown graph $(\mathbb{N}, >_R)$.

Example 12. Let $(\mathcal{N}, \leq_{\mathcal{N}}), (\mathcal{N}', \leq_{\mathcal{N}'})$ be computably related systems of ordinal notations. Then we have

- (a) $\mathcal{N} \leq_{CD} \mathcal{N}' \Rightarrow \mathcal{N}'$ gives a notation to at least all the ordinals $\mathcal{N}$ gives a notation to;

¹⁸ Neither of mapping G vertices into G' vertices nor mapping G paths into G' paths will give us the same characterization results that we have in Theorem 13 below.

- (b) $\mathcal{N} \leq_{CD} R \Leftrightarrow \mathcal{N}$ gives a notation to all and only the ordinals $< \omega \cdot i + j$ for some $i \in \{0, 1\}, j \in \mathbb{N}$; and
 (c) $R \not\leq_{CD} \mathcal{N}$.

Theorem 13. Let $G \in \mathcal{G}_{\text{comp}}$, $G' \in \mathcal{G}$. We have

$$\mathbf{TPcp}_G \mathbf{Ex} \subseteq \mathbf{TPcp}_{G'} \mathbf{Ex} \Leftrightarrow G \leq_{CD} G'.$$

Next are three corollaries to Theorem 13 (or its proof). The first two are regarding the other restricted learnability notions of the present paper. The third is our hierarchy theorem for ordinal notations.

Corollary 14. Let $G \in \mathcal{G}_{\text{comp}}$. We have

$$\mathbf{TPcp}_G \mathbf{Ex} \setminus \bigcup_{\substack{G' \in \mathcal{G}_{\text{comp}} \\ G \not\leq_{CD} G'}} \mathbf{Pcs}_{G'} \mathbf{Ex} \neq \emptyset.$$

Next is a characterization of the graph dependence of relative learning power for the restricted learning criteria not covered by Theorem 13.

Corollary 15. For all $G, G' \in \mathcal{G}_{\text{comp}}$ we have

$$G \leq_{CD} G' \Leftrightarrow \mathbf{RPcp}_G \mathbf{Ex} \subseteq \mathbf{RPcp}_{G'} \mathbf{Ex} \quad (15)$$

$$\Leftrightarrow \mathbf{RPcs}_G \mathbf{Ex} \subseteq \mathbf{RPcs}_{G'} \mathbf{Ex} \quad (16)$$

$$\Leftrightarrow \mathbf{Pcp}_G \mathbf{Ex} \subseteq \mathbf{Pcp}_{G'} \mathbf{Ex} \quad (17)$$

$$\Leftrightarrow \mathbf{Pcs}_G \mathbf{Ex} \subseteq \mathbf{Pcs}_{G'} \mathbf{Ex}. \quad (18)$$

Recall that, from Section 2, for a graph $G \in \mathcal{G}$ and $m \in G$, we ambiguously use m to refer to the countdown-graph $\{n \in G \mid m \rightarrow^+ n\}$. For two sets M, N we write $M \# N$ iff $(M \not\subseteq N \wedge N \not\subseteq M)$.

Corollary 16. Let $(\mathcal{N}, \leq_{\mathcal{N}})$ be a computably related system of ordinal notations. Let $u, v \in \mathcal{N}$. Then we have

$$u <_{\mathcal{N}} v \Leftrightarrow u <_{CD} v \quad (19)$$

$$\Leftrightarrow \mathbf{TPcp}_u \mathbf{Ex} \subset \mathbf{TPcp}_v \mathbf{Ex}. \quad (20)$$

Furthermore, if $\mathcal{N}$ gives a notation to at least all ordinals $< \omega \cdot 2$, then

$$\mathbf{TPcp}_{\mathcal{N}} \mathbf{Ex} \# \mathbf{TPcp}_R \mathbf{Ex}. \quad (21)$$

References

- [ACJS04] Ambainis, A., Case, J., Jain, S., Suraj, M.: Parsimony hierarchies for inductive inference. *Journal of Symbolic Logic* 69, 287–328 (2004)
 [AZ07] Akama, Y., Zeugmann, T.: Consistent and coherent learning with δ -delay. Technical Report TCS-TR-A-07-29, Hokkaido Univ. (October 2007)

- [Bär74] Bärzdīņš, J.: Inductive inference of automata, functions and programs. In: Int. Math. Congress, Vancouver, pp. 771–776 (1974)
- [BB75] Blum, L., Blum, M.: Toward a mathematical theory of inductive inference. *Information and Control* 28, 125–155 (1975)
- [Cas74] Case, J.: Periodicity in generations of automata. *Mathematical Systems Theory* 8, 15–32 (1974)
- [CJSW04] Case, J., Jain, S., Stephan, F., Wiehagen, R.: Robust learning – rich and poor. *Journal of Computer and System Sciences* 69, 123–165 (2004)
- [CK08] Case, J., Kötzing, T.: Dynamically delayed postdictive completeness and consistency in machine inductive inference (2008), <http://www.cis.udel.edu/~case/papers/PCpPcsDelayTR.pdf>
- [CKP07] Case, J., Kötzing, T., Paddock, T.: Feasible iteration of feasible learning functionals. In: Hutter, M., Servadio, R.A., Takimoto, E. (eds.) ALT 2007. LNCS (LNAI), vol. 4754, pp. 34–48. Springer, Heidelberg (2007)
- [FS93] Freivalds, R., Smith, C.: On the role of procrastination in machine learning. *Information and Computation* 107(2), 237–271 (1993)
- [Ful88] Fulk, M.: Saving the phenomena: Requirements that inductive machines not contradict known data. *Inform. and Comp.* 79, 193–209 (1988)
- [JORS99] Jain, S., Osherson, D., Royer, J., Sharma, A.: *Systems that Learn: An Introduction to Learning Theory*, 2nd edn. MIT Press, Cambridge (1999)
- [LV97] Li, M., Vitanyi, P.: *An Introduction to Kolmogorov Complexity and Its Applications*, 2nd edn. Springer, Heidelberg (1997)
- [Min76] Minicozzi, E.: Some natural properties of strong identification in inductive inference. In: *Theoretical Computer Science*, pp. 345–360 (1976)
- [Pit89] Pitt, L.: Inductive inference, DFAs, and computational complexity. In: Jantke, K.P. (ed.) AII 1989. LNCS, vol. 397, pp. 18–44. Springer, Heidelberg (1989)
- [Rog67] Rogers, H.: *Theory of Recursive Functions and Effective Computability*. McGraw Hill, New York (1967) (Reprinted by MIT Press, Cambridge, Massachusetts, 1987)
- [RC94] Royer, J., Case, J.: Subrecursive Programming Systems. In: *Progress in Theoretical Computer Science*, Birkhäuser (1994)
- [SSV04] Sharma, A., Stephan, F., Ventsov, Y.: Generalized notions of mind change complexity. *Information and Computation* 189, 235–262 (2004)
- [Wie76] Wiehagen, R.: Limes-erkennung rekursiver Funktionen durch spezielle Strategien. *Elek. Informationsverarbeitung und Kyb.* 12, 93–99 (1976)
- [Wie78] Wiehagen, R.: *Zur Theorie der Algorithmischen Erkennung*. Dissertation B. Humboldt University of Berlin (1978)

Dynamic Modeling in Inductive Inference

John Case and Timo Kötzing*

Department of Computer and Information Sciences,
University of Delaware, Newark, DE 19716-2586, USA
{case,koetzing}@cis.udel.edu

Abstract. Introduced is a new inductive inference paradigm, *Dynamic Modeling*. Within this learning paradigm, for *example*, function h *learns* function g iff, in the i -th iteration, h and g both produce output, h gets the sequence of all outputs from g in prior iterations as input, g gets all the outputs from h in prior iterations as input, and, from some iteration on, the sequence of h 's outputs will be *programs* for the *output sequence* of g .

Dynamic Modeling provides an idealization of, for example, a social interaction in which h seeks to discover program models of g 's behavior it sees in interacting with g , and h *openly* discloses to g its sequence of candidate program models to see what g says back.

Sample results: every g can be so learned by some h ; there are g that can only be learned by an h if g can also learn that h back; there are extremely secretive h which cannot be learned back by any g they learn, but which, nonetheless, succeed in learning infinitely many g ; quadratic-time learnability is strictly more powerful than linear-time learnability.

This latter result, as well as others, follow immediately from general correspondence theorems obtained from a *unified* approach to the paradigms within inductive inference.

Many proofs, some sophisticated, employ machine self-reference, a.k.a., recursion theorems.

1 Introduction and Motivation

In **Computational Limit Learning of Computable Functions** mapping the non-negative integers to same, an *algorithmic learner* is iteratively given more and more *finite* information generated by a *computable target function*. From this information, the learner, in each iteration, (may) synthesize a (suitably interpreted) natural number as output. In the literature, many criteria of successful learning have been proposed. Each such learning criterion defines precisely, possibly among other things, in what way the information will be generated by the target function and how the sequence of iteratively generated outputs and the target function have to relate for the learning to be considered successful. Sometimes each output number will be interpreted as (numerically naming) a program, other times each

* The authors would like to thank Samuel E. Moelius III for various fruitful discussions, especially on reactive learners.

number will represent a prediction for a yet unseen data point. It is helpful for the present paper to briefly consider some illustrative examples.

Ex-learning [Gol67] exemplifies the category of *identification*. h Ex-learns target g iff, in the i -th iteration, h outputs a conjecture on input $g(0) \dots g(i-1)$ and there are j, e such that $\forall k \geq j : h(g(0) \dots g(k-1)) = e$ and e is a program for g .¹ Such a program e could be carried away and used *offline*.

Nv-learning [Bär71, BB75] exemplifies the category of *extrapolation*. h Nv-learns target g iff, h is total and, in the i -th iteration, h outputs a conjecture on input $g(0) \dots g(i-1)$ and there is a j such that $\forall k \geq j : h(g(0) \dots g(k-1)) = g(k)$.² The successful extrapolants $h(g(0) \dots g(k-1))$, $k \geq j$, can be used *online*.

Coord-learning [MO99] exemplifies the category of *coordination*. h Coord-learns a target g iff, in the i -th iteration, h and g both produce output, h gets the sequence of all outputs from g in prior iterations as input, g gets all the outputs from h in prior iterations as input, and, from some iteration on, the sequence of h 's outputs will be the same as the sequence of g 's outputs. The finally successfully coordinated matching outputs can be used *online*.

We see above that, while Coord-learning features a *reactive learner* g , Ex- and Nv-learning feature a *passive learner* g .

	Passive Learner	Reactive Learner
Off-Line	Identification	??
On-Line	Extrapolation	Coordination

In the just above table, there is a missing, not heretofore studied category entry for *offline* with *reactive learner*. We refer to this category as *dynamic modeling*, and it is the subject of the present paper. **XBc**-learning exemplifies the category of *dynamic modeling*. h **XBc**-learns a target g iff, in the i -th iteration, h and g both produce output, h gets the sequence of all outputs from g in prior iterations as input, g gets all the outputs from h in prior iterations as input, and, from some iteration on, the sequence of h 's outputs will be *programs* for the *output sequence* of g .³

In cognitive science, *theory of mind* refers to ones having a model (or models) of another's thoughts, emotions, and perspectives — including those different from ones own. Ideally, one might have a *program* (or programs) generating the behavior of the other, but — the behavior presented by the other would, in reality, be all and only that resulting from crossfeeding between oneself and the other. While one is attempting to synthesize program(s) for the other, a technique to employ

¹ The term 'Ex' stands for *explanatory* [CS83].

² The term 'Nv' stands for *next value* [Bär71].

³ **X** we pronounce *cross*, and it is short for *crossfeed*. Of course crossfeeding of data is common to both the categories of coordination and dynamic modeling. In Section 3 we use the **X** also in talking about the former category. **Bc** stands for *behaviorally correct* [CS83].

is to pass on a sequence of remarks such as, “I think you are like . . .,” (where . . . might be a program), and, then, attend to the resultantly elicited sequence of reactions of the other — as one formulates further programs/models of the other. Of course, in reality, one might, in seeking social understanding, carry out variants, including highly filtered variants, of the just above scenario. The unfiltered and very idealized scenario above is, nonetheless, covered by dynamic modeling.

Next we summarize the contents of the remaining sections.

In Section 2 below we present mathematical preliminaries.

Section 3 presents a *unified* approach to limiting learning criteria. This pays off in Section 5 where we can *then* provide general results applying to many criteria at once and, thereby, *quickly* obtain some nice corollaries.⁴

Section 4 involves cooperativeness vs. secretiveness in dynamic modeling. Considered are dynamic modelers which may or may not, in return, be dynamically modeled themselves. Proposition 4 implies that *no* computable g can keep models of its behavior totally secret; moreover, for any computable g , there are infinitely many constant functions h that **XBc**-learn g .⁵

Surprisingly, Theorem 6 implies that there is a computable g so that, *no* computable h that **XBc**-learns g can keep models of its behavior a secret from g , i.e., such h gives itself away: g , in turn, **XBc**-learns h . Positively, such a g is, then, *extremely cooperative*: informally, g can figure out the behavior of any computable h that figures out its behavior. Furthermore, such a g can be chosen to be linter-time computable! The proof (included) of Theorem 6 is particularly elegant.

We say that computable h is *extremely uncooperative* iff $\{\text{computable } g \mid h \text{ XBc-learns } g \wedge g \text{ XBc-learns } h\} = \emptyset$. Theorem 10 implies there are extremely uncooperative computable h which, nonetheless, are infinitely successful, i.e., such that h **XBc**-learns infinitely many computable g .

Results in Section 4 feature open disclosure of certain learners’ models of another while not disclosing their own models to the other. For comparison and contrast, a *zero-knowledge proof* [BSMP91] permits open, convincing disclosure of its existence without disclosing how it works.

Section 5 features two general and powerful correspondence theorems (Theorems 17 and 19) regarding many of the criteria discussed above and in Section 3.

The first *immediately* yields Corollary 18 which implies, for *example*, that quadratic-time **XBc**-learnability is strictly more powerful than linter-time **XBc**-learnability.⁶

⁴ In Section 3 Ex-learning will be called **GEx**-learning, where the **G** is for Gold [Gol67]. Nv-learning will be called **RGM**, where the **R** is for (total) computable learner and learner, and the **M** is for Matching. Coord-learning will be called **XM**-learning.

⁵ Actually, Proposition 4 is stated for a more restrictive criterion within the dynamic modeling category.

⁶ Nothing like this happens for, for example, Ex-learning, since by an extension of Pitt’s postponement tricks from [Pit89], (otherwise unrestricted) Ex-learning with linter-time learners is just as powerful as Ex-learning. As we’ll see from Theorem 14, also in Section 5, such extended postponement tricks *do*, nonetheless, apply to (even a restricted special case of) the **XBc**-learning of *any* computably enumerable set of computable functions.

Theorem 19 *immediately* yields Corollary 20 which provides a number of learning criteria hierarchies and separations. An *example*: the powers of **XBc**-learning and of Coord-learning are incomparable.

Many proofs are left out because of space constraints, and many proofs, some sophisticated, employ machine self-reference techniques, including Kleene Recursion Theorem (**krt**) [Rog67, page 214, problem 11-4] and Case's Operator Recursion Theorem (**ort**) [Cas74, Cas94]. The latter achieves infinitary self (and other) reference. See www.cis.udel.edu/~case/papers/DynamicModelingTR.pdf for a more complete version of the present paper.

2 Mathematical Preliminaries

Any unexplained complexity-theoretic notions are from [RC94]. All unexplained general computability-theoretic notions are from [Rog67].

$\mathbb{N}$ denotes the set of natural numbers, $\{0, 1, 2, \dots\}$. $*$ is a symbol such that, for all $n \in \mathbb{N}$, $n < *$. For two functions f, g and $n \in \mathbb{N}$, $f =^n g$ means that f and g differ on at most n arguments, $f =^* g$ means that f and g differ only on finitely many arguments.

The symbols $\subseteq, \subset, \supseteq, \supset$ respectively denote the subset, proper subset, superset and proper superset relation between sets. $\mathfrak{P}$ and $\mathfrak{R}$, respectively, denote the set of all partial and total functions $\mathbb{N} \rightarrow \mathbb{N}$, respectively.

The quantifier $\forall^\infty x$ means “for all but finitely many x ”, the quantifier $\exists^\infty x$ means “for infinitely many x ”.

We sometimes denote a function f of $n > 0$ arguments $x_1, \dots, x_n$ in lambda notation (as in Lisp) as $\lambda x_1, \dots, x_n. f(x_1, \dots, x_n)$. For example, with $c \in \mathbb{N}$, $\lambda x. c$ is the constantly c function of one argument.

A function ψ is *partial computable* iff there is a Turing machine computing ψ . $\mathcal{R}$ and $\mathcal{P}$ denote the set of all (total) computable and partial computable functions $\mathbb{N} \rightarrow \mathbb{N}$, respectively. If ψ is not defined for some argument x , then we denote this fact by $\psi(x) \uparrow$, and we say that ψ on x *diverges*. The opposite is denoted by $\psi(x) \downarrow$, and we say that ψ on x *converges*.

We say that a partial function ψ *converges to p* iff $\forall^\infty x : \psi(x) \downarrow = p$.

[RC94, §3] describes an *efficiently* numerically named or coded⁷ programming system for multi-tape Turing machines (TMs) which compute the partial computable functions $\mathbb{N} \rightarrow \mathbb{N}$. Herein we name this system φ . φ_p denotes the partial computable function computed by the TM-program with code number p in the φ -system, and Φ_p denotes the partial computable *runtime* function of the TM-program with code number p in the φ -system. By [RC94, Lemma 3.13], s-m-n, which provides for algorithmic storage of data in programs, has a *lintime* instance in this system. Hence, it can be shown that this system also has a lintime instance of 1-1 **ort**.

⁷ This numerical coding guarantees that many simple operations involving the coding run in linear time. This is by contrast with historically more typical codings featuring prime powers and corresponding at least exponential costs to do simple things.

Whenever we consider tuples of natural numbers as input to functions, it is understood that a fixed *efficient* pairing function $\langle \cdot, \cdot \rangle$ (as in [RC94]) is used to code (left-associatively) the tuples into a single natural number.

A *finite sequence* is a mapping with a finite initial segment of $\mathbb{N}$ as domain (and range, $\mathbb{N}$). $\emptyset$ denotes the empty sequence (and, also, the empty set). The set of all finite sequences is denoted by Seq . For each finite sequence σ , we will denote the first element, if any, of that sequence by $\sigma(0)$, the second, if any, by $\sigma(1)$ and so on. $\#elets(\sigma)$ denotes the number of elements in a finite sequence σ , that is, the cardinality of its domain. $\text{last}(\sigma)$ denotes the last element of σ (if any).

For a partial function g and $i \in \mathbb{N}$, if $\forall j < i : g(j) \downarrow$, then $g[i]$ is defined to be the finite sequence $g(0), \dots, g(i-1)$.

We take $\mathbb{N} = \{0, 1\}^*$ for ease of measurement of the size of each natural number. Following [LV97], we fix an easily computed and inverted coding of all finite sequences of natural numbers into $\mathbb{N}$ so that the size of any sequence, defined as the size of its coding, is sensible for measuring the computational complexity of functions which take sequences as inputs. In particular the size of any sequence σ should be linear in $\#elets(\sigma)$ and the size of each natural number in σ .

We let **LinF**, **PF** and **EXPF** be the set of all lintime, polytime and exptime computable functions, respectively.

Henceforth, we will many times identify a finite sequence σ with its code number $\langle \sigma \rangle_{\text{seq}}$. However, when we employ expressions such as $\sigma(x)$, $\sigma = f$ and $\sigma \subset f$, we consider σ as a *sequence*, not as a number.

There are 1-1 function pad and function unpad_1 , both $\in \mathbf{LinF}$ and such that for all n, e : $\varphi_{\text{pad}(n,e)} = \varphi_e$ and $\text{unpad}_1(\text{pad}(n, e)) = n$.

3 Unified Approach to Limit Learning Criteria

In this section, in the interest of generality, we give many definitions for limit learning also involving *non*-algorithmic learning. Nonetheless, all of the *results* given in the present paper concern only *algorithmic* (including complexity bounded) learning.

3.1 Definitions

Any set $\mathcal{C} \subseteq \mathfrak{P}$ is a *learner admissibility restriction*; intuitively, a learner admissibility restriction defines which functions are admissible as potential learners, e.g., $\mathcal{P}, \mathcal{R}, \mathbf{LinF}, \dots$.

Every computable operator $\mathfrak{P} \times \mathfrak{R} \rightarrow \mathfrak{P}^2$ is called a *sequence generating operator*; intuitively, a sequence generating operator defines how learner and learnee interact to generate two infinite sequences, one for learner-outputs (we call this sequence the *learner-sequence*) and one for learnee-outputs.⁸ For example, for

⁸ Essentially, *these* computable operators are the recursive operators of [Rog67] but with two arguments and two outputs and restricted to the indicated domain.

Ex-learning (to be renamed in Section 3.2), for a (then, passive) learner g , its learner outputs would be $g(0), g(1), \dots$. Below, in general, even for reactive learners, we refer to the sequence of learner-outputs as a *data-sequence*.

For $h \in \mathcal{P}$, $g \in \mathcal{R}$ and β a sequence generating operator, we call the first component of $\beta(h, g)$ the *learner-sequence* of h given g (denoted by $\beta_1(h, g)$), the second the *data-sequence* of g given h (denoted by $\beta_2(h, g)$).

Every subset of $\mathfrak{P}^2$ is called a *sequence acceptance criterion*. Intuitively, a sequence acceptance criterion defines which learner-sequences are considered a successful learning of a data-sequence. Any two such sequence acceptance criteria δ and δ' can be combined by intersecting them. For ease of notation we write $\delta\delta'$ instead of $\delta \cap \delta'$.

A *learning criterion* (or short *criterion*) is a 3-tuple consisting of a learner admissibility restriction, a sequence generating operator and a sequence acceptance criterion. Let $\mathcal{C}$, β , δ , respectively, be a learner admissibility restriction, a sequence generating operator and a sequence acceptance criterion, respectively. For $h \in \mathfrak{P}$, $g \in \mathfrak{R}$ we say that h ($\mathcal{C}, \beta, \delta$)-learns g iff $h \in \mathcal{C}$ and $\beta(h, g) \in \delta$. For $h \in \mathfrak{P}$ and $\mathcal{S} \subseteq \mathfrak{R}$ we say that h ($\mathcal{C}, \beta, \delta$)-learns $\mathcal{S}$ iff, for all $g \in \mathcal{S}$, h ($\mathcal{C}, \beta, \delta$)-learns g . The set of ($\mathcal{C}, \beta, \delta$)-learnable sets of computable functions is

$$\mathcal{C}\beta\delta := \{\mathcal{S} \subseteq \mathcal{R} \mid \exists h \in \mathfrak{P} : h \text{ } (\mathcal{C}, \beta, \delta)\text{-learns } \mathcal{S}\}. \quad (1)$$

We refer to the sets $\mathcal{C}\beta\delta$ as in (1) as *learnability classes*. Instead of writing the tuple $(\mathcal{C}, \beta, \delta)$, we will ambiguously write $\mathcal{C}\beta\delta$. For $h \in \mathfrak{P}$, the set of all computable learners ($\mathcal{C}, \beta, \delta$)-learned by h is denoted by $\mathcal{C}\beta\delta(h) := \{g \in \mathcal{R} \mid h \text{ } (\mathcal{C}, \beta, \delta)\text{-learns } g\}$.

For any sequence generating operator β , we can turn a given sequence acceptance criterion δ into a learner admissibility restriction $\mathcal{T}_\beta\delta$ by admitting only those learners that obey δ on all possible input:

$$\mathcal{T}_\beta\delta := \{h \in \mathfrak{P} \mid \forall g \in \mathfrak{R} : \beta(h, g) \in \delta\}.$$

3.2 Examples

In this section we give many examples illustrating our definitions and give an overview as to how our notation covers criteria from the literature. Past this section, we will not be concerned with *every* example given in this section, but some of them will be employed.

Example 1. Two typical learner admissibility restrictions are $\mathcal{P}$ and $\mathcal{R}$. Furthermore, any set of functions computable with a resource restriction (such as the set of all ltime computable functions) may be used as a learner admissibility restriction. For each sequence generating operator β and each $\mathfrak{F} \subseteq \mathfrak{R}$, the set $\text{Rel}_{\beta, \mathfrak{F}}$ of all functions reliable on $\mathfrak{F}$ [Min76, BB75] (defined below) is also a learner admissibility restriction.

$$\begin{aligned} \text{Rel}_{\beta, \mathfrak{F}} := \{h \in \mathcal{P} \mid & \forall g \in \mathfrak{F} \forall p, q \in \mathcal{P} : (\beta(h, g) = (p, q) \\ & \wedge p \text{ converges to some } p') \Rightarrow \varphi_{p'} = q\}. \end{aligned}$$

When denoting criteria with $\mathcal{P}$ as the learner admissibility restriction, we will omit $\mathcal{P}$.

Example 2. We define the following example sequence generating operators. All learners give an initial conjecture, say, of 0, based on no data.

- *Goldstyle* [Gol67]: $\mathbf{G} : \mathfrak{P} \times \mathfrak{R} \rightarrow \mathfrak{P} \times \mathfrak{R}, (h, g) \mapsto (\lambda i. h(g[i]), g)$.
- *Iterative* [Wie76]: $\mathbf{It} : \mathfrak{P} \times \mathfrak{R} \rightarrow \mathfrak{P} \times \mathfrak{R}, (h, g) \mapsto (p, q)$ such that $p(0) = 0$, $\forall n : p(n+1) = h(q(n), p(n))$ and $q = g$.
- *Transductive*: $\mathbf{Td} : \mathfrak{P} \times \mathfrak{R} \rightarrow \mathfrak{P} \times \mathfrak{R}, (h, g) \mapsto (p, q)$ such that $p(0) = 0$, $\forall n : p(n+1) = h(q(n))$ and $q = g$.
- *Crossfeeding* [MO99]: $\mathbf{X} : \mathfrak{P} \times \mathfrak{R} \rightarrow \mathfrak{P} \times \mathfrak{P}, (h, g) \mapsto (p, q)$ such that $\forall n \, p(n) = h(q[n]) \wedge q(n) = g(p[n])$.
- *Learnee Iterative*: $\mathbf{Li} : \mathfrak{P} \times \mathfrak{R} \rightarrow \mathfrak{P} \times \mathfrak{P}, (h, g) \mapsto (p, q)$ such that $\forall n \, p(n) = h(q[n]) \wedge q(0) = 0 \wedge \forall n : q(n+1) = g(p(n), q(n))$.

“ $\mathbf{G}$ ” is a reference to Gold [Gol67]. Intuitively, $\mathbf{G}$ takes a learner h and a learner g , and feeds longer and longer initial segments of g into h , considering the successive outputs as coding an infinite sequence of hypotheses. The second output is just g , meaning that the target concept to be learned is all of g . In this setting, the learner gets a lot of information about the learner, while the learner does not react at all to the learning process. For $\mathbf{It}$ and $\mathbf{Td}$ defined above, a learner for the latter has less information at its disposal than for the former.

Regarding $\mathbf{X}$, learner and learner have symmetrical information in each iteration. $\mathbf{Li}$ lessens the information that the learner has in a similar way that iterative learning lessens the information of the learner.

The first three bullets given in Example 2 involve passive learners, while the last two examples involve reactive learners.

We note the following three important properties relating $\mathbf{G}$ and $\mathbf{X}$, which are of importance to this paper. Let $f, g, h, p, q \in \mathcal{P}$.

$$\mathbf{X}(h, g) = (p, q) \Rightarrow \mathbf{G}(h, q) = (p, q). \quad (2)$$

$$\mathbf{X}(h, g) = (p, q) \Rightarrow \mathbf{G}(g, p) = (q, p). \quad (3)$$

$$\mathbf{X}(h, \lambda \sigma. f(\# \text{elets}(\sigma))) = \mathbf{G}(h, f). \quad (4)$$

Example 3. We define the following sequence acceptance criteria.

- *Explanatory*: $\mathbf{Ex} = \{(p, q) \in \mathfrak{P}^2 \mid p \text{ converges to some } p' \text{ and } \varphi_{p'} = q\}$.
- *Explanatory with up to a* $a \in \mathbb{N} \cup \{*\}$ errors [CS83, BB75]:
 $\mathbf{Ex}^a = \{(p, q) \in \mathfrak{P}^2 \mid p \text{ converges to some } p' \text{ and } \varphi_{p'} =^a q\}$.
- *Behaviorally correct* [CS83, Bär74b]: $\mathbf{Bc} = \{(p, q) \in \mathfrak{P}^2 \mid \forall^\infty n \, \varphi_{p(n)} = q\}$.
- *Behaviorally correct with up to a* $a \in \mathbb{N} \cup \{*\}$ errors [CS83]:
 $\mathbf{Bc}^a = \{(p, q) \in \mathfrak{P}^2 \mid \forall^\infty n \, \varphi_{p(n)} =^a q\}$.
- *Matching* [Bär71, BB75, MO99]: $\mathbf{M} = \{(p, q) \in \mathfrak{P} \times \mathfrak{R} \mid p =^* q\}$.

All the above criteria include global restrictions on the path to successful learning. The following defines several criteria only involving local restrictions.

- *Postdictively complete* [Bār74a, BB75, Wie76]: $\mathbf{Pcp} = \{(p, q) \in \mathfrak{R}^2 \mid \forall n \forall i < n : \varphi_{p(n)}(i) = q(i)\}$.
- *Hypotheses are programs for total functions* [CS83]: $\mathbf{T} = \{(p, q) \in \mathfrak{R}^2 \mid \forall n : \varphi_{p(n)} \in \mathcal{R}\}$.
- *Always giving hypotheses*: $\mathfrak{R}^2$.

The idea of dividing a learning criterion is not entirely new. For example, Freivalds et. al. [FKS95] defined *admissible sequences for a given function*, which basically defines a binary predicate on a pair of infinite sequences.

We can now express several learning criteria as defined in the prior literature (left-hand-side below) with our notation system (right-hand-side below). Recall that the default learner admissibility restriction is $\mathcal{P}$; hence, all learning criteria displayed just below are for *algorithmic* learners.

$$\begin{aligned}
& \text{Ex} \leftrightarrow \mathbf{GEx} \\
& \text{Bc} \leftrightarrow \mathbf{GBc} \\
& \text{Nv} \leftrightarrow \mathcal{R}\mathbf{GM} \\
& \text{Nv}' \leftrightarrow \mathbf{GR}^2\mathbf{M} \\
& \text{Nv}'' \text{ ([Pod74])} \leftrightarrow \mathbf{GM} \\
& \text{Cons} \leftrightarrow \mathbf{GPcpEx} \\
& \text{R-Cons} \leftrightarrow \mathcal{R}\mathbf{GPcpEx} \\
& \text{T-Cons} \leftrightarrow \mathcal{T}_{\mathbf{G}}\mathbf{PcpGEx} \\
& \text{Reliable on } \mathcal{R} \leftrightarrow \text{Rel}_{\mathcal{R}}\mathbf{GEx} \\
& \text{It} \leftrightarrow \mathbf{ItEx} \\
& \text{learnable by a player ([MO99])} \leftrightarrow \mathbf{XM} \\
& \text{learnable by a total player ([MO99])} \leftrightarrow \mathcal{R}\mathbf{XM}
\end{aligned}$$

A sequence acceptance criterion δ is said to be *degenerate* iff $\exists(p, q) \in \delta : p =^* \lambda x. \uparrow$. All sequence acceptance criteria given above are non-degenerate, and the authors don't know of any degenerate sequence acceptance criteria implicit in the prior literature. We conjecture that any degenerate sequence acceptance criteria would be useless to model learning. Hence, the present paper will solely focus on non-degenerate such criteria.

It is easy to see that, for non-degenerate δ , we have for all $\mathcal{C} \subseteq \mathcal{P}$ that the learnability classes $\mathcal{C}\mathbf{X}\delta$ and $\mathcal{C}\mathbf{X}\mathcal{R}^2\delta$ are *equal*.

Similarities between extrapolation (like $\mathbf{GM}$) and coordination (like $\mathbf{XM}$) have been pointed out in [CJM⁺05]. In particular, *blind* learners are defined as functions where each output only depends on the *length* of it's input, and with each function $g \in \mathcal{R}$, a blind learner $g' = \lambda \sigma. g(\# \text{elems}(\sigma))$ is associated. The mapping $\Theta = \lambda g. g'$ is then a natural embedding of learners in the $\mathbf{G}$ -sense to learners in the $\mathbf{X}$ -sense, more formally, for all $\delta \in \mathcal{P} \times \mathcal{R}$ and $\mathcal{S} \subseteq \mathcal{R}$,

$$\mathcal{S} \in \mathbf{G}\delta \Leftrightarrow \Theta(\mathcal{S}) \in \mathbf{X}\delta. \quad (5)$$

The special case of (5) with $\delta = \mathbf{M}$ is used in [CJM⁺05].

4 Cooperation and Secretiveness

The main emphasis of the present paper, as seen in Section 1, features $\mathbf{XBc}$ -learning, but, based on the thinking of Section 3, one might wonder why we didn't

talk about **XEx**-learning. We'll talk about it now. Suppose h **XEx**-learns g . The learner-sequence of h interacting with g , is, then, a total, almost everywhere constant function. Suitable g , then, easily **XEx**-learn h .⁹

The proposition just below implies that, *any* computable g gets **XBc**-learned by some computable h . Hence, any g can have its secrets learned by some learner. The interesting thing, then, is whether, when h **XBc**-learns g , h can keep models of itself secret from g . This is considered in Theorem 6 further below.

Proposition 4. Let $g \in \mathcal{R}$. Then there are infinitely many (total) constant functions h **XBc**- (in fact, **XEx**-) learning g .

Proof. Let $n \in \mathbb{N}$. There is, by **krt**, e_n such that, with

$$p = \lambda x. \text{pad}(n, e_n), \quad (7)$$

$$\forall x : \varphi_{e_n}(x) = g(p[x]). \quad (8)$$

Let $h_n \in \mathcal{R}$ such that

$$\forall \sigma : h_n(\sigma) = \text{pad}(n, e_n). \quad (9)$$

There is $q \in \mathcal{R}$ such that $\mathbf{X}(h_n, g) = (p, q)$. Then we have, for all t and x , we have

$$\varphi_{p(t)}(x) \stackrel{(7)}{=} \varphi_{\text{pad}(n, e_n)}(x) = \varphi_{e_n}(x) \stackrel{(8)}{=} g(p[x]) \stackrel{\text{choice of } q}{=} q(x). \quad (10)$$

Hence, h_n **XBc**-learns g . Trivially, we have for all $l \neq m$, $h_l \neq h_m$. This shows that there are infinitely many different (constant) functions **XEx**-learning g . $\square$

Definition 5. We define the following sequence acceptance criteria.

- Cooperative Bc: $\mathbf{Bcc} = \{(p, q) \in \mathcal{R}^2 \mid \forall^\infty n \varphi_{p(n)} = q \wedge \forall^\infty n \varphi_{q(n)} = p\}$
(= $\mathbf{BcBc}^{-1}$).
- Secretive Bc: $\mathbf{Bcs} = \{(p, q) \in \mathcal{R}^2 \mid \forall^\infty n \varphi_{p(n)} = q \wedge \neg \forall^\infty n \varphi_{q(n)} = p\}$
(= $\mathbf{Bc}\overline{\mathbf{Bc}^{-1}}$).

Clearly, for all $h, g \in \mathcal{R}$, h **XBcc**-learns g iff, h **XBc**-learns g and g **XBc**-learns h ; similarly, h **XBcs**-learns g iff, h **XBc**-learns g and g does *not* **XBc**-learn h .

It is easy to see that there are computable functions which are not **XBcc**-learnable, for example $\lambda \sigma. \# \text{elets}(\sigma)$.¹⁰ At first glance, it seems likely that all computable functions can be **XBcs**-learned, as, by Proposition 4 above, for

⁹ We have, more generally, that

$$\exists g \in \mathcal{R} \forall h \in \mathcal{R} : h \text{ **XEx**-learns } g \Rightarrow g \text{ **XEx**-learns } h. \quad (6)$$

¹⁰ For all $g \in \mathcal{R}$, and $p, q \in \mathcal{R}$ such that $\mathbf{X}(\lambda \sigma. \# \text{elets}(\sigma), g) = (p, q)$, we have that p is the identity on $\mathbb{N}$; hence, $\lambda \sigma. \# \text{elets}(\sigma)$ does not **XBc**-learn g .

any given function g , there are infinitely many functions h **XBc**-learning g . We were, then, surprised that not all computable functions can be **XBcs**-learned, as seen below in Theorem 6. Intuitively, this theorem means that there is a $g \in \mathcal{R}$ such that, for all $h \in \mathcal{P}$, if h **XBc**-learns g , then h has to give away enough information about itself so that g will be able to **XBc**-learn h . Even more surprisingly, such a g can be chosen to be linterme computable! On the other hand, Theorem 6 also has a positive interpretation: it *is* possible to find a function g that will **XBc**-learn every function h that **XBc**-learns g – in other words, there are *extremely cooperative functions* that will cooperate with *any* function **XBc**-learning them. We denote the set of extremely cooperative functions with $EC := \{h \in \mathcal{R} \mid \forall g \in \mathcal{R} : g \text{ XBc-learns } h \Rightarrow h \text{ XBc-learns } g\}$.¹¹

Theorem 6 (Secretiveness Fails)

$$\exists g \in \mathbf{LinF} : \{g\} \notin \mathbf{XBcs}.$$

Proof. By 1-1, linterme **ort** there is 1-1 $g \in \mathbf{LinF}$ such that

$$\forall \tau, x : \varphi_{g(\tau)}(x) = \text{last}(g^{-1}(\varphi_{\text{last}(\tau)}(x+1))).^{12} \quad (11)$$

Let $h \in \mathcal{P}$ be such that $g \in \mathbf{XBc}(h)$. Let $p, q \in \mathcal{R}$ be such that $\mathbf{X}(h, g) = (p, q)$. Since $(p, q) \in \mathbf{Bc}$, there is n_0 such that

$$\forall n \geq n_0 : \varphi_{p(n)} = q. \quad (12)$$

Claim: $\forall^\infty n : \varphi_{q(n)} = p$.

Proof. We have

$$\forall n \geq n_0 + 1, x : \varphi_{p(n-1)}(x+1) \stackrel{(12)}{=} q(x+1) \stackrel{\text{choice of } q}{=} g(p[x+1]). \quad (13)$$

Hence, for all $n \geq n_0 + 1$ and all x ,

$$\begin{aligned} \varphi_{q(n)}(x) &\stackrel{\text{choice of } q}{=} \varphi_{g(p[n])}(x) \stackrel{(11)}{=} \text{last}(g^{-1}(\varphi_{p(n-1)}(x+1))) \\ &\stackrel{(13)}{=} \text{last}(g^{-1}(g(p[x+1]))) = \text{last}(p[x+1]) = p(x). \end{aligned} \quad (14)$$

□ (FOR CLAIM)

Hence, by the claim, $g \in \mathbf{XBcc}(h)$; therefore, $\{g\} \notin \mathbf{XBcs}$.

□ (FOR THEOREM)

[MO99] examined uncooperativeness of coordinators. In particular, two sets of total computable functions are constructed such that any learner learning *all* of the functions from one of the sets cannot coordinate with *any* function

¹¹ It is easy to see that $EC = \{h \in \mathcal{R} \mid \mathbf{XBcs}^{-1}(h) = \emptyset\}$.

¹² Note that $\varphi_{g(\tau)}(x)$ might be undefined for various reasons, for example last is not total. Furthermore, note that accessing g^{-1} is a valid use of **ort**.

from the other set. Furthermore, [CJM⁺05] extended this result showing that for all $k \geq 2$, one can find k such sets of uncooperative learners. Below, we give an analog of this result for cooperation in the **XBcc**-sense, where we give an infinite family of uncooperative sets, so that any learner that can **XBc**-learn *any* of the functions in one of the sets cannot **XBc**-learn *any* of the functions of any other set.

Theorem 7 (Incompatible Mutual Cooperation Camps). There is a 1-1 $e \in \mathcal{R}$ such that for all m, n , $\varphi_{e(m,n)}$ total and, defining $\mathcal{S}_n := \{\varphi_{e(m,n)} \in \mathcal{R} \mid m \in \mathbb{N}\}$, for each n all members of $\mathcal{S}_n$ **XBcc**-learn each other, while each function $h \in \mathcal{P}$ **XBcc**-learns functions from at most one of the sets in $\{\mathcal{S}_n \mid n \in \mathbb{N}\}$.

The theorem just above can be shown by an application of the Generalized Delayed Recursion Theorem [Cas74, Theorem 23].

As a contrast to the extremely cooperative functions as defined above, we say that $h \in \mathcal{R}$ is *extremely uncooperative* iff **XBcc**(h) = $\emptyset$ (i.e., h cooperates with no function). The set of all extremely uncooperative functions is denoted by EU . Trivially, $EU \neq \emptyset$, as EU contains each function h that doesn't **XBc**-learn any function. Interestingly, EU is rather big in the sense that the closure under **LinF** composition of EU is equal to $\mathcal{R}$.¹³ However, many of the functions $h \in EU$ will not **XBc**-model anything. We define a (computable) operator below, turning a given learner h into an uncooperative learner h' , which intuitively doesn't lose too much of the learning power of h .

Furthermore, Theorem 10 below states the existence of $h \in EU$ with **XBc**(h) infinite.

Definition 8. For all $h \in \mathcal{R}$ there is, by lintime **ort**, $h' \in \mathbf{LinF}$ such that

$$\forall \sigma, x : \varphi_{h'(\sigma)}(x) = \begin{cases} \varphi_h(\sigma)(x), & \text{if } \sigma = \emptyset \vee \exists s \geq \#elets(\sigma), t : \\ \varphi_{h(\sigma)}(s)(t) \downarrow \neq h'(\varphi_h(\sigma)[t]) \downarrow; & \\ \uparrow, & \text{otherwise.} \end{cases} \quad (15)$$

Intuitively, h' makes conjectures mostly behaviorally equivalent to those of h , but modified so that the conjectures are definitely wrong as soon as the input seems to learn the outputs of h' .

Define $\Psi = \lambda h \in \mathcal{R}. h'$. N.B. For each $h \in \mathcal{R}$, $\Psi(h) \in \mathbf{LinF}$.

Lemma 9. Let $h \in \mathcal{R}$. Then **XBcc**($\Psi(h)$) = $\emptyset$.

Theorem 10 (Extremely Uncooperative Infinitely Successful Learners). There are functions $h \in \mathcal{R}$ such that **XBcs**($\Psi(h)$) is infinite, but **XBcc**($\Psi(h)$) = $\emptyset$ (i.e., no function **XBc**-learned by $\Psi(h)$ can **XBc**-learn $\Psi(h)$).

The next theorem intuitively implies that requiring a learner to be extremely uncooperative will decrease its learning power with respect to plain uncooperative learning.

Corollary 11. $EU\mathbf{XBcs} \subset \mathcal{R}\mathbf{XBcs}$.¹⁴

¹³ Let $f \in \mathcal{R}$, let p be such that $\varphi_p = \lambda x. \uparrow$. Then $f' = \lambda x. \text{pad}(f(x), p) \in EU$. Define $a = \lambda x. \text{unpad}_1(x) \in \mathbf{LinF}$. Then $a \circ f' = f$.

¹⁴ Less surprisingly, one can also show $EC\mathbf{XBcc} \subset \mathcal{R}\mathbf{XBcc}$.

5 General Crossfeeding

Most lemmas, propositions and theorems of this section carry over with slightly modified hypotheses to the case of **Li** in the place of **X**.

Just below is a proposition with corollary regarding which sequence acceptance criteria allow dynamical modeling all of $\mathcal{R}$.

Proposition 12. Let δ be a sequence acceptance criterion, let $\mathcal{C} \subseteq \mathcal{P}$. Then we have

$$\mathcal{R} \in \mathcal{C}\mathbf{X}\delta \Leftrightarrow \mathcal{R} \in \mathcal{C}\mathbf{G}\delta.$$

We get the following corollary by a theorem of Harrington, cited in [CS83].

Corollary 13

$$\mathcal{R} \in \mathbf{XBc}^*.$$

Gold [Gol67] introduced learning by enumeration. Analogous to, but harder to prove than the case for **G**-style learning, we have, by the next theorem that any computably enumerable set of (total) computable functions are **XEx**-modelable,

Theorem 14 (Dynamic Modeling by Enumeration). Let $r \in \mathcal{R}$ be an enumeration of program numbers of (total) computable functions. We have

$$\{\varphi_{r(n)} \mid n \in \mathbb{N}\} \in \mathbf{XEx}^{15}$$

Proof. By **ort**, there are programs e, e' , as well as an infinite enumeration $s \in \mathcal{R}$ of programs such that, with $h = \varphi_e$ and $u = \varphi_{e'}$,¹⁶

$$\forall \sigma : u(\sigma) = \mu k \leq \# \text{elets}(\sigma) \cdot \sigma \subseteq \mathbf{X}_2(h, \varphi_{r(k)}); \quad (17)$$

$$\forall m : \varphi_{s(m)} = \mathbf{X}_2(h, \varphi_{r(m)}); \quad (18)$$

$$\forall \sigma : h(\sigma) = s(u(\sigma)). \quad (19)$$

By induction, we can show $h, u \in \mathcal{R}$. Let $n \in \mathbb{N}$. Let $g = \varphi_{r(n)}$. We show that h **XEx**-learns g . Let $p, q \in \mathcal{R}$ be such that $\mathbf{X}(h, g) = (p, q)$. Obviously, for all j , $u(q[j]) \downarrow \leq n$. Also, u is monotone in the sense that $\forall i, j : i \leq j \Rightarrow u(q[i]) \leq u(q[j])$. Thus, there is m such that

$$\forall^\infty j : u(q[j]) = m. \quad (20)$$

¹⁵ In fact, **Ex** could be replaced by any δ from a wide set of sequence acceptance criteria.

¹⁶ For a number-theoretic (partial) predicate P and $n \in \mathbb{N}$, we let

$$\mu x \leq n \cdot P(x) = \begin{cases} x, & \text{if } \forall y < x : P(y) \downarrow \neq \text{true and } P(x) \downarrow = \text{true;} \\ n+1, & \text{if } \forall y < n+1 : P(y) \downarrow \neq \text{true;} \\ \uparrow, & \text{otherwise.} \end{cases} \quad (16)$$

Hence, p converges. By (17) and (20),

$$\forall^\infty j : q[j] \subseteq \mathbf{X}_2(h, \varphi_{r(m)}); \text{ thus,} \quad (21)$$

$$q = \mathbf{X}_2(h, \varphi_{r(m)}). \quad (22)$$

The following completes the proof.

$$\forall^\infty j : \varphi_{p(j)} \stackrel{\text{def } p}{=} \varphi_{h(q[j])} \stackrel{(19)}{=} \varphi_{s(m)} \stackrel{(18)}{=} \mathbf{X}_2(h, \varphi_{r(m)}) \stackrel{(22)}{=} q. \quad (23)$$

□

The enumeration technique used in our proof of the theorem just above can be modified with techniques from [RC94] so as to obtain a linter-time computable learner for any given computably enumerable set of functions. However, in general it is not the case that any **XEx**-learnable set is also **XEx**-learnable by a linter-time learner. This will be stated formally in Corollary 18 below, for which we now give definitions to set it up.

Definition 15. Let δ be a sequence acceptance criterion.

- δ is called *non-trivial* iff $\forall \sigma : \sigma \diamond \mathcal{R} \notin \mathbf{G}\delta$.
- Let $\mathcal{D} \subseteq \mathcal{R}$. δ *allows for linter-time path finding for $\mathcal{D}$* iff there is $r \in \mathbf{LinF}$ such that, for all σ, τ of equal length and $q \in \mathcal{D}$ and e with $\sigma \subseteq q = \varphi_e$, we have $(\tau \diamond \lambda x. r(x, e, \sigma), q) \in \delta$

We illustrate the definitions by giving the following examples.

Example 16

- **Bc*** is a trivial sequence acceptance criterion, while **Ex**, **Bc**, **Bcs**, **Bcc** and **M** are non-trivial.
- **Ex**, **Bc** and **Bc*** allow for linter-time path finding for $\mathcal{R}$ as witnessed by $r = \lambda x, e, \sigma. e$. **M** allows for linter-time pathfinding for total finite variants of constant functions.¹⁷

Note that non-triviality is inherited by subsets, and allowing for linter-time path finding by supersets.

Theorem 17 (Learner Correspondence). Let δ be non-trivial such that δ allows for linter-time path finding for total finite variants of constant functions. Let $\mathcal{C}, \mathcal{C}' \subseteq \mathcal{R}$ be closed under generalized composition with **LinF** and **LinF** $\subseteq \mathcal{C}, \mathcal{C}'$. Then

$$\mathcal{C}\mathbf{X}\delta \subseteq \mathcal{C}'\mathbf{X}\delta \Leftrightarrow \mathcal{C} \subseteq \mathcal{C}'.$$

Suppose for discussion Q is a polynomial time bound. Pitt [Pit89] notes that polytime (update) Ex-learning allows unfair postponement tricks, i.e., a learner h can put off outputting significant conjectures based on data σ until it has

¹⁷ **M** allows for not necessarily linter-time path finding for $\mathcal{R}$.

seen a much larger sequence of data τ so that $Q(|\tau|)$ is enough time for h to think about σ as long as it needs. In fact, by an extension of Pitt's postponement tricks, $\mathbf{LinFGE} = \mathbf{GE}$. A direct application of Theorem 17, e.g., Corollary 18, implies that *dynamic modeling* does *not* allow for those kinds of postponement tricks *in general*.¹⁸ We let α , as from [CLRS01, §21.4], be a *very slow growing*, unbounded, linter-time computable function $\leq$ an inverse of Ackermann's function; let $\mathbf{LinF}^{+\varepsilon} := \{\varphi_e \in \mathcal{R} \mid \exists k \forall n : \Phi_e(n) \leq k \cdot |n| \cdot \log(|n|) \cdot \alpha(|n|) + k\}$. The classes $\mathbf{LinF}$ and $\mathbf{LinF}^{+\varepsilon}$ have long been known to separate [HS65].

The following corollary gives a sample of the universal power of Theorem 17.

Corollary 18 (Learner Complexity Matters). *Let $\delta \in \{\mathbf{Ex}, \mathbf{Bc}, \mathbf{M}\}$.*

- (a) $\mathbf{LinFX}\delta \subset \mathbf{LinF}^{+\varepsilon}\mathbf{X}\delta$.
- (b) $\mathbf{PFX}\delta \subset \mathbf{EXPFX}\delta$.

Theorem 17 can be generalized so as to show $\mathcal{RX}\delta \subseteq \mathcal{PX}\delta$ for δ as in Corollary 18.

Theorem 19 (Sequence Acceptance Correspondence). Let δ, δ' be such that $\delta \subseteq \mathcal{R}^2$ and δ' is non-trivial. We have

$$\mathbf{X}\delta \subseteq \mathbf{X}\delta' \Leftrightarrow \delta \subseteq \delta'.$$

The following corollary gives a sample of the universal power of Theorem 19.

Corollary 20 (Hierarchies and Separations)

- (a) For all $a, b \in \mathbb{N} \cup \{*\}$: $\mathbf{XBc}^a \not\subseteq \mathbf{XEx}^b$.
- (b) For all $a, b \in \mathbb{N} \cup \{*\}$: $\mathbf{XEx}^a \subseteq \mathbf{XBc}^b \Leftrightarrow a \leq b$.
- (c) For all $n \in \mathbb{N}$: $\mathbf{XM} \not\subseteq \mathbf{XEx}^*, \mathbf{XBc}^n$.
- (d) For all $n \in \mathbb{N}$: $\mathbf{XEx}^*, \mathbf{XBc}^n \not\subseteq \mathbf{XM}$.

References

- [Bär71] Bärzdiņš, J.: Prognostication of automata and functions. Information Processing 1, 81–84 (1971)
- [Bär74a] Bärzdiņš, J.: Inductive inference of automata, functions and programs. In: Int. Math. Congress, Vancouver, pp. 771–776 (1974)
- [Bär74b] Bärzdiņš, J.: Two theorems on the limiting synthesis of functions. In: Theory of Algorithms and Programs, Latvian State University, Riga, vol. 210, pp. 82–88 (1974)

¹⁸ However, as we saw from Theorem 14 above, such extended postponement tricks *do*, nonetheless, apply to the *special case* of the $\mathbf{XEx}$ -learning of *any* computably enumerable set of computable functions.

However, we believe we can show that Theorem 14 doesn't hold for $\mathbf{LinFXPcpEx}$ in place of $\mathbf{LinFXEx}$; hence, postdictive completeness prevents some postponement tricks.

- [BB75] Blum, L., Blum, M.: Toward a mathematical theory of inductive inference. *Information and Control* 28, 125–155 (1975)
- [BSMP91] Blum, M., De Santis, A., Micali, S., Persiano, G.: Noninteractive zero-knowledge. *SIAM J. Comput.* 20(6), 1084–1118 (1991)
- [Cas74] Case, J.: Periodicity in generations of automata. *Mathematical Systems Theory* 8, 15–32 (1974)
- [Cas94] Case, J.: Infinitary self-reference in learning theory. *Journal of Experimental and Theoretical Artificial Intelligence* 6, 3–16 (1994)
- [CJM⁺05] Case, J., Jain, S., Montagna, F., Simi, G., Sorbi, A.: On learning to coordinate: Random bits help, insightful normal forms, and competency isomorphisms. *Journal of Computer and System Sciences* 71(3), 308–332 (2005); Special issue for selected learning theory papers from COLT 2003, FOCS 2003, and STOC 2003
- [CS83] Case, J., Smith, C.: Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science* 25, 193–220 (1983)
- [CLRS01] Cormen, T., Leiserson, C., Rivest, R., Stein, C.: *Introduction to Algorithms*, 2nd edn. MIT Press, Cambridge (2001)
- [FKS95] Freivalds, R., Kinber, E.B., Smith, C.H.: On the intrinsic complexity of learning. *Information and Computation* 123(1), 64–71 (1995)
- [Gol67] Gold, E.: Language identification in the limit. *Information and Control* 10, 447–474 (1967)
- [HS65] Hartmanis, J., Stearns, R.: On the computational complexity of algorithms. *Transactions of the American Mathematical Society* 117, 285–306 (1965)
- [LV97] Li, M., Vitanyi, P.: *An Introduction to Kolmogorov Complexity and Its Applications*, 2nd edn. Springer, Heidelberg (1997)
- [Min76] Minicozzi, E.: Some natural properties of strong identification in inductive inference. In: *Theoretical Computer Science*, pp. 345–360 (1976)
- [MO99] Montagna, F., Osherson, D.: Learning to coordinate: A recursion theoretic perspective. *Synthese* 118, 363–382 (1999)
- [Pit89] Pitt, L.: Inductive inference, DFAs, and computational complexity. In: Jantke, K.P. (ed.) *AII 1989. LNCS*, vol. 397, pp. 18–44. Springer, Heidelberg (1989)
- [Pod74] Podnieks, K.: Comparing various concepts of function prediction. *Theory of Algorithms and Programs* 210, 68–81 (1974)
- [RC94] Royer, J., Case, J.: Subrecursive Programming Systems: Complexity and Succinctness. In: *Research monograph in Progress in Theoretical Computer Science*, Birkhäuser, Boston (1994)
- [Rog67] Rogers, H.: *Theory of Recursive Functions and Effective Computability*. McGraw Hill, New York (1967) (Reprinted by MIT Press, Cambridge, Massachusetts, 1987)
- [Wie76] Wiehagen, R.: Limes-erkennung rekursiver Funktionen durch spezielle Strategien. *Elektronische Informationsverarbeitung und Kybernetik* 12, 93–99 (1976)

Optimal Language Learning

John Case and Samuel E. Moelius III

Department of Computer & Information Sciences
University of Delaware
103 Smith Hall
Newark, DE 19716
{case,moelius}@cis.udel.edu

Abstract. Gold’s original paper on inductive inference introduced a notion of an *optimal learner*. Intuitively, a learner identifies a class of objects optimally iff there is no *other* learner that: requires *as little* of each presentation of each object in the class in order to identify that object, and, for *some* presentation of *some* object in the class, requires *less* of that presentation in order to identify that object. Wiehagen considered this notion in the context of *function* learning, and *characterized* an optimal function learner as one that is *class-preserving*, *consistent*, and (in a very strong sense) *non-U-shaped*, with respect to the class of functions learned.

Herein, Gold’s notion is considered in the context of *language* learning. Intuitively, a language learner identifies a class of languages optimally iff there is no other learner that: requires as little of each *text* for each language in the class in order to identify that language, and, for some text for some language in the class, requires less of that text in order to identify that language.

Many interesting results concerning optimal language learners are presented. First, it is shown that a characterization analogous to Wiehagen’s does *not* hold in this setting. Specifically, optimality is *not* sufficient to guarantee Wiehagen’s conditions; though, those conditions *are* sufficient to guarantee optimality. Second, it is shown that the failure of this analog is *not* due to a restriction on algorithmic learning power imposed by non-U-shapedness (in the strong form employed by Wiehagen). That is, non-U-shapedness, even in this strong form, does *not* restrict algorithmic learning power. Finally, for an arbitrary optimal learner $\mathbf{F}$ of a class of languages $\mathcal{L}$, it is shown that $\mathbf{F}$ optimally identifies a subclass $\mathcal{K}$ of $\mathcal{L}$ iff $\mathbf{F}$ is class-preserving with respect to $\mathcal{K}$.

1 Introduction

Gold’s original paper on inductive inference introduced a notion of an *optimal learner* [Gol67]. Intuitively, a learner identifies a class of objects optimally iff there is no *other* learner that: requires *as little* of each presentation of each object in the class in order to identify that object, and, for *some* presentation of *some* object in the class, requires *less* of that presentation in order to identify that object.

Gold's notion is perhaps most easily exemplified in the context of function learning, where each object (i.e., function) has one (canonical) presentation, namely, the sequence of all finite initial segments of that function ordered by inclusion (i.e., " $\subseteq$ "). We briefly recall the relevant definitions.

Let $\mathbb{N}$ be the set of natural numbers, $\{0, 1, 2, \dots\}$. Let $\varphi_0, \varphi_1, \dots$ be any acceptable numbering of the partial computable functions from $\mathbb{N}$ to $\mathbb{N}$ [Rog67]. For each function $f : \mathbb{N} \rightarrow \mathbb{N}$, and each $n \in \mathbb{N}$, let $f[n]$ denote the initial segment of f whose domain is of size n . A learner $\mathbf{F}$ *identifies* a class of functions $\mathcal{F} \subseteq \mathbb{N} \rightarrow \mathbb{N} \stackrel{\text{def}}{=} \{f \mid \text{for each } f \in \mathcal{F}, \text{ there exists } n \in \mathbb{N} \text{ such that } \varphi_{\mathbf{F}(f[n])} = f \text{ and } (\forall i \geq n)[\mathbf{F}(f[i]) = \mathbf{F}(f[n])]\}$.

For each function learner $\mathbf{F}$, and each $f : \mathbb{N} \rightarrow \mathbb{N}$, let conv be as follows.

$$\text{conv}(\mathbf{F}, f) = \begin{cases} n, & \text{where } n \text{ is least such that } \varphi_{\mathbf{F}(f[n])} = f \\ & \text{and } (\forall i \geq n)[\mathbf{F}(f[i]) = \mathbf{F}(f[n])], \text{ if} \\ & \text{such an } n \text{ exists;} \\ \text{undefined, otherwise.} \end{cases} \quad (1)$$

Intuitively, $\text{conv}(\mathbf{F}, f)$ indicates *how much* of f must be presented to $\mathbf{F}$ in order for $\mathbf{F}$ to identify f . Thus, if $\mathbf{F}$ and $\mathbf{G}$ are two function learners and $\text{conv}(\mathbf{F}, f) \leq \text{conv}(\mathbf{G}, f)$ (which are both defined), then $\mathbf{F}$ requires *as little* of f as $\mathbf{G}$ requires to identify f .

In the context of function learning, Gold's notion can be made precise as follows. A function learner $\mathbf{F}$ *optimally identifies* a class of functions $\mathcal{F} \stackrel{\text{def}}{=} \{f \mid \mathbf{F} \text{ identifies } \mathcal{F}, \text{ and, for each function learner } \mathbf{G},$

$$\begin{aligned} (\forall f \in \mathcal{F})[\text{conv}(\mathbf{G}, f) \leq \text{conv}(\mathbf{F}, f)] &\Rightarrow \\ (\forall f \in \mathcal{F})[\text{conv}(\mathbf{F}, f) \leq \text{conv}(\mathbf{G}, f)]. \end{aligned} \quad (2)$$

Thus, $\mathbf{F}$ optimally identifies $\mathcal{F}$ iff, for every *other* function learner $\mathbf{G}$, if $\mathbf{G}$ requires *as little* of each $f \in \mathcal{F}$ as $\mathbf{F}$ requires to identify f , then (conversely) $\mathbf{F}$ requires *as little* of each $f \in \mathcal{F}$ as $\mathbf{G}$ requires to identify f . Equivalently: there is *no* other learner $\mathbf{G}$ such that, $\mathbf{G}$ requires as little of each $f \in \mathcal{F}$ as $\mathbf{F}$ requires to identify f , and, for *some* $f \in \mathcal{F}$, $\mathbf{G}$ requires *less* of f than $\mathbf{F}$ requires to identify f .

Wiehagen [Wie91] considered optimal learners in the context of function learning, and characterized them as follows.

Theorem 1 (Wiehagen [Wie91]). Suppose that a function learner $\mathbf{F}$ identifies a class of functions $\mathcal{F}$. Then, $\mathbf{F}$ optimally identifies $\mathcal{F} \Leftrightarrow$ (a) through (c) below.

(a) $\mathbf{F}$ is *class-preserving* [Wie91] with respect to $\mathcal{F}$, i.e.,

$$(\forall f \in \mathcal{F})(\forall n \in \mathbb{N})[\varphi_{\mathbf{F}(f[n])} \in \mathcal{F}]. \quad (3)$$

(b) $\mathbf{F}$ is *consistent* [Bar77, BB75] with respect to $\mathcal{F}$, i.e.,

$$(\forall f \in \mathcal{F})(\forall n \in \mathbb{N})[f[n] \subseteq \varphi_{\mathbf{F}(f[n])}]. \quad (4)$$

(c) $\mathbf{F}$ is *strongly non-U-shaped* [Wie91]¹ with respect to $\mathcal{F}$, i.e.,

$$(\forall f \in \mathcal{F})(\forall n \in \mathbb{N}) \left[\varphi_{\mathbf{F}(f[n])} = f \Rightarrow (\forall i \geq n) [\mathbf{F}(f[i]) = \mathbf{F}(f[n])] \right]. \quad (5)$$

Herein, we consider optimal learners in the context of *language* learning, as done in [OSW86, Ch. 8]. In this setting, the situation is slightly more complicated, since, for nearly every object (i.e., language), there is *more than one* presentation (i.e., text). We briefly recall the relevant definitions.

For each $p \in \mathbb{N}$, let $W_p = \{x \in \mathbb{N} \mid \varphi_p(x) \text{ converges}\}$. Thus, $W_0, W_1, \dots$ is an enumeration of the recursively enumerable (r.e.) sets [Rog67]. A *language* is a subset of $\mathbb{N}$. A *text* for a language L is a function $T : \mathbb{N} \rightarrow (\mathbb{N} \cup \{\#\})$ such that L is *exactly* the non- $\#$ elements of the range of T , i.e., $L = \{x \in \mathbb{N} \mid (\exists i)[T(i) = x]\}$. (The symbol ‘ $\#$ ’ is pronounced *pause*.) Clearly, a text uniquely determines a language. Furthermore, if L is a *non-empty* language, then there are *uncountably* many texts for L . A language learner $\mathbf{F}$ *identifies* a class of languages $\mathcal{L} \stackrel{\text{def}}{=}$ for each $L \in \mathcal{L}$, and each text T for L , there exists $n \in \mathbb{N}$ such that $W_{\mathbf{F}(T[n])} = L$ and $(\forall i \geq n) [\mathbf{F}(T[i]) = \mathbf{F}(T[n])]$.

For each language learner $\mathbf{F}$, and each text T , let conv be as follows.

$$\text{conv}(\mathbf{F}, T) = \begin{cases} n, & \text{where } n \text{ is least such that } W_{\mathbf{F}(T[n])} = L, \\ & (\forall i \geq n) [\mathbf{F}(T[i]) = \mathbf{F}(T[n])], \text{ and } T \text{ is} \\ & \text{a text for } L, \text{ if such } n \text{ and } L \text{ exist;} \\ \text{undefined,} & \text{otherwise.} \end{cases} \quad (6)$$

In the context of language learning, Gold’s notion can be made precise as follows. A language learner $\mathbf{F}$ *optimally identifies* a class of languages $\mathcal{L} \stackrel{\text{def}}{=}$ $\mathbf{F}$ identifies $\mathcal{L}$, and, for each language learner $\mathbf{G}$,

$$(\forall L \in \mathcal{L})(\forall T \text{ a text for } L) [\text{conv}(\mathbf{G}, T) \leq \text{conv}(\mathbf{F}, T)] \Rightarrow (\forall L \in \mathcal{L})(\forall T \text{ a text for } L) [\text{conv}(\mathbf{F}, T) \leq \text{conv}(\mathbf{G}, T)]. \quad (7)$$

This definition has an interpretation similar to that of the function learning setting. Specifically: $\mathbf{F}$ optimally identifies $\mathcal{L}$ iff, for every *other* language learner $\mathbf{G}$, if $\mathbf{G}$ requires *as little* of each text for each $L \in \mathcal{L}$ as $\mathbf{F}$ requires to identify L , then (conversely) $\mathbf{F}$ requires *as little* of each text for each $L \in \mathcal{L}$ as $\mathbf{G}$ requires to identify L . Equivalently: there is *no* other learner $\mathbf{G}$ such that, $\mathbf{G}$ requires as little of each text for each $L \in \mathcal{L}$ as $\mathbf{F}$ requires to identify L , and, for *some* text for *some* $L \in \mathcal{L}$, $\mathbf{G}$ requires *less* of that text than $\mathbf{F}$ requires to identify L .

Many interesting results concerning optimal language learners are presented. First, we show that a characterization analogous to Wiehagen’s (Theorem 1 above) does *not* hold in this setting. Specifically, optimality is *not* sufficient to guarantee Wiehagen’s conditions; though, those conditions *are* sufficient to

¹ Wiehagen actually used the term *semantically finite* in place of *strongly non-U-shaped*. However, there is a clear connection between this notion and that of *non-U-shapedness* [CCJS07, BCM⁺08, CCJS08, CM08b]. Our choice of terminology is meant to expose this connection.

guarantee optimality (Theorem 8 in Section 3). Second, we show that the failure of this analog is *not* due to a restriction on algorithmic learning power imposed by strong non-U-shapedness. That is, strong non-U-shapedness does *not* restrict algorithmic learning power (Theorem 12 in Section 3). Finally, for an arbitrary optimal learner $\mathbf{F}$ of a class of languages $\mathcal{L}$, we show that $\mathbf{F}$ optimally identifies a subclass $\mathcal{K}$ of $\mathcal{L}$ iff $\mathbf{F}$ is class-preserving with respect to $\mathcal{K}$ (Theorem 13 in Section 4).

A primary motivation for considering optimal language learners is the following. There is no generally accepted notion of *efficient algorithmic* language learning.² Optimal learners are, in some sense, *maximally efficient*, in that they use as little of the presentation of an object as possible. Thus, one way to argue that an algorithmic learner is efficient, is to argue that it is *relatively efficient* compared to an optimal learner. We give an example (beginning with (8) below), following some necessary definitions.

Let σ range over finite initial segments of texts. For each text T , and each $n \in \mathbb{N}$, let $T[n]$ denote the initial segment of T of length n . For each σ , let $\text{content}(\sigma) = \{x \in \mathbb{N} \mid (\exists i)[\sigma(i) = x]\}$. Let K be the diagonal halting problem, i.e., $K = \{p \in \mathbb{N} \mid p \in W_p\}$ [Rog67]. For each set $A \subseteq \mathbb{N}$, let $\overline{A} = \mathbb{N} - A$ and $A + 1 = \{x + 1 \mid x \in A\}$.

Let $\mathcal{L}$ be as follows.

$$\mathcal{L} = \{\{0\}\} \cup \{\{p + 1\} \mid p \in K\} \cup \{\{0, p + 1\} \mid p \in \overline{K}\}. \quad (8)$$

Let f be such that, for each finite $A \subset \mathbb{N}$,

$$W_{f(A)} = A.^3 \quad (9)$$

For each σ , let $\mathbf{M}$ and $\mathbf{F}$ be as follows.

$$\mathbf{M}(\sigma) = f(\text{content}(\sigma)). \quad (10)$$

$$\mathbf{F}(\sigma) = \begin{cases} f(\{0\}), & \text{if } \text{content}(\sigma) \subseteq \{0\}; \\ f(\text{content}(\sigma)), & \text{if } \text{content}(\sigma) \cap (K + 1) \neq \emptyset; \\ f(\{0\} \cup \text{content}(\sigma)), & \text{if } \text{content}(\sigma) \cap (\overline{K} + 1) \neq \emptyset. \end{cases} \quad (11)$$

The discussion proceeds with the observation of a few facts.

Fact 1. $\mathcal{L}$ is *not* algorithmically, optimally identifiable.⁴

Proof. By way of contradiction, let $\mathbf{M}'$ be an algorithmic learner that optimally identifies $\mathcal{L}$. Then, by (b) $\Rightarrow$ (c) of Theorem 8 (Section 3 below), $\mathbf{M}'$ class-preservingly and consistently identifies $\mathcal{L}$. Note that, for each $p \in \mathbb{N}$, there is exactly *one* $L \in \mathcal{L}$ such that $p + 1 \in L$. It follows that

$$\overline{K} = \{p \in \mathbb{N} \mid 0 \in W_{\mathbf{M}'(p+1)}\}. \quad (12)$$

² See [Pit89] for a discussion.

³ Such an f exists by s-m-n [Rog67].

⁴ This is shown for a nearly identical class of languages in [OSW86, Proposition 8.2.3A]. The proof of Fact 1 is included here for illustration.

Since the right hand side of (12) is r.e. (by supposition), this is a contradiction. $\square$ (*Fact 1*)

Fact 2. $\mathbf{M}$ algorithmically identifies $\mathcal{L}$, but *not* optimally.

Proof. Clearly, $\mathbf{M}$ identifies $\mathcal{L}$, and $\mathbf{M}$ is algorithmic. Thus, by Fact 1, $\mathbf{M}$ *cannot* optimally identify $\mathcal{L}$. $\square$ (*Fact 2*)

Fact 3. $\mathbf{F}$ optimally identifies $\mathcal{L}$, but *not* algorithmically.

Proof. Clearly, $\mathbf{F}$ identifies $\mathcal{L}$. Furthermore, $\mathbf{F}$ is class-preserving, consistent, and strongly non-U-shaped with respect to $\mathcal{L}$. Thus, by (a) $\Rightarrow$ (b) of Theorem 8 (Section 3 below), $\mathbf{F}$ optimally identifies $\mathcal{L}$. Finally, by Fact 1, $\mathbf{F}$ *cannot* be algorithmic. $\square$ (*Fact 3*)

Fact 4. On any text T for a language in $\mathcal{L}$, $\mathbf{M}$ requires at most *one more* data-point than $\mathbf{F}$ requires to converge to a correct hypothesis on T . Formally: for each text T for a language in $\mathcal{L}$,

$$|\text{content}(T[\text{conv}(\mathbf{M}, T)])| \leq |\text{content}(T[\text{conv}(\mathbf{F}, T)])| + 1. \quad (13)$$

Proof. A straightforward case analysis. $\square$ (*Fact 4*)

Fact 4 gives a sense in which $\mathbf{M}$ is *relatively efficient* compared to $\mathbf{F}$. Generalizations of this notion might allow, e.g., that the size of the set on the right-hand side of (13) be the argument of an arbitrary polynomial.⁵

Of course, any such notion of relative efficiency is meaningful *only* for those classes of languages for which there exists an optimal learner. Fortunately, however, Proposition 8.2.1A in [OSW86] says that, for *every* identifiable class of languages, there exists an optimal learner.

We hope that the ideas presented here provide for a useful notion of efficient algorithmic language learning.

2 Preliminaries

Computability-theoretic concepts not covered below are treated in [Rog67].

Lowercase math-italic letters (e.g., a, b, c), with or without decorations, range over elements of $\mathbb{N}$, unless stated otherwise. Uppercase math-italic letters (e.g., A, B, C), with or without decorations, range over subsets of $\mathbb{N}$, unless stated otherwise. $D_0, D_1, \dots$ denotes a canonical enumeration of all finite subsets of $\mathbb{N}$. $\mathcal{K}$ and $\mathcal{L}$ range over collections of subsets of $\mathbb{N}$. $\mathcal{E} \stackrel{\text{def}}{=} \{W_p \mid p \in \mathbb{N}\}$. For each A , $|A|$ denotes the cardinality of A . For each finite, *non-empty* A , $\max A$ denotes the maximum element of A . For an arbitrary set $\mathfrak{X}$, $2^{\mathfrak{X}}$ denotes the collection of all subsets of $\mathfrak{X}$.

⁵ We do not mean to suggest that the content-based measure of (13) represents the *best* possible measure of relative efficiency, just that it is a reasonable one. Alternatives might involve, e.g., *mind-change complexity* [BF74, CS83].

For each one-argument partial function ψ , and each x , $\psi(x)\downarrow$ denotes that $\psi(x)$ converges; $\psi(x)\uparrow$ denotes that $\psi(x)$ diverges.⁶ We use $\uparrow$ to denote the value of a divergent computation. Φ denotes a fixed Blum complexity measure for φ . For each i and s , $W_i^s \stackrel{\text{def}}{=} \{x \mid x < s \wedge \Phi_i(x) \leq s\}$.

$\mathbb{N}_\# \stackrel{\text{def}}{=} \mathbb{N} \cup \{\#\}$. TXT denotes the set of all texts, i.e., functions of type $\mathbb{N} \rightarrow \mathbb{N}_\#$. SEQ denotes the set of all sequences, i.e., finite initial segments of texts. T , with or without decorations, ranges over elements of TXT . Lowercase Greek letters (e.g., ρ , σ , τ), with or without decorations, range over elements of SEQ , unless stated otherwise. For each L and $\mathcal{L}$, TXT_L , $\text{TXT}_{\mathcal{L}}$, SEQ_L , and $\text{SEQ}_{\mathcal{L}}$ are defined as follows.

$$\text{TXT}_L = \{T \mid \text{content}(T) = L\}. \quad (14)$$

$$\text{TXT}_{\mathcal{L}} = \{T \mid (\exists L \in \mathcal{L})[\text{content}(T) = L]\}. \quad (15)$$

$$\text{SEQ}_L = \{\sigma \mid (\exists T \in \text{TXT}_L)[\sigma \subset T]\}. \quad (16)$$

$$\text{SEQ}_{\mathcal{L}} = \{\sigma \mid (\exists T \in \text{TXT}_{\mathcal{L}})[\sigma \subset T]\}. \quad (17)$$

In order to disambiguate expressions such as $\text{SEQ}_\emptyset$, we write $\emptyset$ for the empty language, and $\{\}$ for the empty class of languages.

For each $A \subseteq \mathbb{N}_\#$, $A^* \stackrel{\text{def}}{=} \{\sigma \mid (\forall i)[\sigma(i)\downarrow \Rightarrow \sigma(i) \in A]\}$. Similarly, for each $A \subseteq \mathbb{N}_\#$, $A^\omega \stackrel{\text{def}}{=} \{T \mid (\forall i)[T(i) \in A]\}$. For each $x \in \mathbb{N}_\#$, x^ω denotes the unique element of $\{x\}^\omega$. For each $A \subseteq \mathbb{N}_\#$, $A^{\leq\omega} = A^* \cup A^\omega$. In particular, $\mathbb{N}_\#^{\leq\omega} = \text{SEQ} \cup \text{TXT}$.

For each $f \in \mathbb{N}_\#^{\leq\omega}$, $\text{content}(f) \stackrel{\text{def}}{=} \{x \in \mathbb{N} \mid (\exists i)[f(i) = x]\}$. For each $f \in \mathbb{N}_\#^{\leq\omega}$ and n , $f[n]$ denotes the initial segment of f of length n , if it exists; f , otherwise. For each σ , $|\sigma|$ denotes the length of σ (equivalently, $|\{i \mid \sigma(i)\downarrow\}|$). For each non-empty σ , $\sigma^- \stackrel{\text{def}}{=} \sigma[|\sigma| - 1]$. For each σ , and each $f \in \mathbb{N}_\#^{\leq\omega}$, $\sigma \cdot f$ denotes the concatenation of σ and f (in that order). Similarly, for each $A \subseteq \text{SEQ}$ and $\mathcal{B} \subseteq \mathbb{N}_\#^{\leq\omega}$, $A \cdot \mathcal{B} \stackrel{\text{def}}{=} \{\sigma \cdot f \mid \sigma \in A \wedge f \in \mathcal{B}\}$. λ denotes the empty sequence (equivalently, the everywhere divergent function).

Following conventions similar to [JORS99], **F**, **G**, and **H**, with our without decorations, range over arbitrary (partial) functions of type $\text{SEQ} \rightarrow \mathbb{N}$; whereas, **M**, with our without decorations, ranges over *algorithmic* (partial) functions of type $\text{SEQ} \rightarrow \mathbb{N}$.

conv was defined in (6) (Section 1). For each **F** and T , we write $\text{conv}(\mathbf{F}, T)\downarrow$ when $\text{conv}(\mathbf{F}, T)$ is defined, and $\text{conv}(\mathbf{F}, T)\uparrow$ when $\text{conv}(\mathbf{F}, T)$ is undefined. An expression that the reader will see frequently is

$$T[\text{conv}(\mathbf{F}, T)], \quad (18)$$

which is the *shortest* initial segment of T causing **F** to converge to a correct hypothesis for $\text{content}(T)$ (if such an initial segment exists).

⁶ For each one-argument partial function ψ , and each x , $\psi(x)$ converges iff there exists y such that $\psi(x) = y$; $\psi(x)$ diverges iff there is no y such that $\psi(x) = y$. If ψ is partial computable, and x is such that $\psi(x)$ diverges, then one can imagine that a program for ψ goes into an infinite loop on input x .

Proposition 2. $(\forall \mathbf{F}, T, \sigma)[T[\text{conv}(\mathbf{F}, T)] \subseteq \sigma \subset T \Rightarrow W_{\mathbf{F}(\sigma)} = \text{content}(T)]$.

Proof of Proposition. Let $\mathbf{F}$, T , and σ be fixed, and suppose that $T[\text{conv}(\mathbf{F}, T)] \subseteq \sigma \subset T$. Let $n = \text{conv}(\mathbf{F}, T)$. By the definition of conv , $W_{\mathbf{F}(T[n])} = \text{content}(T)$ and $(\forall i \geq n)[\mathbf{F}(T[i]) = \mathbf{F}(T[n])]$. Let i be such that $T[i] = \sigma$. Clearly, $i \geq n$. Thus, $W_{\mathbf{F}(\sigma)} = W_{\mathbf{F}(T[i])} = W_{\mathbf{F}(T[n])} = \text{content}(T)$. $\square$ (*Proposition 2*)

The following are the Gold-style learning criteria of relevance to this paper.

Definition 3. Let $\mathbf{F}$ and $\mathcal{L}$ be fixed.

(a) (**Gold [Gol67]**) $\mathbf{F}$ identifies $\mathcal{L} \Leftrightarrow$

$$(\forall \sigma \in \text{SEQ}_{\mathcal{L}})[\mathbf{F}(\sigma) \downarrow] \wedge (\forall T \in \text{TXT}_{\mathcal{L}})[\text{conv}(\mathbf{F}, T) \downarrow]. \quad (19)$$

(b) (**Wiehagen [Wie91]**) $\mathbf{F}$ class-preservingly identifies $\mathcal{L} \Leftrightarrow \mathbf{F}$ identifies $\mathcal{L}$ and

$$(\forall \sigma \in \text{SEQ}_{\mathcal{L}})[W_{\mathbf{F}(\sigma)} \in \mathcal{L}]. \quad (20)$$

(c) (**Angluin [Ang80]**) $\mathbf{F}$ consistently identifies $\mathcal{L} \Leftrightarrow \mathbf{F}$ identifies $\mathcal{L}$ and

$$(\forall \sigma \in \text{SEQ}_{\mathcal{L}})[\text{content}(\sigma) \subseteq W_{\mathbf{F}(\sigma)}]. \quad (21)$$

(d) (**Baliga, et al. [BCM⁺08], Carlucci, et al. [CCJS08]**) $\mathbf{F}$ non-U-shapedly identifies $\mathcal{L} \Leftrightarrow \mathbf{F}$ identifies $\mathcal{L}$ and

$$(\forall L \in \mathcal{L})(\forall \sigma, \tau \in \text{SEQ}_L)[[\sigma \subseteq \tau \wedge W_{\mathbf{F}(\sigma)} \neq W_{\mathbf{F}(\tau)}] \Rightarrow W_{\mathbf{F}(\sigma)} \neq L]. \quad (22)$$

(e) (**Wiehagen [Wie91]**) $\mathbf{F}$ strongly non-U-shapedly⁷ identifies $\mathcal{L} \Leftrightarrow \mathbf{F}$ identifies $\mathcal{L}$ and

$$(\forall L \in \mathcal{L})(\forall \sigma, \tau \in \text{SEQ}_L)[[\sigma \subseteq \tau \wedge \mathbf{F}(\sigma) \neq \mathbf{F}(\tau)] \Rightarrow W_{\mathbf{F}(\sigma)} \neq L]. \quad (23)$$

N.B. Some authors (including ourselves, at times) make allowances outside of those of Definition 3(a), such as: (1) allowing $(\exists \sigma \in \text{SEQ}_{\mathcal{L}})[\mathbf{F}(\sigma) \uparrow]$, and (2) allowing $\mathbf{F} : \text{SEQ} \rightarrow (\mathbb{N} \cup \{?\})$. However, for the purposes of this paper, insisting that $\mathbf{F}$ satisfy the more stringent requirements of Definition 3(a) greatly simplifies the presentation. Moreover, such insistence does not affect the essential content of our results.

Definition 4. For each $\mathcal{L}$, $\mathcal{L}$ is *identifiable* $\Leftrightarrow (\exists \mathbf{F})[\mathbf{F}$ identifies $\mathcal{L}]$.

N.B. “ $\mathcal{L}$ is identifiable” is *not* equivalent to “ $\mathcal{L}$ is *algorithmically* identifiable”, the latter of which would mean $(\exists \mathbf{M})[\mathbf{M}$ identifies $\mathcal{L}]$.

Definition 5. Let $\mathcal{L}$ be fixed.

(a) Suppose that $\mathbf{F}$ and $\mathbf{G}$ each identify $\mathcal{L}$. Then, (i) and (ii) below.

(i) $\mathbf{F} \preceq_{\mathcal{L}} \mathbf{G} \Leftrightarrow (\forall T \in \text{TXT}_{\mathcal{L}})[\text{conv}(\mathbf{F}, T) \leq \text{conv}(\mathbf{G}, T)]$.

(ii) $\mathbf{F} \prec_{\mathcal{L}} \mathbf{G} \Leftrightarrow [\mathbf{F} \preceq_{\mathcal{L}} \mathbf{G} \wedge (\exists T \in \text{TXT}_{\mathcal{L}})[\text{conv}(\mathbf{F}, T) < \text{conv}(\mathbf{G}, T)]]$.

(b) For each $\mathbf{F}$, $\mathbf{F}$ *optimally identifies* $\mathcal{L} \Leftrightarrow [\mathbf{F}$ identifies $\mathcal{L} \wedge (\forall \mathbf{G})[\mathbf{G} \not\prec_{\mathcal{L}} \mathbf{F}]]$.

⁷ See footnote 1 above.

3 Properties of Optimal Learners

In this section, we show that a characterization analogous to Wiehagen's (Theorem 1 in Section 1) does *not* hold in the language learning setting. Specifically, optimality is *not* sufficient to guarantee Wiehagen's conditions; though, those conditions *are* sufficient to guarantee optimality (Theorem 8 below). We also show that the failure of this analog is *not* due to a restriction on algorithmic learning power imposed by strong non-U-shapedness. That is, strong non-U-shapedness does *not* restrict algorithmic learning power (Theorem 12 below).

The proof of Theorem 8 relies on the following two lemmas.

Lemma 6. Suppose that $\mathbf{F}$ class-preservingly, consistently, and strongly non-U-shapedly identifies $\mathcal{L}$. Then, for each $\sigma \in \text{SEQ}_{\mathcal{L}}$, there exists $L \in \mathcal{L}$ such that

$$\text{content}(\sigma) \subseteq L \wedge (\forall T \in \text{TXT}_{\mathcal{L}})[\sigma \subset T \Rightarrow \text{conv}(\mathbf{F}, T) \leq |\sigma|]. \quad (24)$$

Proof. Let $\mathbf{F}$, $\mathcal{L}$, and σ be as stated. Since $\mathbf{F}$ class-preservingly identifies $\mathcal{L}$, $W_{\mathbf{F}(\sigma)} \in \mathcal{L}$. Let $L = W_{\mathbf{F}(\sigma)}$. Since $\mathbf{F}$ consistently identifies $\mathcal{L}$, $\text{content}(\sigma) \subseteq L$. Since $\mathbf{F}$ strongly non-U-shapedly identifies $\mathcal{L}$, $(\forall \tau \in \text{SEQ}_{\mathcal{L}})[\sigma \subseteq \tau \Rightarrow \mathbf{F}(\sigma) = \mathbf{F}(\tau)]$. Clearly, the lemma follows. $\square$ (Lemma 6)

Lemma 7. Suppose that $\mathbf{F}$ and $\mathbf{G}$ each identify $\mathcal{L}$. Further suppose that $A \subseteq \text{SEQ}$ is such that

$$(\forall \sigma \notin A)[\mathbf{F}(\sigma) = \mathbf{G}(\sigma)]. \quad (25)$$

Then,

$$(\forall T \in \text{TXT}_{\mathcal{L}}) \left[\text{conv}(\mathbf{F}, T) < \text{conv}(\mathbf{G}, T) \Rightarrow (\exists \sigma \in A)[T[\text{conv}(\mathbf{F}, T)] \subseteq \sigma \subset T] \right]. \quad (26)$$

Proof. Let $\mathbf{F}$, $\mathbf{G}$, $\mathcal{L}$, and A be as stated. Let $T \in \text{TXT}_{\mathcal{L}}$ be such that $\text{conv}(\mathbf{F}, T) < \text{conv}(\mathbf{G}, T)$. Clearly, there exists σ such that $T[\text{conv}(\mathbf{F}, T)] \subseteq \sigma \subset T$ and $\mathbf{F}(\sigma) \neq \mathbf{G}(\sigma)$. By (25), $\sigma \in A$. $\square$ (Lemma 7)

The following is the first main result of this section.

Theorem 8. Let $\mathbf{F}$ and $\mathcal{L}$ be fixed. Then,

$$(a) \not\Rightarrow (b) \not\Rightarrow (c), \quad (27)$$

where (a) through (c) are as follows.⁸

- (a) $\mathbf{F}$ class-preservingly, consistently, and strongly non-U-shapedly identifies $\mathcal{L}$.
- (b) $\mathbf{F}$ optimally identifies $\mathcal{L}$.
- (c) $\mathbf{F}$ class-preservingly and consistently identifies $\mathcal{L}$.

⁸ (a) $\Rightarrow$ (b) of Theorem 8 is an improvement on Proposition 8.2.2A in [OSW86].

Proof. Let $\mathbf{F}$ and $\mathcal{L}$ be as stated.

(a) $\Rightarrow$ (b): Suppose that $\mathbf{F}$ class-preservingly, consistently, and strongly non-U-shapedly identifies $\mathcal{L}$. Further suppose, by way of contradiction, that there exist $\mathbf{G}$, $L \in \mathcal{L}$, and $T \in \text{TxT}_L$ such that $\mathbf{G} \preceq_{\mathcal{L}} \mathbf{F}$ and $\text{conv}(\mathbf{G}, T) < \text{conv}(\mathbf{F}, T)$. Let $\sigma = T[\text{conv}(\mathbf{G}, T)]$. By Lemma 6, there exists $L' \in \mathcal{L}$ such that $\text{content}(\sigma) \subseteq L'$ and

$$(\forall T' \in \text{TxT}_{L'})[\sigma \subset T' \Rightarrow \text{conv}(\mathbf{F}, T') \leq |\sigma|]. \quad (28)$$

If $L = L'$, then

$$\begin{aligned} \text{conv}(\mathbf{F}, T) &\leq |\sigma| && \{\text{by (28)}\} \\ &= \text{conv}(\mathbf{G}, T) && \{\text{by the choice of } \sigma\} \\ &< \text{conv}(\mathbf{F}, T) && \{\text{by the choice of } T\} \end{aligned}$$

— a contradiction. So, it must be the case that $L \neq L'$. Note that, by the choice of σ ,

$$W_{\mathbf{G}(\sigma)} = W_{\mathbf{G}(T[\text{conv}(\mathbf{G}, T)])} = L. \quad (29)$$

Let $T' \in \text{TxT}_{L'}$ be *any* such that $\sigma \subset T'$. Then,

$$\begin{aligned} \text{conv}(\mathbf{F}, T') &\leq |\sigma| && \{\text{by (28)}\} \\ &< \text{conv}(\mathbf{G}, T') && \{\text{by (29) and } L \neq L'\}. \end{aligned}$$

But this contradicts $\mathbf{G} \preceq_{\mathcal{L}} \mathbf{F}$.

(b) $\Rightarrow$ (c): Suppose that $\mathbf{F}$ optimally identifies $\mathcal{L}$. Further suppose, by way of contradiction, that $\mathbf{F}$ does *not* class-preservingly identify $\mathcal{L}$, or that $\mathbf{F}$ does *not* consistently identify $\mathcal{L}$. Then, there exists $\rho \in \text{SEQ}_{\mathcal{L}}$ such that *at least one* of (i) or (ii) below holds.

- (i) $W_{\mathbf{F}(\rho)} \notin \mathcal{L}$ (in the case that $\mathbf{F}$ does *not* class-preservingly identify $\mathcal{L}$).
- (ii) $\text{content}(\rho) \not\subseteq W_{\mathbf{F}(\rho)}$ (in the case that $\mathbf{F}$ does *not* consistently identify $\mathcal{L}$).

Let $T \in \text{TxT}_{\mathcal{L}}$ be such that $\rho \subset T$.

Claim 8.1. Suppose that $\sigma \in \text{SEQ}_{\mathcal{L}}$ is such that $\rho \subseteq \sigma \subseteq T[\text{conv}(\mathbf{F}, T) - 1]$. Then,

$$(\forall T' \in \text{TxT}_{\mathcal{L}})[\sigma \subset T' \Rightarrow \text{conv}(\mathbf{F}, T') > |\sigma|]. \quad (30)$$

Proof of Claim. The proof is by induction on the length of σ . The case when $\rho = \sigma$ is straightforward by the choice of ρ and (i) or (ii) above. So, let σ be such that $\rho \subset \sigma \subseteq T[\text{conv}(\mathbf{F}, T) - 1]$, and suppose that

$$(\forall T' \in \text{TxT}_{\mathcal{L}})[\sigma^- \subset T' \Rightarrow \text{conv}(\mathbf{F}, T') > |\sigma^-|]. \quad (31)$$

Further suppose, by way of contradiction, that, for some $T' \in \text{TxT}_{\mathcal{L}}$, $\sigma \subset T'$ and $\text{conv}(\mathbf{F}, T') \leq |\sigma|$. Then, by (31), $\text{conv}(\mathbf{F}, T') = |\sigma|$. Let $\mathbf{G}$ be such that, for each τ ,

$$\mathbf{G}(\tau) = \begin{cases} \mathbf{F}(\sigma), & \text{if } \tau = \sigma^-; \\ \mathbf{F}(\tau), & \text{otherwise.} \end{cases} \quad (32)$$

Clearly, $\mathbf{G}$ identifies $\mathcal{L}$ and $\text{conv}(\mathbf{G}, T') \leq |\sigma^-| < |\sigma|$. Thus, if it can be shown that $\mathbf{G} \preceq_{\mathcal{L}} \mathbf{F}$, then this would (as desired) contradict the fact that $\mathbf{F}$ optimally identifies $\mathcal{L}$. So, suppose that $\mathbf{G} \not\preceq_{\mathcal{L}} \mathbf{F}$. Let $T'' \in \text{TxT}_{\mathcal{L}}$ be such that $\text{conv}(\mathbf{F}, T'') < \text{conv}(\mathbf{G}, T'')$. Then, by Lemma 7 (with $A = \{\sigma^-\}$), $T''[\text{conv}(\mathbf{F}, T'')] \subseteq \sigma^- \subset T''$. But this contradicts (31). $\square$ (*Claim 8.1*)

Let $\mathbf{G}$ be such that, for each σ ,

$$\mathbf{G}(\sigma) = \begin{cases} \mathbf{F}(T[\text{conv}(\mathbf{F}, T)]), & \text{if } \rho \subseteq \sigma \subseteq T[\text{conv}(\mathbf{F}, T) - 1]; \\ \mathbf{F}(\sigma), & \text{otherwise.} \end{cases} \quad (33)$$

Clearly, $\mathbf{G}$ identifies $\mathcal{L}$ and $\text{conv}(\mathbf{G}, T) \leq |\rho| < \text{conv}(\mathbf{F}, T)$. Thus, if it can be shown that $\mathbf{G} \preceq_{\mathcal{L}} \mathbf{F}$, then this would (as desired) contradict the fact that $\mathbf{F}$ optimally identifies $\mathcal{L}$. So, suppose that $\mathbf{G} \not\preceq_{\mathcal{L}} \mathbf{F}$. Let $T' \in \text{TxT}_{\mathcal{L}}$ be such that $\text{conv}(\mathbf{F}, T') < \text{conv}(\mathbf{G}, T')$. Then, by Lemma 7, there exists σ such that $\rho \subseteq \sigma \subseteq T[\text{conv}(\mathbf{F}, T) - 1]$ and $T'[\text{conv}(\mathbf{F}, T')] \subseteq \sigma \subset T'$. But this contradicts Claim 8.1.

(a) $\neq$ (b): Let $\mathcal{L} = \{\emptyset, \{0\}\}$. Let $p_{\emptyset}$ and $p_{\{0\}}$ be grammars for $\emptyset$ and $\{0\}$, respectively. Let $\mathbf{M}$ be as follows.

$$\begin{aligned} \mathbf{M}(\lambda) &= p_{\{0\}} \cdot \\ \mathbf{M}(0 \cdot \{\#, 0\}^*) &= p_{\{0\}} \cdot \\ \mathbf{M}(\# \cdot \{\#\}^*) &= p_{\emptyset} \cdot \\ \mathbf{M}(\# \cdot \{\#\}^* \cdot 0 \cdot \{\#, 0\}^*) &= p_{\{0\}} \cdot \end{aligned} \quad (34)$$

Clearly, $\mathbf{M}$ identifies $\mathcal{L}$. Note that $\mathbf{M}$ is U-shaped, e.g., on the text $\# \cdot 0^{\omega}$. It remains to show that $\mathbf{M}$ optimally identifies $\mathcal{L}$. By way of contradiction, let $\mathbf{F}$ be such that $\mathbf{F} \prec_{\mathcal{L}} \mathbf{M}$. Note that, for each $T \in \text{TxT}_{\mathcal{L}}$, and each n ,

$$\begin{aligned} \text{conv}(\mathbf{M}, \#^{\omega}) &= 1; \\ \text{conv}(\mathbf{M}, 0 \cdot \{\#, 0\}^{\omega}) &= 0; \\ \text{conv}(\mathbf{M}, \# \cdot \#^n \cdot 0 \cdot \{\#, 0\}^{\omega}) &= n + 2. \end{aligned} \quad (35)$$

Let $T \in \text{TxT}_{\mathcal{L}}$ be such that $\text{conv}(\mathbf{F}, T) < \text{conv}(\mathbf{M}, T)$. Clearly, $\text{conv}(\mathbf{M}, T) \geq 1$. Thus, by (35), it suffices to consider the following cases.

CASE $[T = \#^{\omega}]$. Then, by (35), it must be the case that $\text{conv}(\mathbf{F}, T) = 0$ and, thus, $W_{\mathbf{F}(\lambda)} = \emptyset$. It follows that $\text{conv}(\mathbf{F}, 0^{\omega}) > 0$. But then, by (35), $\text{conv}(\mathbf{F}, 0^{\omega}) > \text{conv}(\mathbf{M}, 0^{\omega})$, which contradicts $\mathbf{F} \preceq_{\mathcal{L}} \mathbf{M}$.

CASE $[T \in (\# \cdot \#^n \cdot 0 \cdot \{\#, 0\}^{\omega})$, for some $n]$. Then, by (35), it must be the case that $\text{conv}(\mathbf{F}, T) \leq n + 1$. Furthermore, by Proposition 2, $W_{\mathbf{F}(\#^{n+1})} = \{0\}$. It follows that $\text{conv}(\mathbf{F}, \#^{\omega}) > n + 1 \geq 1$. But then, by (35), $\text{conv}(\mathbf{F}, \#^{\omega}) > \text{conv}(\mathbf{M}, \#^{\omega})$, which contradicts $\mathbf{F} \preceq_{\mathcal{L}} \mathbf{M}$.

(b) $\neq$ (c): Let $\mathcal{L} = \{\emptyset\}$. Let $p_{\emptyset}$ and $p'_{\emptyset}$ be any two distinct grammars for $\emptyset$. Let $\mathbf{M}$ be as follows.

$$\begin{aligned} \mathbf{M}(\lambda) &= p_{\emptyset} \cdot \\ \mathbf{M}(\# \cdot \{\#\}^*) &= p'_{\emptyset} \cdot \end{aligned} \quad (36)$$

Clearly, $\mathbf{M}$ class-preservingly and consistently identifies $\mathcal{L}$. That $\mathbf{M}$ does *not* optimally identify $\mathcal{L}$ is witnessed by, e.g., $\lambda\sigma \cdot p_{\emptyset}$. $\square$ (*Theorem 8*)

Remark 9. Requiring that $\mathbf{F}$ class-preservingly, consistently, and *decisively* [BCM⁺08, CCJS08] identify $\mathcal{L}$ is *not* sufficient to guarantee that $\mathbf{F}$ optimally identifies $\mathcal{L}$, as witnessed by the $\mathbf{M}$ constructed in the proof that (b) $\not\Rightarrow$ (c) of Theorem 8. Requiring that $\mathbf{F}$ class-preservingly, consistently, and *non-U-shapedly* identify $\mathcal{L}$ is similarly insufficient.

Problem 10. Is there an *intuitive* property which is *less* restrictive than strong non-U-shapedness, and which, when combined with class-preservation and consistency, *characterizes* optimality? More formally: does there exist an intuitive predicate $P \subseteq ((\text{SEQ} \rightarrow \mathbb{N}) \times \mathcal{E})$ satisfying (a) through (c) below?

- (a) For each $\mathbf{F}$ and $\mathcal{L}$, if $\mathbf{F}$ strongly non-U-shapedly identifies $\mathcal{L}$, then $P(\mathbf{F}, \mathcal{L})$.
- (b) For each $\mathbf{F}$ and $\mathcal{L}$, if $\mathbf{F}$ class-preservingly and consistently identifies $\mathcal{L}$ and $P(\mathbf{F}, \mathcal{L})$, then $\mathbf{F}$ optimally identifies $\mathcal{L}$.
- (c) For each $\mathbf{F}$ and $\mathcal{L}$, if $\mathbf{F}$ optimally identifies $\mathcal{L}$, then $P(\mathbf{F}, \mathcal{L})$.

One might wonder whether strong non-U-shapedness restricts *algorithmic* learning power, and, if so, whether this contributes to the failure of the analog of Wiehagen’s characterization (Theorem 1 in Section 1). However, as Theorem 12 below states, strong non-U-shapedness does, in fact, *not* restrict algorithmic learning power. The proof is abbreviated, due to space constraints. The complete proof may be found in [CM08a].

The proof of Theorem 12 relies on the following lemma.

Lemma 11. For each $\mathbf{M}$, there exists a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$ such that, for each i , $D_i \subseteq W_{f(i)}$ and $\mathbf{M}$ does *not* identify $W_{f(i)}$.

Proof. See [CM08a].

□ (Lemma 11)

Theorem 12. For each $\mathcal{L}$, $(\exists \mathbf{M})[\mathbf{M} \text{ identifies } \mathcal{L}] \Leftrightarrow (\exists \mathbf{M})[\mathbf{M} \text{ strongly non-U-shapedly identifies } \mathcal{L}]$.⁹

Proof (Sketch). Clearly, $(\exists \mathbf{M})[\mathbf{M} \text{ strongly non-U-shapedly identifies } \mathcal{L}] \Rightarrow (\exists \mathbf{M})[\mathbf{M} \text{ identifies } \mathcal{L}]$. Thus, it suffices to show the converse. Let $\mathcal{L}$ be fixed, and let $\mathbf{M}$ be such that $\mathbf{M}$ identifies $\mathcal{L}$. Without loss of generality, suppose that $\mathbf{M}$ is prudent and total ([Ful90, Theorem 15] and [JORS99, proof of Proposition 4.15]).¹⁰ A machine $\mathbf{M}'$ is constructed such that $\mathbf{M}'$ strongly non-U-shapedly identifies $\mathcal{L}$. (Roughly speaking, certain conjectures of $\mathbf{M}'$ will *self-destruct* when conditions are met that would cause $\mathbf{M}'$ to make a mind-change. See (38).)

Let $f : \mathbb{N} \rightarrow \mathbb{N}$ be as in Lemma 11 for $\mathbf{M}$. For each ρ , σ , s , and τ , let $P(\rho, \sigma, s, \tau) \Leftrightarrow$ there exists α satisfying (a) through (c) below.

- (a) $|\alpha| \leq s$.
- (b) $\text{content}(\alpha) \subseteq W_{\mathbf{M}(\rho)}^s$.
- (c) $(\exists \alpha' \subseteq \alpha \cdot \tau)[\mathbf{M}(\sigma \cdot \alpha') \neq \mathbf{M}(\sigma)]$.

⁹ Theorem 20 in [BCM⁺08] says: for each $\mathcal{L}$, $(\exists \mathbf{M})[\mathbf{M} \text{ identifies } \mathcal{L}] \Leftrightarrow (\exists \mathbf{M})[\mathbf{M} \text{ non-U-shapedly identifies } \mathcal{L}]$. Theorem 12 above is an improvement on this result.

¹⁰ Strictly speaking, the technique employed in the proof of [JORS99, Proposition 4.15] takes an arbitrary learner into a total, *not necessarily prudent* learner. However, it is easily seen that if this technique is applied to an *already* prudent learner, then the resulting learner is prudent, as well.

For each ρ , σ , s , and τ , let $P^*(\rho, \sigma, s, \tau) \Leftrightarrow$ there exists α satisfying (a) through (c) just above, and (d) just below.

$$(d) (\exists s' \leq s) [\emptyset \neq (W_{\mathbf{M}(\rho)}^{s'} - \text{content}(\sigma)) \subseteq \text{content}(\alpha)].$$

Clearly, P and P^* are computable predicates. Intuitively, P helps to determine when a segment of text may be extended in a way that causes $\mathbf{M}$ to make a mind-change. In this sense, the arguments of P play the following roles.

- ρ is used to determine a conjecture of $\mathbf{M}$ (i.e., $\mathbf{M}(\rho)$); the elements used to extend the segment of text σ are drawn from the conjectured language.
- σ is the segment of text to be extended.
- s is used to *bound* the process of searching for an extension, and helps to keep P computable.
- τ is a segment of text that should appear at the end of the extension.

Let $g : (\text{SEQ} \times \mathbb{N}) \rightarrow \text{SEQ}$ be such that, for each ρ and s ,

$$g(\rho, s) = \begin{cases} \lambda, & \text{if } s = 0; \\ \sigma \cdot \alpha, & \text{if } s \neq 0 \wedge P^*(\rho, \sigma, s, \lambda), \text{ where } \sigma = g(\rho, s-1) \\ & \text{and } \alpha \text{ is any as in (a) through (d) above for} \\ & \rho, \sigma, s, \text{ and } \tau (= \lambda); \\ \sigma, & \text{otherwise, where } \sigma = g(\rho, s-1). \end{cases} \quad (37)$$

Clearly, g is computable. Let $g_{\lim} : \text{SEQ} \rightarrow \text{SEQ}$ be such that, for each ρ , $g_{\lim}(\rho) = \lim_{s \rightarrow \infty} g(\rho, s)$. It can be shown that $g_{\lim}$ is well-defined.

By 1-1 s-m-n [Rog67], there exists a 1-1, computable function $h : (\text{SEQ} \times \mathbb{N}) \rightarrow \mathbb{N}$ such that, for each ρ and s ,

$$W_{h(\rho, s)} = \begin{cases} W_{\mathbf{M}(\rho)}, & \text{if } (\forall s' > s)[g(\rho, s') = g(\rho, s)], \\ W_{f(i)}, & \text{otherwise, where } D_i = W_{\mathbf{M}(\rho)}^{s'} \text{ for the least} \\ & s' > s \text{ such that } g(\rho, s') \neq g(\rho, s). \end{cases} \quad (38)$$

For each ρ , s , and τ , let $Q(\rho, s, \tau) \Leftrightarrow$ (a) through (c) below.

- (a) $g(\rho, s) = g(\rho, s + |\tau|)$.
- (b) $\text{content}(g(\rho, s)) \subseteq \text{content}(\tau)$.
- (c) $P(\rho, g(\rho, s), |\tau|, \tau) \Rightarrow \text{content}(\tau) \subseteq \text{content}(g(\rho, s))$.

Clearly, Q is a computable predicate. Many of the conjectures of $\mathbf{M}'$ are of the form $h(\rho, s)$, for some ρ and s . For such conjectures, Q helps to determine appropriate values of ρ and s . Q also helps to determine when such conjectures should be abandoned.

For each τ , let

$$\mathbf{M}'(\tau) = \begin{cases} \mathbf{M}'(\tau^-), & \text{if } (*) [\tau \neq \lambda \\ & \wedge (\exists \rho, s)[\mathbf{M}(\tau^-) = h(\rho, s) \wedge Q(\rho, s, \tau)]]; \\ h(\rho, |\tau|), & \text{where } \rho \subseteq \tau \text{ is shortest such that } Q(\rho, |\tau|, \tau), \\ & \text{if } \neg(*) \text{ and such a } \rho \text{ exists;} \\ f(0), & \text{otherwise.} \end{cases} \quad (39)$$

Clearly, $\mathbf{M}'$ is computable. Let $L \in \mathcal{L}$ and $T \in \text{Txt}_L$ be fixed. The proof that $\mathbf{M}'$ is strongly *non-U-shaped* on T is sketched in Claim 12.1 just below.

Claim 12.1. For each i , if $\mathbf{M}'(T[i]) \neq \mathbf{M}'(T[i+1])$, then $W_{\mathbf{M}'(T[i])} \neq L$.

Proof of Claim (Sketch). By way of contradiction, let i be such that $\mathbf{M}'(T[i]) \neq \mathbf{M}'(T[i+1])$ and $W_{\mathbf{M}'(T[i])} = L$. Then, clearly, there exist $\rho \subseteq T[i]$ and $s \leq i$ satisfying

$$\mathbf{M}'(T[i]) = h(\rho, s) \wedge Q(\rho, s, T[i]) \wedge \neg Q(\rho, s, T[i+1]). \quad (40)$$

Note that

$$\begin{aligned} L &= W_{\mathbf{M}'(T[i])} \quad \{\text{by supposition}\} \\ &= W_{h(\rho, s)} \quad \{\text{since } \mathbf{M}'(T[i]) = h(\rho, s)\}. \end{aligned}$$

Clearly then, by (38), $W_{\mathbf{M}(\rho)} = W_{h(\rho, s)} (= L)$. Consider the following cases (based on $\neg Q(\rho, s, T[i+1])$), each of which leads to a contradiction.

CASE $[g(\rho, s) \neq g(\rho, s+i+1)]$. Then, clearly, by (38), $L \neq W_{h(\rho, s)}$ — a contradiction.

CASE $[\text{content}(g(\rho, s)) \not\subseteq \text{content}(T[i+1])]$. From $Q(\rho, s, T[i])$, it follows that $\text{content}(g(\rho, s)) \subseteq \text{content}(T[i]) \subseteq \text{content}(T[i+1])$. Thus, assuming this case leads to a contradiction.

CASE $[P(\rho, g(\rho, s), i+1, T[i+1]) \wedge \text{content}(T[i+1]) \not\subseteq \text{content}(g(\rho, s))]$. Let α be as in (a) through (c) in the definition of P for $P(\rho, g(\rho, s), i+1, T[i+1])$. Then, in particular, $\text{content}(\alpha) \subseteq W_{\mathbf{M}(\rho)}^{i+1}$ and $(\exists \alpha' \subseteq \alpha \cdot T[i+1])[\mathbf{M}(g(\rho, s) \cdot \alpha') \neq \mathbf{M}(g(\rho, s))]$. From this and the fact that $W_{\mathbf{M}(\rho)} = L$ (shown above), it can be shown that there exists $s' > s$ such that $g(\rho, s') \neq g(\rho, s)$. Clearly, then, by (38), $L \neq W_{h(\rho, s)}$ — a contradiction. $\square$ (Claim 12.1)

The proof that $\mathbf{M}'$ identifies L from T is omitted (see [CM08a]). $\square$ (Theorem 12)

4 Optimal Identification of Subclasses

In this section, we show that, for an arbitrary optimal learner $\mathbf{F}$ of a class of languages $\mathcal{L}$, $\mathbf{F}$ optimally identifies a subclass $\mathcal{K}$ of $\mathcal{L}$ iff $\mathbf{F}$ is class-preserving with respect to $\mathcal{K}$ (Theorem 13 below). The reader may wonder: if $\mathbf{F}$ optimally identifies $\mathcal{L}$, how can there exist a subclass $\mathcal{K}$ of $\mathcal{L}$ such that $\mathbf{F}$ does *not* optimally identify $\mathcal{K}$? Intuitively, this can occur as follows. A learner $\mathbf{G}$, knowing that a language L satisfies $L \in \mathcal{L} - \mathcal{K}$, never outputs a grammar for L . This, in turn, can allow $\mathbf{G}$ to converge to a correct hypothesis on *less* of some $T \in \text{Txt}_{\mathcal{K}}$ for which $(\exists \sigma \in \text{Seq}_L)[\sigma \subset T]$.¹¹

The following is the main result of this section.

Theorem 13. Suppose that $\mathbf{F}$ optimally identifies $\mathcal{L}$. Then,

$$(\forall \mathcal{K} \subseteq \mathcal{L})[\mathbf{F} \text{ optimally identifies } \mathcal{K} \Leftrightarrow \mathbf{F} \text{ class-preservingly identifies } \mathcal{K}]. \quad (41)$$

¹¹ This can be observed in the learners $\mathbf{M}$ and $\mathbf{F}$ given near the end of Section 1.

Proof. Let $\mathbf{F}$ and $\mathcal{L}$ be as stated, and let $\mathcal{K} \subseteq \mathcal{L}$ be fixed.

$(\Rightarrow)$: Immediate by (b) $\Rightarrow$ (c) of Theorem 8.

$(\Leftarrow)$: By way of contradiction, suppose that $\mathbf{F}$ class-preservingly identifies $\mathcal{K}$, but *not* optimally. Let $\mathbf{G}$ be such that $\mathbf{G} \prec_{\mathcal{K}} \mathbf{F}$. Let $\mathbf{H}$ be such that, for each σ ,

$$\mathbf{H}(\sigma) = \begin{cases} \mathbf{G}(\sigma), & \text{if } \sigma \in \text{SEQ}_{\mathcal{K}}; \\ \mathbf{F}(\sigma), & \text{otherwise.} \end{cases} \quad (42)$$

Clearly, $\mathbf{H}$ identifies $\mathcal{L}$ and $\mathbf{H} \prec_{\mathcal{K}} \mathbf{F}$. Thus, if it can be shown that $\mathbf{H} \preceq_{(\mathcal{L}-\mathcal{K})} \mathbf{F}$, then this would (as desired) contradict the fact that $\mathbf{F}$ optimally identifies $\mathcal{L}$. So, suppose $\mathbf{H} \not\preceq_{(\mathcal{L}-\mathcal{K})} \mathbf{F}$. Let $L \in \mathcal{L} - \mathcal{K}$ and $T \in \text{TXT}_L$ be such that $\text{conv}(\mathbf{F}, T) < \text{conv}(\mathbf{H}, T)$. By Lemma 7 (with $A = \text{SEQ} - \text{SEQ}_{\mathcal{K}}$), there exists σ such that $\sigma \in \text{SEQ}_{\mathcal{K}}$ and $T[\text{conv}(\mathbf{F}, T)] \subseteq \sigma \subset T$. By the latter and Proposition 2, $W_{\mathbf{F}(\sigma)} = L$ ($\notin \mathcal{K}$), which contradicts the supposition that $\mathbf{F}$ class-preservingly identifies $\mathcal{K}$. $\square$ (*Theorem 13*)

Remark 14. One might hope for a characterization similar to Theorem 13, but involving only $\mathcal{K}$ and $\mathcal{L}$ (and *not* $\mathbf{F}$) on the right-hand side of the “ $\Leftarrow$ ”, i.e., $\mathbf{F}$ optimally identifies $\mathcal{K} \Leftrightarrow P(\mathcal{K}, \mathcal{L})$, for some predicate $P \subseteq (2^{2^{\mathbb{N}}} \times 2^{2^{\mathbb{N}}})$. However, such a characterization is not possible, as the following example demonstrates. Let $\mathcal{L} = \{\{0\}, \{1\}\}$. Let $p_{\{0\}}$ and $p_{\{1\}}$ be grammars for $\{0\}$ and $\{1\}$, respectively. For each σ , let $\mathbf{M}_0$ and $\mathbf{M}_1$ be as follows.

$$\mathbf{M}_0(\sigma) = \begin{cases} p_{\{0\}}, & \text{if } \text{content}(\sigma) \subseteq \{0\}; \\ p_{\{1\}}, & \text{otherwise.} \end{cases} \quad (43)$$

$$\mathbf{M}_1(\sigma) = \begin{cases} p_{\{1\}}, & \text{if } \text{content}(\sigma) \subseteq \{1\}; \\ p_{\{0\}}, & \text{otherwise.} \end{cases} \quad (44)$$

It is easy to verify that both $\mathbf{M}_0$ and $\mathbf{M}_1$ optimally identify $\mathcal{L}$. However, if one lets $\mathcal{K} = \{\{0\}\}$, then $\mathbf{M}_0$ optimally identifies $\mathcal{K}$; whereas, $\mathbf{M}_1$ does *not*.

Despite Remark 14, Corollary 15 below gives a useful *necessary* condition similar to Theorem 13. Moreover, this condition involves only $\mathcal{K}$ and $\mathcal{L}$ (and *not* $\mathbf{F}$) on the right-hand side of the “ $\Rightarrow$ ”.

Corollary 15. Suppose that $\mathbf{F}$ optimally identifies $\mathcal{L}$. Then,

$$(\forall \mathcal{K} \subseteq \mathcal{L}) \left[\mathbf{F} \text{ optimally identifies } \mathcal{K} \Rightarrow (\forall L, L' \in \mathcal{L}) [[L \notin \mathcal{K} \wedge L \subseteq L'] \Rightarrow L' \notin \mathcal{K}] \right]. \quad (45)$$

Proof of Corollary. Let $\mathbf{F}$, $\mathcal{L}$, and $\mathcal{K}$ be as stated, and suppose that $\mathbf{F}$ optimally identifies $\mathcal{L}$. Further suppose, by way of contradiction, that $L, L' \in \mathcal{L}$ are such that $[L \notin \mathcal{K} \wedge L \subseteq L' \wedge L' \in \mathcal{K}]$. Let $T \in \text{TXT}_L$ be fixed, and let $\sigma = T[\text{conv}(\mathbf{F}, T)]$. Clearly, $W_{\mathbf{F}(\sigma)} = L$ ($\notin \mathcal{K}$). Furthermore, since $L \subseteq L'$, $\sigma \in \text{SEQ}_{L'}$. Thus, $\mathbf{F}$ does *not* class-preservingly identify $\mathcal{K}$, which contradicts $(\Rightarrow)$ of Theorem 13. $\square$ (*Corollary 15*)

Acknowledgements. The authors would like to thank: James Royer, whose insightful comments inspired much of the work herein; and Timo Kötzing, whose careful eye caught errors in early versions of the proof of Theorem 12.

References

- [Ang80] Angluin, D.: Inductive inference of formal languages from positive data. *Information and Control* 45(2), 117–135 (1980)
- [Bar77] Barzdin, J.M.: Inductive inference of automata, functions, and programs. *American Mathematical Society Translations: Series 2* 109, 107–112 (1977); In: *Proceedings of the 20th International Conference of Mathematicians* (1974) (appeared originally) (in Russian)
- [BB75] Blum, L., Blum, M.: Toward a mathematical theory of inductive inference. *Information and Control* 28(2), 125–155 (1975)
- [BCM⁺08] Baliga, G., Case, J., Merkle, W., Stephan, F., Wiehagen, W.: When unlearning helps. *Information and Computation* 206(5), 694–709 (2008)
- [BF74] Bärzdiņš, J., Freivalds, R.: Prediction and limiting synthesis of recursively enumerable classes of functions. *Latvijas Valsts Univ. Zinatn. Raksti* 210, 101–111 (1974) (in Russian)
- [CCJS07] Carlucci, L., Case, J., Jain, S., Stephan, F.: Results on memory-limited U-shaped learning. *Information and Computation* 205(10), 1551–1573 (2007)
- [CCJS08] Carlucci, L., Case, J., Jain, S., Stephan, F.: Non-U-shaped vacillatory and team learning. *Journal of Computer and Systems Sciences* 74(4), 409–430 (2008)
- [CM08a] Case, J., Moelius, S.E.: Optimal language learning (expanded version). Technical report, University of Delaware (2008), <http://www.cis.udel.edu/~moelius/publications>
- [CM08b] Case, J., Moelius, S.E.: U-shaped, iterative, and iterative-with-counter learning. *Machine Learning* 72(1-2), 63–88 (2008)
- [CS83] Case, J., Smith, C.: Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science* 25(2), 193–220 (1983)
- [Ful90] Fulk, M.: Prudence and other conditions on formal language learning. *Information and Computation* 85(1), 1–11 (1990)
- [Gol67] Gold, E.M.: Language identification in the limit. *Information and Control* 10(5), 447–474 (1967)
- [JORS99] Jain, S., Osherson, D., Royer, J., Sharma, A.: *Systems that Learn: An Introduction to Learning Theory*, 2nd edn. MIT Press, Cambridge (1999)
- [OSW86] Osherson, D., Stob, M., Weinstein, S.: *Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists*, 1st edn. MIT Press, Cambridge (1986)
- [Pit89] Pitt, L.: Inductive inference, DFAs, and computational complexity. In: Jantke, K.P. (ed.) *AII 1989. LNCS*, vol. 397, pp. 18–44. Springer, Heidelberg (1989)
- [Rog67] Rogers, H.: *Theory of Recursive Functions and Effective Computability*. McGraw Hill, New York (1967) (Reprinted, MIT Press, 1987)
- [Wie91] Wiehagen, R.: A thesis in inductive inference. In: Dix, J., Schmitt, P.H., Jantke, K.P. (eds.) *NIL 1990. LNCS*, vol. 543, pp. 184–207. Springer, Heidelberg (1991)

Numberings Optimal for Learning

Sanjay Jain^{1,*} and Frank Stephan^{2,*}

¹ Department of Computer Science,
National University of Singapore, Singapore 117543, Republic of Singapore
sanjay@comp.nus.edu.sg

² Department of Computer Science and Department of Mathematics,
National University of Singapore, Singapore 117543, Republic of Singapore
fstephan@comp.nus.edu.sg

Abstract. This paper extends previous studies on learnability in non-acceptable numberings by considering the question: for which criteria which numberings are optimal, that is, for which numberings it holds that one can learn every learnable class using the given numbering as hypothesis space. Furthermore an effective version of optimality is studied as well. It is shown that the effectively optimal numberings for finite learning are just the acceptable numberings. In contrast to this, there are non-acceptable numberings which are optimal for finite learning and effectively optimal for explanatory, vacillatory and behaviourally correct learning. The numberings effectively optimal for explanatory learning are the K -acceptable numberings. A similar characterization is obtained for the numberings which are effectively optimal for vacillatory learning. Furthermore, it is studied which numberings are optimal for one and not for another criterion: among the criteria of finite, explanatory, vacillatory and behaviourally correct learning all separations can be obtained; however every numbering which is optimal for explanatory learning is also optimal for consistent learning.

1 Introduction

Consider the following model of learning. The learner receives, over time, more and more data about the concept to be learnt. From time to time, the learner conjectures a potential explanation for the data it is receiving. One can say that the learner learns the concept if the sequence of conjectures eventually converges to a correct explanation for the concept. This is essentially the notion of explanatory learning considered by Gold [9]. The concepts considered are usually recursively enumerable (r.e.) languages (subsets of natural numbers) or computable functions. In this paper we will be concentrating on learning languages. The explanations thus take the form of grammars or indices from some hypothesis space or numbering of recursively enumerable languages.

Learning of just one r.e. language is not useful, as a learner which just conjectures a grammar for the language, on any data, would be successful on the

* Supported in part by NUS grant number R252-000-308-112.

language. Thus, it is more useful to consider learnability of a class of languages. A learner explanatorily learns a class of languages if it explanatorily learns each language in the class. Since Gold's paper [9], several other criteria of learnability have been explored and some of them will be considered in the current paper.

The learnability of the class depends not only on the class itself but also on the underlying numbering used as a hypothesis space. Angluin [1] initiated the systematic study of uniformly recursive hypothesis spaces; as such hypothesis spaces can contain only some but not all recursive sets, these spaces have to be selected in dependence of the class to be learnt. Lange and Zeugmann [12, 13, 20] investigated the topic thoroughly. De Jongh and Kanazawa [5] investigated to which extent one can generalize Angluin's characterization of learnability [1] from uniformly recursive to uniformly r.e. hypothesis spaces. Zilles [21] studied the question how to synthesize a learner from an index of a uniformly r.e. hypothesis space. Most of this and related work considered specialized hypothesis spaces, which permit only to learn some and not all classes; these specialized hypothesis spaces often do not even contain all r.e. sets.

In contrast to this, the focus of the present work lies on the question which hypothesis spaces are optimal for learning in the sense that every learnable class can be learnt using this hypothesis space. Therefore, a valid hypothesis space $A_0, A_1, A_2, \dots$ must satisfy that $\{\langle e, x \rangle : x \in A_e\}$ is recursively enumerable and that, for every r.e. set B , there is an index e with $B = A_e$. In particular, acceptable, K -acceptable and Ke -numberings are considered. (Here a numbering $A_0, A_1, A_2, \dots$ is acceptable (K -acceptable) if for every further numbering $B_0, B_1, B_2, \dots$ there is a recursive (K -recursive) function f such that $B_e = A_{f(e)}$ for all e . A Ke -numbering is a numbering of all r.e. sets for which the grammar equivalence problem is K -recursive.) A more restrictive notion is that of an effectively optimal hypothesis space where additionally one can effectively obtain a learner for the class using $A_0, A_1, A_2, \dots$ as hypothesis space from any learner for the class using another numbering $B_0, B_1, B_2, \dots$ as hypothesis space.

The optimality of the hypothesis space depends on the criterion of learning considered. The main criteria considered are finite, explanatory, vacillatory and behaviourally correct learning as defined below in Definition 1; but some interesting results are also obtained for other criteria of learning.

Intuitively, a learner M finitely learns [9] a language class if, for every language L in the class, for any order of presentation of elements of L , M outputs only one conjecture and the conjecture is an index for L . A learner M explanatorily learns [9] a language class if, for every language L in the class, for any order of presentation of elements of L , M outputs a sequence of conjectures which converges to an index for L . A learner M behaviourally correctly learns [2, 16] a language class if, for every language L in the class, for any order of presentation of elements of L , M outputs an infinite sequence of conjectures, all but finitely many of which are indices for L . Vacillatory learning [4] is a restriction of behaviourally correct learning, where the learner outputs only finitely many distinct conjectures (although some of them might be repeated infinitely often).

The most prominent numberings are the acceptable numberings and Friedberg numberings. Acceptable numberings are used by many authors as the standard hypothesis space [9] and every learnable class (according to most criteria) is also learnable using an acceptable numbering — one exception is the criterion of learning with additional information, see Theorem 23. However, one-one numberings, also known as Friedberg numberings [7], are not optimal for learning [6, 11]. A central contribution of the present work is to show that there are many optimal numberings besides the acceptable numberings, but that it depends a lot on the underlying learning criterion which numberings are optimal for learning and which are not. For example, a nearly acceptable numbering (as defined in Definition 4) is effectively optimal for explanatory, vacillatory and behaviourally correct learning as well as optimal for finite learning (see Proposition 5).

In Theorem 6, we show characterizations for numberings which are effectively optimal for finite, explanatory and vacillatory learning. In particular, a numbering $A_0, A_1, A_2, \dots$ is effectively optimal for finite learning iff the numbering is acceptable. A numbering $A_0, A_1, A_2, \dots$ is effectively optimal for explanatory learning iff the numbering is K -acceptable. One can also similarly characterize effectively optimal numberings for vacillatory learning. We do not have a good characterization of numberings which are effectively optimal for behaviourally correct learning.

We show that there are numberings which are (non-effectively) optimal but not effectively optimal for various criteria of inference: Theorem 9 gives this result for finite learning; Theorem 13 gives this result for explanatory and vacillatory learning; Theorem 14 gives this result for behaviourally correct learning.

We also show that the set of optimal numberings for finite, explanatory, vacillatory and behaviourally correct learning are incomparable. Theorem 9 gives this result for finite learning versus explanatory, vacillatory and behaviourally correct learning. Theorem 12 gives this result for behaviourally correct learning versus finite, explanatory and vacillatory learning. Theorem 11 gives this result for explanatory and vacillatory learning versus finite learning and behaviourally correct learning. Theorem 10 gives this result for vacillatory learning versus explanatory learning. The numbering $A_0, A_1, A_2, \dots$ in Theorem 13 gives this result for explanatory learning versus vacillatory learning.

In Section 4 we give special attention to consistent learning. Theorem 20 shows that optimal numberings for explanatory learning are optimal for consistent learning. This is one of the rare cases of an inclusion in the sense that every numbering optimal for a criterion I is also optimal for a different criterion J . The inclusion also holds with effective optimality in place of optimality. However, there are numberings which are effectively optimal for consistent learning but not optimal for finite, explanatory, vacillatory or behaviourally correct learning.

2 Preliminaries

For the ease of notation, learnability of r.e. subsets of the natural numbers, $\mathbb{N}$, is studied (and other possible domains are ignored). The learners use some hypothesis space to represent their conjectures.

The standard hypothesis space $W_0, W_1, W_2, \dots$ is some fixed *acceptable numbering* [17], that is, for every further numbering $A_0, A_1, A_2, \dots$ of r.e. sets, there is a recursive function f with $W_{f(e)} = A_e$ for all e . In general, every numbering of all r.e. sets can be a hypothesis space. A numbering $A_0, A_1, A_2, \dots$ is called *K-acceptable* iff, for every further numbering $B_0, B_1, B_2, \dots$, there is a *K*-recursive function f with $A_{f(e)} = B_e$ for all e . Here *K* denotes the halting problem $\{x : x \in W_x\}$.

A *text* T is a member of $(\mathbb{N} \cup \{\#\})^\infty$. $T(0), T(1), T(2)$ and so on denote the members of T ; $T[n]$ denotes $T(0)T(1)\dots T(n-1)$. A *sequence* σ is a member of $(\mathbb{N} \cup \{\#\})^*$. λ denotes an empty sequence. For a text T let $\text{content}(T) = \{T(n) : n \in \mathbb{N} \wedge T(n) \in \mathbb{N}\}$; similarly one defines $\text{content}(\sigma)$. The length of a sequence σ , denoted $|\sigma|$, is the number of elements in the domain of σ . One says that $\sigma \preceq T$ and $\sigma \preceq \tau$ iff σ is a prefix of T and τ , respectively. Furthermore, T is a *text for L* iff $L = \text{content}(T)$. Note that there is a uniformly recursive method for generating a text T_e for W_e ; this text T_e is called a *canonical text* for W_e .

The general model of learning is that the learner M assigns, to every prefix $T[n]$ of a given text T for the set L to be learnt, an index $M(T[n])$ interpreted as M 's conjecture for the language L ; for finite learning, the learner M is allowed to output a special symbol “?” which denotes that the learner does not wish to make a conjecture at this point. One says that a learner M converges on a text T to an index e (denoted $M(T) = e$) iff $M(T[n]) = e$ for almost all n . Furthermore, one says that M outputs an index e on T iff there is an n with $M(T[n]) = e$. The following definition gives various criteria of learning.

Definition 1 (Bārzdis [2], Case [4], Gold [9], Osherson and Weinstein [16]). A class S is *finitely learnable using a numbering* $A_0, A_1, A_2, \dots$ [9] iff there is a learner which, for every $L \in S$ and every text T for L , outputs exactly one index e , besides ?, on T and this index e satisfies $A_e = L$.

A class S is *explanatorily learnable using a numbering* $A_0, A_1, A_2, \dots$ [9] iff there is a learner which, for every $L \in S$ and every text T for L , converges on T to an index e such that $A_e = L$.

A class S is *vacillatorily learnable using a numbering* $A_0, A_1, A_2, \dots$ [4] iff there is a learner which, for every $L \in S$ and every text T for L , converges on T to an index d such that, for some $e \leq d$, $A_e = L$.

A class S is *behaviourally correctly learnable using a numbering* $A_0, A_1, A_2, \dots$ [2, 16] iff there is a learner M which, for every $L \in S$ and every text T for L , satisfies $A_{M(T[n])} = L$ for almost all n . Note that it is permitted, but not required, that the $M(T[n])$ are syntactically different.

Note that definition of vacillatorily learnable, as defined by Case [4], requires the learner to eventually output its conjecture only from finitely many correct indices for the input language — that is, the learner eventually vacillates among only finitely many correct indices for the input language. The definition used above is equivalent to this definition and is a useful characterization of vacillatory learning. We defined vacillatory learning using this characterization mainly because of its ease of use in the proofs below.

For ease of notation below, if the hypothesis space is not specified, then the default numbering $W_0, W_1, W_2, \dots$ is assumed as hypothesis space.

Definition 2 (Blum and Blum [3], Fulk [8]). A *stabilizing sequence* for M on L is a sequence σ such that $\text{content}(\sigma) \subseteq L$ and $M(\sigma\tau) = M(\sigma)$ for all $\tau \in (L \cup \{\#\})^*$. A *locking sequence* for M on L is a stabilizing sequence σ for M on L such that $M(\sigma)$ is an index for L (in the hypothesis space used).

Note that by the locking-sequence hunting construction [3, 8] there is a recursive enumeration of learners $M_0, M_1, M_2, \dots$ such that (a) every explanatorily learnable class is learnt by one of these learners, (b) whenever M_e converges on some text for L , then M_e converges on all texts for L to the same index, (c) whenever M_e learns L and T is a text for L , then for some n , $T[n]$ is a locking sequence for M_e on L .

Definition 3. A numbering $A_0, A_1, A_2, \dots$ is called *optimal for explanatory learning* iff every explanatorily learnable class can be learnt using the numbering $A_0, A_1, A_2, \dots$ as hypothesis space; a numbering $A_0, A_1, A_2, \dots$ is called *effectively optimal for explanatory learning* iff for every numbering $B_0, B_1, B_2, \dots$ one can effectively convert explanatory learners using $B_0, B_1, B_2, \dots$ into explanatory learners using $A_0, A_1, A_2, \dots$ as hypothesis space. Similarly, one can also define optimality and effective optimality for other learning criteria.

As $W_0, W_1, W_2, \dots$ is acceptable, for showing (effective) optimality of $A_0, A_1, A_2, \dots$, it is sufficient to consider converting learners using $W_0, W_1, W_2, \dots$ to learners using $A_0, A_1, A_2, \dots$ as hypothesis space.

Let C be the plain Kolmogorov complexity [14] and C^K be the Kolmogorov complexity relative to K . Note that C^K can be approximated from above relative to K . Let C_s^K denote an approximation of C^K relative to K . Approximations from above or from below relative to K have an easy characterization: A function g can be approximated from above (below) relative to K iff there are uniformly recursive functions g_s with $g(n) = \limsup_s g_s(n)$ for all n (respectively, $g(n) = \liminf_s g_s(n)$ for all n).

$\langle \cdot, \cdot \rangle$ denotes a recursive bijection from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$. Here we assume that $\langle \cdot, \cdot \rangle$ is increasing in both its arguments.

3 Optimality and Effective Optimality

The following notion generalizes the notion of acceptable numberings; it is an example of a natural class of numberings which goes beyond acceptable numberings but is still optimal for most of the learning criteria studied in the literature.

Definition 4. A numbering $A_0, A_1, A_2, \dots$ is called *nearly acceptable* iff there is a recursive function f such that $A_{f(d,e)} = W_e$ whenever $d \in W_e$.

Proposition 5. Let $A_0, A_1, A_2, \dots$ be given by the equations

$$\begin{aligned} A_0 &= \emptyset; \\ A_{\langle d,e \rangle + 1} &= W_e \cup \{d\}. \end{aligned}$$

The numbering $A_0, A_1, A_2, \dots$ is nearly acceptable but not acceptable. Furthermore, every nearly acceptable numbering is optimal for finite learning and effectively optimal for explanatory, vacillatory and behaviourally correct learning.

The effectively optimal numberings for finite, explanatory and vacillatory learning are easy to characterize.

Theorem 6. *A numbering $A_0, A_1, A_2, \dots$ of all r.e. sets is*

- (a) *effectively optimal for finite learning iff it is acceptable;*
- (b) *effectively optimal for explanatory learning iff it is K -acceptable;*
- (c) *effectively optimal for vacillatory learning iff there is a limit-recursive function g such that, for all d , there is an $e \leq g(d)$ with $A_e = W_d$.*

Proof. The necessity in the conditions (a) and (b) stems from the fact that, for each set W_d one can make a learner using $W_0, W_1, W_2, \dots$ which always conjectures d ; this learner can then be effectively converted into a learner using $A_0, A_1, A_2, \dots$, which is then simulated on the canonical text T_d for W_d . The simulated learner will reveal — in case (a) directly and in case (b) in the limit — an e with $A_e = W_d$. In the case of vacillatory learning, one can similarly obtain an upper bound $g(d)$ of an e with $A_e = W_d$.

We now consider sufficiency. For case (a), one can just translate all hypotheses and so finite learnability is preserved. In case (b) one can translate all hypotheses in the limit and thus preserve explanatory learnability. In case (c) and given a vacillatory learner M using $W_0, W_1, W_2, \dots$ for the desired class, on input T for a language L in the class, one can find in the limit $\max\{g(i) : i \leq M(T)\}$, which is an upper bound for the smallest e with $A_e = L$, as there is a $d \leq M(T)$ with $W_d = L$. $\square$

We now turn our attention to the separation of effectively and non-effectively optimal numberings, as well as the separation of optimal numberings for various criteria of inference. The following propositions are useful for showing some of our results.

Proposition 7. *If S is a finitely learnable class, then there is a number d such that almost all members of S have at least 2 non-elements below d .*

Proof. Suppose M finitely learns S . Fix an $L \in S$. Let σ be such that $\text{content}(\sigma) \subseteq L$ and $M(\sigma)$ is an index for L . Let $d_1 = \max\{\text{content}(\sigma)\}$. Note that for $L' \in S$ with $L \neq L'$, $\text{content}(\sigma) \not\subseteq L'$. The reason is that otherwise M does not finitely learn $\{L, L'\}$. Thus, for all $L' \in S - \{L\}$, there exists an $i \leq d_1$, such that $i \notin L'$.

Let $S_i = \{L' \in S : i \notin L'\}$. For non-empty S_i , let L_i be a fixed member of S_i and let σ_i be such that $\text{content}(\sigma_i) \subseteq L_i$ and $M(\sigma_i)$ is an index for L_i . Let $d_2 = \max\{\max\{\text{content}(\sigma_i)\} : i \leq d_1 \wedge S_i \neq \emptyset\}$. Note that if $S_i \neq \emptyset$, then for $L' \in S$ with $L_i \neq L'$, $\text{content}(\sigma_i) \not\subseteq L'$; thus, for all $L' \in S - \{L_i\}$, there exists a $j \leq d_2$, such that $j \neq i$ and $j \notin L'$.

Thus, for all $L' \in S - (\{L\} \cup \{L_i : S_i \neq \emptyset, i \leq d_1\})$, the set $(\mathbb{N} - L') \cap \{x : x \leq d_1 + d_2\}$ contains at least two elements. $\square$

For any n and n distinct numbers $a_1, a_2, \dots, a_n$, let $D_e = \{a_1, a_2, \dots, a_n\}$ iff $e = 2^{a_1} + 2^{a_2} + \dots + 2^{a_n}$; furthermore, let $D_0 = \emptyset$. The number e is called the canonical index of D_e .

Proposition 8. *Let a uniformly K -recursive one-one listing $L_0, L_1, L_2, \dots$ of cofinite sets be given such that $i = \min(\mathbb{N} - L_i)$ for all i . Then there is a numbering $H_0, H_1, H_2, \dots$ of r.e. sets and a (non-recursive) function g such that for all i, j :*

- $\forall i > 0 [D_i \subseteq H_i \subseteq D_i \cup \{\max(D_i), \max(D_i) + 1, \max(D_i) + 2, \dots\}]$;
- $\forall i [H_{g(i)} = L_i]$;
- $\forall i [i \notin \{g(0), g(1), g(2), \dots\} \Rightarrow H_i \text{ is finite}]$;
- $\forall i, j [\text{if } C^K(j) \leq 2^i, \text{ then } j < g(i)]$.

The function g can be approximated from below relative to K .

Proof. The function g is defined by the following K -recursive approximation:

- in stage 0: choose $g_0(i)$ such that $D_{g_0(i)} = \{0, 1, 2, \dots, i, i + 1\} - \{i\}$;
- in stage $s + 1$: if there is an x with $\max(D_{g_s(i)}) \leq x \leq s$ and $(x \notin L_i \text{ or } C_s^K(x) \leq 2^i)$,
then choose $g_{s+1}(i)$ such that $D_{g_{s+1}(i)} = \{s + 1\} \cup (L_i \cap \{0, 1, 2, \dots, s\})$
else let $g_{s+1}(i) = g_s(i)$.

Note that whenever $g_{s+1}(i) \neq g_s(i)$, then $\max(D_{g_{s+1}(i)}) > s$ and hence $g_{s+1}(i) > s$. It follows that the set $G = \mathbb{N} - \{g(0), g(1), g(2), \dots\}$ is K -r.e.; that is, there is a recursive approximation with $i \in G \Leftrightarrow \forall^\infty s [i \in G_s]$. Let $H_0 = \emptyset$; for $i > 0$, let

$$H_i = D_i \cup \{t : \exists s [\max(D_i) \leq t \leq s \wedge i \notin G_s]\}.$$

The sets $H_0, H_1, H_2, \dots$ are uniformly r.e.; furthermore, H_i is finite iff $i \in G_s$ for almost all s . In the case that $i = g(j)$ it follows that $\max(D_i)$ is an upper bound on all non-elements of L_j and therefore $H_i = L_j$. This completes the proof. $\square$

Theorem 9. *There is a numbering which is optimal but not effectively optimal for finite learning. This numbering is not optimal for explanatory, vacillatory and behaviourally correct learning.*

Proof. Let $L_e = \mathbb{N} - \{e\}$ for all e ; then choose the numbering $H_0, H_1, H_2, \dots$ according to Proposition 8. Now let $A_{\langle 0,0 \rangle} = \mathbb{N}$ and $A_{\langle 0,e+1 \rangle} = H_e$ for all e . Furthermore, for every e and every $d > 0$, let

$$A_{\langle d,e \rangle} = \bigcup_{s: |\{0,1,2,\dots,d\} - W_{e,s}| \geq 2} W_{e,s}.$$

Note that the resulting numbering covers all r.e. sets: first $\mathbb{N}$ and every set of the form $\mathbb{N} - \{a\}$ is covered by sets of the form $A_{\langle 0,e \rangle}$; second, a set W_e with least non-elements a, b is equal to $A_{\langle d,e \rangle}$ for all $d > a + b$.

Now let S be a finitely learnable class with learner M . Note that if $\mathbb{N} \in S$, then S contains no other languages and thus finite learnability in numbering A

would be trivial. So assume $\mathbb{N} \notin S$. By Proposition 7 there is a number d such that all but finitely many members of S have at least 2 non-elements below d . Without loss of generality, d is so large that these exceptions are all of the form $\mathbb{N} - \{c\}$ with $c \leq d$. Now one builds a new learner N as follows:

- $N(\sigma)$ is an index $\langle 0, e_c \rangle$ for the set $\mathbb{N} - \{c\}$ whenever $\{0, 1, 2, \dots, d\} - \{c\} \subseteq \text{content}(\sigma) \subseteq \mathbb{N} - \{c\}$ and $\mathbb{N} - \{c\} \in S$.
- $N(\sigma) = \langle d, M(\sigma) \rangle$ whenever $M(\sigma)$ is defined (that is, $M(\sigma) \neq ?$) and $c \in \text{content}(\sigma)$ for all c with $\mathbb{N} - \{c\} \in S$.
- $N(\sigma) = ?$, otherwise.

It is easy to see that N is a finite learner for S .

Note that, by the definition of $H_0, H_1, H_2, \dots$ and $A_0, A_1, A_2, \dots$, each of the sets $\mathbb{N} - \{c\}$ have exactly one index $\langle 0, e_c \rangle$ (with respect to $A_0, A_1, A_2, \dots$), which also satisfies $C^K(e_c) > 2^c$. It follows that the class $\{\mathbb{N} - \{c\} : c \in \mathbb{N}\}$ is not behaviourally correctly learnable using $A_0, A_1, A_2, \dots$, as otherwise $C^K(e_c)$ would, for every c , be bounded by c plus a constant independent of c . As $\{\mathbb{N} - \{c\} : c \in \mathbb{N}\}$ is explanatorily learnable, it follows that $A_0, A_1, A_2, \dots$ is not optimal for explanatory, vacillatory and behaviourally correct learning.

Furthermore, $A_0, A_1, A_2, \dots$ is not acceptable as $A_0, A_1, A_2, \dots$ contains only one index for each set of the form $\mathbb{N} - \{c\}$. Thus, by Theorem 6, $A_0, A_1, A_2, \dots$ is not effectively optimal for finite learning. $\square$

Theorem 10. *There is a numbering $A_0, A_1, A_2, \dots$ which is effectively optimal for vacillatory learning but not optimal for explanatory learning.*

Proof. Recall that C^K is the Kolmogorov complexity relative to K . Let $C_s^{K_s}$ be an approximation of C^K after s steps such that, for all x , $C^K(x) = \limsup C_s^{K_s}(x)$ and for all s and c , there are less than 2^c numbers y with $C_s^{K_s}(y) < c$. Now let

$$A_{\langle d, e \rangle} = \bigcup_{s: C_s^{K_s}(d) > 2^e} W_{e, s}.$$

Then $A_{\langle d, e \rangle}$ is finite for those d and e where $C^K(d) \leq 2^e$. Furthermore, for every e and all sufficiently large d it holds that $C^K(d) > 2^e$.

No class S containing infinitely many infinite sets is explanatorily learnable using this numbering. The reason is that given the least index e of an infinite member of the class, the learner would converge on the canonical text of W_e to an index of Kolmogorov complexity (relative to K) at most a constant above that of e ; however every index in the given numbering A for W_e would have Kolmogorov complexity (relative to K) at least 2^e minus a constant. Hence such an explanatory learner cannot exist. As there exist explanatorily learnable classes (such as $\{\langle e, x \rangle : x \in \mathbb{N}\} : e \in \mathbb{N}\}$) containing infinitely many infinite sets, it follows that $A_0, A_1, A_2, \dots$ is not optimal for explanatory learning.

On the other hand, one can use Theorem 6 to obtain that the numbering considered is effectively optimal for vacillatory learning: the reason is that for every e there is a $d \leq 2^{e+1}$ with $C^K(d) > 2^e$ and $A_{\langle d, e \rangle} = W_e$. $\square$

Theorem 11. *There is a numbering $A_0, A_1, A_2, \dots$ which is effectively optimal for explanatory and vacillatory learning but not optimal for behaviourally correct learning or finite learning.*

Proof. Recall that a simple set [17] is r.e., co-infinite and intersects every infinite r.e. set. Let $a_0, a_1, a_2, \dots$ be a recursive one-one enumeration of a simple set B and define

$$A_{\langle d, e \rangle} = \begin{cases} W_e, & \text{if } d \notin B \wedge d > e; \\ \{0, 1, 2, \dots, 2^{s+1} \cdot 3^d \cdot 5^e\}, & \text{if } d = a_s \wedge d > e; \\ \{0, 1, 2, \dots, 3^d \cdot 5^e\}, & \text{if } d \leq e. \end{cases}$$

This numbering is a K -acceptable numbering as, for every e , one can find the least $d \notin B \cup \{0, 1, 2, \dots, e\}$ using the oracle K and then $A_{\langle d, e \rangle} = W_e$. By Theorem 6, the numbering is effectively optimal for explanatory and vacillatory learning.

It remains to show that the numbering is not optimal for behaviourally correct learning or finite learning.

Consider any class S of infinite languages which is behaviourally correctly learnable but not vacillatorily learnable using $W_0, W_1, W_2, \dots$ as a hypothesis space [4]. Suppose that M using the numbering $A_0, A_1, A_2, \dots$ behaviourally correctly learns S . As S is not vacillatorily learnable, it follows by a result of Case [4] that there are $L \in S$ and a recursive text T for L on which the learner M outputs infinitely many distinct conjectures. For every pair $\langle d, e \rangle$ with $d \in \{0, 1, 2, \dots, e\} \cup B$, the set $A_{\langle d, e \rangle}$ is a finite set and has a maximum which is a multiple of $3^d \cdot 5^e$. Hence M outputs only finitely many of these pairs on the text T . Now let E be the infinite r.e. set of all indices $\langle d, e \rangle$ output by M on T such that $d \notin \{0, 1, 2, \dots, e\} \cup B$. The set $\{d : \exists e [\langle d, e \rangle \in E]\}$ is an r.e. set disjoint to B and hence finite. As for every e there are only pairs $\langle d, e \rangle$ with $d > e$ in E , it follows that E is finite as well in contradiction to the assumption.

From this contradiction it can be concluded that M is not a behaviourally correct learner for S and the numbering $A_0, A_1, A_2, \dots$ is not optimal for behaviourally correct learning.

Consider $L_n = \{\langle n, x \rangle : x \in \mathbb{N}\}$. Clearly, $\{L_0, L_1, L_2, \dots\}$ is finitely learnable using $W_0, W_1, W_2, \dots$ as hypothesis space. Suppose by way of contradiction that some learner finitely learns $\{L_0, L_1, L_2, \dots\}$ using $A_0, A_1, A_2, \dots$ as hypothesis space. Then, given n , one can effectively find an index $\langle d_n, e_n \rangle$ such that $A_{\langle d_n, e_n \rangle} = L_n$. In particular, $d_n \notin B$, $d_n > e_n$ and $W_{e_n} = L_n$. Note that all e_n are distinct. But then the set $\{d_n : n \in \mathbb{N}\}$ is an infinite r.e. set disjoint from B , a contradiction to B being a simple set. Thus, $\{L_0, L_1, L_2, \dots\}$ is not finitely learnable using $A_0, A_1, A_2, \dots$ as hypothesis space. $\square$

Theorem 12. *The numbering $A_0, A_1, A_2, \dots$ given by*

$$A_{\langle d, e \rangle} = \bigcup_{s: \exists m [m = \min(W_{e,s}) \wedge (d > |W_{m,s}| \vee |W_{e,s}| \leq |W_m|)]} W_{e,s}$$

is effectively optimal for behaviourally correct learning but neither optimal for explanatory nor for vacillatory nor for finite learning.

Proof. The behaviourally correct learner N using $A_0, A_1, A_2, \dots$ is effectively built by simulating a given learner M using the numbering $W_0, W_1, W_2, \dots$ and defining $N(\sigma) = \langle |\sigma|, M(\sigma) \rangle$. Given a text T for a set L learnt by M , use e_d as short hand for $M(T[d])$ and note that $N(T[d]) = \langle d, e_d \rangle$. The learner N succeeds as shown in the following case distinction.

- $L = \emptyset$: then almost all e_d are indices of the empty set and hence $A_{\langle d, e_d \rangle}$ is empty for almost all d as well.
- $m = \min(L)$ exists and W_m is infinite: then $A_{\langle d, e_d \rangle} = W_{e_d}$ for all d where W_{e_d} is correct. Hence N behaviourally correctly learns L as well.
- $m = \min(L)$ exists and W_m is finite: then $A_{\langle d, e_d \rangle} = W_{e_d}$ for all d where W_{e_d} is correct and $d > |W_m|$. Hence N behaviourally correctly learns L as well.

Let p_n be the n -th prime number, note that $p_n > n$ and n does not divide p_n . Now consider the class consisting of the sets $L_n = \{n, p_n, p_n^2, p_n^3, p_n^4, p_n^5, \dots\}$. The class $\{L_0, L_1, L_2, \dots\}$ is finitely learnable in any acceptable numbering. However $\{L_0, L_1, L_2, \dots\}$ is not vacillatorily learnable in the numbering $A_0, A_1, A_2, \dots$ — otherwise, for any n , one can produce a canonical text for L_n and would then have that the largest hypothesis output by the learner on this text is an upper bound for $|W_n|$, whenever W_n is finite; this contradicts the fact that finiteness of r.e. sets cannot be decided in the limit. $\square$

Theorem 13. *There are numberings $A_0, A_1, A_2, \dots$ and $B_0, B_1, B_2, \dots$ with the following properties.*

- (a) *Both numberings are optimal for explanatory learning.*
- (b) *Both numberings are neither effectively optimal for explanatory nor effectively optimal for vacillatory learning.*
- (c) *Both numberings are not optimal for behaviourally correct learning.*
- (d) *The numbering $A_0, A_1, A_2, \dots$ is not optimal for vacillatory learning.*
- (e) *The numbering $B_0, B_1, B_2, \dots$ is optimal for vacillatory learning.*

Theorem 14. *There is a numbering which is optimal but not effectively optimal for behaviourally correct learning.*

Proof. The idea is to construct a uniformly K -r.e. listing $L_0, L_1, L_2, \dots$ of cofinite sets such that, for every m ,

- $\min(\mathbb{N} - L_m)$ exists and is m ;
- the machines $M_0, M_1, M_2, \dots, M_m$ do not behaviourally correctly learn L_m .

Each set L_m is obtained using movable markers $a_0, a_1, a_2, \dots, a_m$: One constructs a text $T_m \leq_T K$ which enumerates all numbers except m and the final values of those markers which move only finitely often. Each marker a_k is initialized as $m+k+1$. $T_m[s]$ contains only values below $s+m+2$. In the case that the current value of a_k is not in $W_{M_k}(T_m[s])$, move a_k to the value $(s+1)(m+1)+k+1$. Furthermore, $T_m(s)$ is the least number x neither in $\{m\} \cup \text{content}(T_m[s])$ nor a current value of any marker. In the case that the value of a_k changes infinitely often, M_k does not converge on T_m semantically to L_m , as M_k infinitely often

conjectures a set not containing some intermediate value of a_k , even though this intermediate value belongs to L_m . In the case that the value of a_k changes only finitely often, the final value of a_k does not belong to L_m , but belongs to almost all of the conjectures output by M_k on T_m .

The reader should note that there are uniformly recursive approximations $L_{m,s}$ satisfying for all m that

- $\forall x \leq m \forall s [x \in L_{m,s} \Leftrightarrow x < m]$;
- $\forall x > m [x \in L_m \Leftrightarrow \forall^\infty s [x \in L_{m,s}]]$.

Using a construction similar to Proposition 8, one can construct a numbering $H_0, H_1, H_2, \dots$ with the following property: For every k , the cofinite set $\mathbb{N} - D_k$ has exactly one index $g(k)$ and this $g(k)$ satisfies $C^K(g(k)) > 2^k$. Thus no infinite class of cofinite sets can be behaviourally correctly learnt using $H_0, H_1, H_2, \dots$ as a hypothesis space.

Now define, for all e and $d > 0$, that $A_{\langle 0,e \rangle} = H_e$ and $A_{\langle d,e \rangle}$ is the union of all $W_{e,s}$ for which there are m, x such that

- $m < x \leq d \leq s$ and
- $m = \min(\mathbb{N} - W_{e,s})$ and
- either $x \in \bigcap_{t=d, d+1, d+2, \dots, s} L_{m,t} - W_{e,s}$ or $x \in W_{e,s} - L_{m,s}$.

Note that $A_{\langle d,e \rangle}$ is finite if $\{0, 1, 2, \dots, d\} \subseteq W_e$ or there exists a number $m < d$ with $L_m \cap \{0, 1, 2, \dots, d\} = W_e \cap \{0, 1, 2, \dots, d\}$. Furthermore, $H_0, H_1, H_2, \dots$ covers all cofinite sets and hence $A_0, A_1, A_2, \dots$ also covers all cofinite sets. The coverage of the coinfinite sets is now based on the following claim.

Claim 15. *Let B be a given r.e. set such that $B \notin \{\emptyset, \mathbb{N}, L_0, L_1, L_2, \dots\}$. Then there is a constant c such that, for all e with $W_e = B$ and all $d > c$, it holds that $A_{\langle d,e \rangle} = B$.*

To see this claim, let $m = \min(\mathbb{N} - B)$ and $x = \min((L_m - B) \cup (B - L_m))$. Note that $x > m$. If $x \notin L_m$, then let $c = x + 1$, else choose c so large that $\forall s \geq c [c > x \wedge x \in L_{m,s}]$. Let e be such that $W_e = B$. Assume that $d > c$. Note that $x \leq d$. There are two cases.

First $x \in L_m \wedge x \notin B$. Then it holds, for all $s \geq d$, that $x \in \bigcap_{t:d \leq t \leq s} L_{m,t} - W_{e,s}$ and hence $A_{\langle d,e \rangle} = \bigcup_{s:s \geq d} W_{e,s} = W_e$.

Second $x \notin L_m \wedge x \in B$. Then there are infinitely many s with $x \in W_{e,s} - L_{m,s}$ and $A_{\langle d,e \rangle}$ is the union of the sets $W_{e,s}$ for these s ; hence $A_{\langle d,e \rangle} = W_e = B$. This completes the proof of the claim.

Let S be a behaviourally correctly learnable class with learner M and let $I = \{i : H_i \in S \cap \{\mathbb{N}, L_0, L_1, L_2, \dots\}\}$. By choice of $L_0, L_1, L_2, \dots$ and $H_0, H_1, H_2, \dots$, I is finite. For each $i \in I$, let F_i be the tell-tale set for H_i with respect to S . That is, F_i is a finite subset of H_i such that, for all $B \in S - \{H_i\}$, $\neg[F_i \subseteq B \subseteq H_i]$. One now defines a new learner N as follows:

$$N(\sigma) = \begin{cases} \langle 0, i \rangle, & \text{if } i \in I \text{ and } F_i \subseteq \text{content}(\sigma) \subseteq H_i; \\ \langle |\sigma|, M(\sigma) \rangle, & \text{if such an } i \in I \text{ does not exist.} \end{cases}$$

If there are several $i \in I$ qualifying, one just takes the least of these i . The new learner N clearly learns $\{H_i : i \in I\}$. Now consider any text T for a set $B \in S - \{\mathbb{N}, L_0, L_1, L_2, \dots\}$. Then, for all sufficiently large s , $W_{M(T[s])} = B$, $s > c$ for the constant c from the claim and there is no $i \in I$ with $F_i \subseteq \text{content}(T[s]) \subseteq H_i$. It follows that $N(T[s]) = \langle s, M(T[s]) \rangle$ and $A_{N(T[s])} = A_{\langle s, M(T[s]) \rangle} = B$. Hence N behaviourally correctly learns B using $A_0, A_1, A_2, \dots$ and $A_0, A_1, A_2, \dots$ is optimal for behaviourally correct learning.

Now assume by way of contradiction that $A_0, A_1, A_2, \dots$ is effectively optimal for behaviourally correct learning. Thus, one can effectively find a learner N_d for $\{\mathbb{N} - D_d\}$ (using the numbering $A_0, A_1, A_2, \dots$). Let T_d be a text for $\mathbb{N} - D_d$, obtained effectively from d . Let h be a partial K -recursive function such that $h(d) = e$, if N_d on T_d converges to e ; otherwise, $h(d)$ is undefined. Note that $h(d) = g(d)$ for all d such that $\mathbb{N} - D_d = L_n$ for some n . Furthermore, $C^K(h(d)) \leq d + c$, for some constant c , whenever $h(d)$ is defined. However, recall that $C^K(g(d)) \geq 2^d$ for all d . This leads to contradiction, as there exist infinitely many distinct d such that $\mathbb{N} - D_d = L_n$ for some n . It follows that $A_0, A_1, A_2, \dots$ is not effectively optimal for behaviourally correct learning. $\square$

4 Consistent Learning

There are various versions of requiring consistency for learning. For example, one can either require that consistency holds only for texts for sets from the class to be learnt or for all texts. Furthermore, one might either require that a learner is partial or that a learner is total. In the following, the version is chosen which Wiehagen and Zeugmann [19] called “totally consistent” and where the learner has to be total and always outputs hypotheses containing all data seen so far (even on data not belonging to any set to be learnt).

Definition 16 (Wiehagen and Liepe [18]). A learner M is consistent iff for every sequence σ it holds that $M(\sigma)$ is defined and $\text{content}(\sigma) \subseteq W_{M(\sigma)}$. A class S is consistently learnable iff there is a consistent learner which explanatorily learns S .

Proposition 17. *If a numbering is effectively optimal for explanatory learning then it is also effectively optimal for consistent learning.*

Proof. Let $A_0, A_1, A_2, \dots$ be a numbering which is effectively optimal for explanatory learning. Then there is, by Theorem 6, a recursive function f such that, for all e , $d = \lim_s f(e, s)$ exists and $A_d = W_e$. Now let S be a consistently learnable class and let M be a consistent learner using $W_0, W_1, W_2, \dots$ for S . The new learner using $A_0, A_1, A_2, \dots$ for S is given as

$$N(\sigma) = f(M(\sigma), s) \text{ for the least } s \text{ with } s > |\sigma| \wedge \text{content}(\sigma) \subseteq A_{f(M(\sigma), s), s}.$$

As M is consistent, $\text{content}(\sigma) \subseteq W_{M(\sigma)}$. Furthermore, $f(M(\sigma), s)$ converges to a fixed value d as s goes to infinity; this d satisfies $\text{content}(\sigma) \subseteq A_{d, s}$ for almost

all s . Hence, if s is sufficiently large, $\text{content}(\sigma) \subseteq A_{f(M(\sigma),s),s}$ as well. It follows that above new learner N is total and consistent.

Furthermore, when M converges on a text T to e , then N converges to a value $d = \lim_s f(e, s)$. The reason is that there are only finitely many s for which $f(e, s)$ differs from d ; thus if the initial segment $\sigma \preceq T$ processed by M is sufficiently large, then $M(\sigma) = e$ and all $s > |\sigma|$ satisfy $f(e, s) = d$ — hence $N(\sigma) = d$. By the definition of f , $A_d = W_e$. So it follows that N using $A_0, A_1, A_2, \dots$ explanatorily learns S . $\square$

Definition 18 (Osherson, Stob and Weinstein [15], Fulk [8]). A learner is called *prudent* if it learns (according to the relevant criterion) every set for which it outputs a hypothesis on some data.

Theorem 19. *If M consistently learns a class S , then there is also a consistent and prudent learner N for S .*

Theorem 20. *If $A_0, A_1, A_2, \dots$ is optimal for explanatory learning, then $A_0, A_1, A_2, \dots$ is also optimal for consistent learning.*

Proof. Let T_e be the canonical text for W_e ; note that the T_e are all uniformly recursive. Assume that $A_0, A_1, A_2, \dots$ is optimal for explanatory learning and let S be a consistently learnable class. By Theorem 19 there is a prudent and consistent learner M using $W_0, W_1, W_2, \dots$ for S . As $A_0, A_1, A_2, \dots$ is optimal for explanatory learning, there is also a further explanatory learner P using $A_0, A_1, A_2, \dots$ for the class consistently learnt by M . The new consistent learner N using $A_0, A_1, A_2, \dots$ is defined as follows:

$$N(\sigma) = P(T_{M(\sigma)}[n]) \text{ for the least } n \text{ with } n > |\sigma| \text{ and } \text{content}(\sigma) \subseteq A_{P(T_{M(\sigma)}[n]),n}.$$

The learner N uses $A_0, A_1, A_2, \dots$ and is partial-recursive. As $M(\sigma)$ is the index of a set containing $\text{content}(\sigma)$, the learner P converges on the text $T_{M(\sigma)}$ to an index c with $\text{content}(\sigma) \subseteq W_{M(\sigma)} = A_c$. Hence the parameter n in the algorithm to compute $N(\sigma)$ is always found; so the learner N is total and consistent. Furthermore, if M converges on a text to e , then P is, from some time onwards, always simulated on T_e . As P converges on T_e to an index d with $A_d = W_e$ and as N always chooses a parameter $n > |\sigma|$, it follows that N converges to this d as well. Hence N explanatorily learns all the sets consistently learnt by M ; in particular, N explanatorily learns the class S . This shows that $A_0, A_1, A_2, \dots$ is optimal for consistent learning. $\square$

The converse is not true. There is a numbering which is effectively optimal for consistent learning but not optimal for explanatory learning.

Theorem 21. *There is a numbering $A_0, A_1, A_2, \dots$ such that:*

- (a) $A_0, A_1, A_2, \dots$ is effectively optimal for consistent learning;
- (b) $A_0, A_1, A_2, \dots$ is not optimal for finite, explanatory, vacillatory or behaviourally correct learning.

Note that the proof of Theorem 9 gives a numbering which is optimal for finite learning but not optimal for consistent learning. The proof of Theorem 10 gives a numbering which is effectively optimal for vacillatory learning but not optimal for consistent learning. The proof of Theorem 12 gives a numbering which is effectively optimal for behaviourally correct learning but not optimal for consistent learning. Separation of non-effective and effective optimality for consistent learning can be obtained using the numbering $A_0, A_1, A_2, \dots$ in the omitted proof of Theorem 13.

5 Learning with Additional Information

Learning with additional information is a scenario in which a learner receives, besides the text of the set to be learnt, also an upper bound on an index (in the numbering used as hypothesis space) for the set to be learnt. We can consider the learner as receiving two items as input: first an upper bound on an index for the input language and second the text for the language to be learnt.

Definition 22. A class S is explanatorily learnable with additional information using $A_0, A_1, A_2, \dots$ iff there is a learner M such that, for every d, e with $d > e \wedge A_e \in S$ and for every text T for A_e , $\lim_{n \rightarrow \infty} M(d, T[n])$ converges to an index c with $A_c = A_e$.

Note that the additional information d must be chosen according to the hypothesis space $A_0, A_1, A_2, \dots$ used and not according to any other numbering.

Jain and Stephan [11] called a numbering $A_0, A_1, A_2, \dots$ of all r.e. sets a Ke -numbering iff $\{\langle i, j \rangle : A_i = A_j\} \leq_T K$. Ke -numberings are generalizations of Friedberg numberings and can never be acceptable or K -acceptable.

Theorem 23. *A numbering is optimal for learning with additional information iff it is effectively optimal for learning with additional information iff it is a Ke -numbering.*

Remark 24. It follows along the lines of previous work [11] that the classes $\{L_0, L_1, L_2, \dots\}$ given by $L_n = \{2m : m \in \mathbb{N}\} \cup \{2n + 1\}$ and $\{H_0, H_1, H_2, \dots\}$ given by $H_n = \{2m : m \leq |W_n|\} \cup \{2n + 1\}$ are both finitely learnable using $W_0, W_1, W_2, \dots$, but for every Ke -numbering $A_0, A_1, A_2, \dots$ at least one of these classes is not vacillatorily learnable using this numbering. Hence Ke -numberings are not optimal for finite, explanatory and vacillatory learning.

An open question by Jain and Stephan [11] asks whether every behaviourally correctly learnable class has a learner which uses a Ke -numbering as hypothesis space. The natural counterpart of this question is to ask for the existence of a Ke -numbering which is optimal for behaviourally correct learning. This question is open. The same question is open for consistent learning as well.

References

- [1] Angluin, D.: Inductive inference of formal languages from positive data. *Information and Control* 45, 117–135 (1980)
- [2] Bärzdīņš, J.: Two theorems on the limiting synthesis of functions. In: *Theory of Algorithms and Programs*, LSU, Riga, Latvia, vol. 1, pp. 82–88 (1974)
- [3] Blum, L., Blum, M.: Toward a mathematical theory of inductive inference. *Information and Control* 28, 125–155 (1975)
- [4] Case, J.: The power of vacillation in language learning. *SIAM Journal on Computing* 28, 1941–1969 (1999)
- [5] de Jongh, D., Kanazawa, M.: Angluin’s theorem for indexed families of r.e. sets and applications. In: *Proceedings of the Ninth Annual Conference on Computational Learning Theory*, pp. 193–204. ACM Press, New York (1996)
- [6] Freivalds, R., Kinber, E., Wiehagen, R.: Inductive inference and computable one-one numberings. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik* 28, 463–479 (1982)
- [7] Friedberg, R.: Three theorems on recursive enumeration. *The Journal of Symbolic Logic* 23(3), 309–316 (1958)
- [8] Fulk, M.: Prudence and other conditions on formal language learning. *Information and Computation* 85, 1–11 (1990)
- [9] Gold, E.M.: Language identification in the limit. *Information and Control* 10, 447–474 (1967)
- [10] Jain, S., Sharma, A.: Learning with the knowledge of an upper bound on program size. *Information and Computation* 102, 118–166 (1993)
- [11] Jain, S., Stephan, F.: Learning in Friedberg numberings. In: Hutter, M., Servedio, R.A., Takimoto, E. (eds.) *ALT 2007. LNCS (LNAI)*, vol. 4754, pp. 79–93. Springer, Heidelberg (2007)
- [12] Lange, S.: *Algorithmic Learning of Recursive Languages*. Habilitationsschrift, Universität Leipzig, Mensch und Buch Verlag, Berlin (2000)
- [13] Lange, S., Zeugmann, T.: Language learning in dependence on the space of hypotheses. In: *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, Santa Cruz, California, United States, pp. 127–136 (1993)
- [14] Li, M., Vitányi, P.: *An Introduction to Kolmogorov Complexity and Its Applications*, 2nd edn. Springer, Heidelberg (1997)
- [15] Osherson, D., Stob, M., Weinstein, S.: *Systems That Learn, An Introduction to Learning Theory for Cognitive and Computer Scientists*. MIT Press, Cambridge (1986)
- [16] Osherson, D.N., Weinstein, S.: Criteria of language learning. *Information and Control* 52, 123–138 (1982)
- [17] Soare, R.I.: *Recursively enumerable sets and degrees. Perspectives in Mathematical Logic*. Springer, Berlin (1987)
- [18] Wiehagen, R., Liepe, W.: Charakteristische Eigenschaften von erkennbaren Klassen rekursiver Funktionen. *Journal of Information Processing and Cybernetics (EIK)* 12, 421–438 (1976)
- [19] Wiehagen, R., Zeugmann, T.: Learning and consistency. In: Lange, S., Jantke, K.P. (eds.) *GOSLER 1994. LNCS*, vol. 961, pp. 1–24. Springer, Heidelberg (1995)
- [20] Zeugmann, T.: *Algorithmisches Lernen von Funktionen und Sprachen*. Habilitationsschrift, Technische Hochschule Darmstadt (1993)
- [21] Zilles, S.: Increasing the power of uniform inductive learners. *Journal of Computer and System Sciences* 70, 510–538 (2005)

Learning with Temporary Memory

Steffen Lange¹, Samuel E. Moelius III², and Sandra Zilles³

¹ Fachbereich Informatik, Hochschule Darmstadt
s.lange@fbi.h-da.de

² Department of Computer & Information Sciences, University of Delaware
moelius@cis.udel.edu

³ Department of Computing Science, University of Alberta
zilles@cs.ualberta.ca

Abstract. In the inductive inference framework of learning in the limit, a variation of the bounded example memory (*Bem*) language learning model is considered. Intuitively, the new model constrains the learner's memory not only in *how much* data may be retained, but also in *how long* that data may be retained. More specifically, the model requires that, if a learner commits an example x to memory in some stage of the learning process, then there is some subsequent stage for which x *no longer* appears in the learner's memory. This model is called *temporary example memory* (*Tem*) learning. In some sense, it captures the idea that *memories fade*.

Many interesting results concerning the *Tem*-learning model are presented. For example, there exists a class of languages that can be identified by memorizing $k + 1$ examples in the *Tem* sense, but that cannot be identified by memorizing k examples in the *Bem* sense. On the other hand, there exists a class of languages that can be identified by memorizing *just 1 example* in the *Bem* sense, but that *cannot* be identified by memorizing *any number of examples* in the *Tem* sense. (The proof of this latter result involves an infinitary self-reference argument.) Results are also presented concerning the special cases of: learning *indexable* classes of languages, and learning (arbitrary) classes of *infinite* languages.

1 Introduction

The following is a common scenario in machine learning. A learner is repeatedly fed elements from an incoming stream of data. From this data, the learner must eventually generate a hypothesis that correctly identifies the contents of this stream of data. This is the case, for example, in many applications of neural networks (see [Mit97]).

In many cases, it would be impractical for a learning algorithm to *reconsider* all previously seen data when forming a new hypothesis. Thus, such learners are often designed to work in an *incremental* fashion, considering only the most recently presented datum, and possibly a few previously seen data that the learner considers to be significant.

This scenario has been studied formally by Lange and Zeugmann [LZ96] in the context of Gold-style language learning [Gol67]. Their model is called *bounded*

example memory (*Bem*) learning. Intuitively, as the learner is fed elements from the incoming stream of data, the learner is allowed to commit up to k of these elements to memory, where k is *a priori* fixed. The learner may change which such elements are stored in its memory at any given time. However, any newly committed element *must* come from the incoming stream of data, *and*, the number of such elements can never exceed k . Among the results presented in [LZ96] is: for each k , there is a class of languages that can be identified by memorizing $k + 1$ examples, but that *cannot* be identified by memorizing only k examples (Theorem 1 below). Further results on the *Bem*-learning model are obtained in [CJLZ99, CCJS07].

The *Bem*-learning model allows that any given example may be stored in the learner's memory *indefinitely*. However, most forms of computer memory are *volatile*, in that they require *energy* in order to retain their contents [RCN03]. Moreover, it has been observed in various areas of machine learning that the length of time for which data may be stored in a learner's memory can have an effect upon the capabilities of that learner (e.g., in reinforcement learning [LM92, McC96, Bak02] and in neural networks [HS97]).

Motivated by these observations, we consider a variation of the *Bem*-learning model in which the learner's memory is constrained not only in *how much* data may be stored, but also in *how long* that data may be stored. More specifically, we consider a model which requires that, if a learner commits an example x to memory in some stage of the learning process, then there is some subsequent stage for which x *no longer* appears in the learner's memory. We call this new model *temporary example memory* (*Tem*) learning. In some sense, this model captures the idea that *memories fade*.

Many interesting results concerning the *Tem*-learning model are presented. For example, there exists a class of languages that can be identified by memorizing $k + 1$ examples in the *Tem* sense, but that *cannot* be identified by memorizing k examples in the *Bem* sense (Theorem 3). Thus, being able to store $k + 1$ examples temporarily, can allow one to learn more than being able to store k example indefinitely. On the other hand, there exists a class of languages that can be identified by memorizing *just 1 example* in the *Bem* sense, but that *cannot* be identified by memorizing *any number of examples* in the *Tem* sense (Theorem 4). Thus, being able to store just 1 example indefinitely, can allow one to learn more than being able to store any number of examples temporarily.

Results are also presented concerning the special cases of: learning *indexable* classes of languages, and learning (arbitrary) classes of *infinite* languages. For the case of indexable classes of languages, there exists such a class that can be identified by memorizing an arbitrary but finite number of examples in the *Bem* sense, but that *cannot* be identified by memorizing an arbitrary but finite number of examples in the *Tem* sense (Theorem 5). In the case of classes of infinite languages, however, a completely different picture emerges. In particular, any such class that can be identified by memorizing an arbitrary but finite number of examples in the *Bem* sense, can also be identified by memorizing an arbitrary but finite number of examples in the *Tem* sense (Theorem 8). Intuitively, this

latter result says that, when learning classes of infinite languages, restriction to temporary memory is, in fact, *not* a proper restriction.

In the context of both learning indexable classes of languages, and learning (arbitrary) classes of infinite languages, some problems remain open. These problems are stated formally in Sections 5 and 6.

Due to space constraints, many proofs are omitted or abbreviated. Complete proofs of all theorems can be found in [LMZ08].

2 Preliminaries

Computability-theoretic concepts not covered below are treated in [Rog67].

$\mathbb{N}$ denotes the set of natural numbers, $\{0, 1, 2, \dots\}$. Lowercase italicized letters (e.g., a, b, c), with or without decorations, range over elements of $\mathbb{N}$, unless stated otherwise. In some cases, we treat $\mathbb{N}$ as the set of all strings over some finite alphabet Σ . In such cases, lowercase typewriter-font letters (e.g., $\mathbf{a}, \mathbf{b}, \mathbf{c}$) are used to denote alphabet symbols. For a symbol $\mathbf{a}$ and $n \in \mathbb{N}$, $\mathbf{a}^n$ denotes the string consisting of n repetitions of $\mathbf{a}$ (e.g., $\mathbf{a}^3 = \mathbf{aaa}$). For all strings x , $|x|$ denotes the length of x , i.e., the number of symbols in x .

A *language* is a subset of $\mathbb{N}$. Uppercase italicized letters (e.g., A, B, C), with or without decorations, range over languages. For all A , $\text{Fin}(A)$ denotes the collection of all finite subsets of A . For all nonempty $A \subseteq \mathbb{N}$, $\min A$ denotes the minimum element of A , where $\min \emptyset \stackrel{\text{def}}{=} \infty$. For all nonempty, finite $A \subseteq \mathbb{N}$, $\max A$ denotes the maximum element of A , where $\max \emptyset \stackrel{\text{def}}{=} -1$. $\mathcal{L}$, with or without decorations, ranges over collections of languages.

Let $\#$ be a reserved symbol. For all languages L , t is a text for $L \stackrel{\text{def}}{=} t = (x_i)_{i \in \mathbb{N}}$, where $\{x_i \mid i \in \mathbb{N}\} \subseteq \mathbb{N} \cup \{\#\}$, and $L = \{x_i \mid i \in \mathbb{N}\} - \{\#\}$. For all L , $\text{Text}(L)$ denotes the set of all texts for L . For all texts $t = (x_i)_{i \in \mathbb{N}}$, $\text{content}(t) \stackrel{\text{def}}{=} \{x_i \mid i \in \mathbb{N}\} - \{\#\}$. For all texts t , and all $n \in \mathbb{N}$, $t[n]$ denotes the initial segment of t of length n .

For all one-argument partial functions ψ , and all $x \in \mathbb{N}$, $\psi(x) \downarrow$ denotes that $\psi(x)$ converges; $\psi(x) \uparrow$ denotes that $\psi(x)$ diverges. We use $\uparrow$ to denote the value of a divergent computation.

σ , with or without decorations, ranges over finite initial segments of texts for arbitrary languages. For all σ , $|\sigma|$ denotes the length of σ (equivalently, the size of the domain of σ). For all $\sigma = (x_i)_{i < n}$, $\text{content}(\sigma) \stackrel{\text{def}}{=} \{x_i \mid i < n\} - \{\#\}$. λ denotes the empty initial segment (equivalently, the everywhere divergent function). For all σ_0 and σ_1 , $\sigma_0 \cdot \sigma_1$ denotes the concatenation of σ_0 and σ_1 .

$\varphi_0, \varphi_1, \dots$ denotes any fixed, acceptable numbering of all one-argument partial computable functions from $\mathbb{N}$ to $\mathbb{N}$. Φ denotes a fixed Blum complexity measure for φ . For each $i, s, x \in \mathbb{N}$,

$$\varphi_i^s(x) \stackrel{\text{def}}{=} \begin{cases} \varphi_i(x), & \text{if } [x < s \wedge \Phi_i(x) \leq s]; \\ \uparrow, & \text{otherwise.} \end{cases} \quad (1)$$

For each $i, s \in \mathbb{N}$, $W_i^s \stackrel{\text{def}}{=} \{x \mid \varphi_i^s(x) \downarrow\}$. For each $i \in \mathbb{N}$, $W_i \stackrel{\text{def}}{=} \bigcup_{s \in \mathbb{N}} W_i^s$. For each $s \in \mathbb{N}$, $W_\uparrow \stackrel{\text{def}}{=} W_\uparrow^s \stackrel{\text{def}}{=} \emptyset$.

An *inductive inference machine (IIM)* is a partial computable function whose inputs are initial segments of texts, and whose outputs are elements of $\mathbb{N}$ [OSW86]. $\mathbf{M}$, with or without decorations, ranges over IIMs.

Definitions 1 through 3 below introduce formally the Gold-style learning criteria of relevance to this paper. Therein, *Lim*, *Sdr*, and *It* are mnemonic for *limiting*, *set-driven*, and *iterative*, respectively. The first of these, *Lim*-learning (Definition 1 below), is the most fundamental. Intuitively, an IIM $\mathbf{M}$ is fed successively longer finite initial segments of a text for a target language L . $\mathbf{M}$ successfully identifies the language (from the given text) iff $\mathbf{M}$ converges to a hypothesis that correctly identifies the language (i.e., to a j such that $W_j = L$).

Definition 1 (Gold [Gol67])

- (a) Let $\mathbf{M}$ be an IIM, and let L be a language. $\mathbf{M}$ *LimTxt-identifies* L iff, for each text $t = (x_i)_{i \in \mathbb{N}} \in \text{Text}(L)$, there exists $n \in \mathbb{N}$ such that $W_{\mathbf{M}(t[n])} = L$ and $\mathbf{M}(t[i]) = \mathbf{M}(t[n])$ for all $i \geq n$.
- (b) Let $\mathbf{M}$ be an IIM, and let $\mathcal{L}$ be a class of languages. $\mathbf{M}$ *LimTxt-identifies* $\mathcal{L}$ iff, for each $L \in \mathcal{L}$, $\mathbf{M}$ *LimTxt-identifies* L .
- (c) $\text{LimTxt} = \{\mathcal{L} \mid (\exists \mathbf{M})[\mathbf{M} \text{ LimTxt-identifies } \mathcal{L}]\}$.

The *Lim*-learning model allows that an IIM consider the entire initial segment of text presented to it when forming a new hypothesis. Thus, the IIM may consider: the *order* in which elements appear within that initial segment, and the *multiplicity* with which they appear. The set-driven (*Sdr*) learning model (Definition 2 below) restricts this. In particular, the *Sdr*-learning model requires that an IIM consider only the *contents* of any initial segment, and *not* the order or multiplicity of the elements therein.

Definition 2 (Wexler and Culicover [WC80])

- (a) Let $\mathbf{M}$ be an IIM, let L be a language, and let $M : \text{Fin}(\mathbb{N}) \rightarrow \mathbb{N}$ be a partial computable function. $\mathbf{M}$ *SdrTxt-identifies* L via M iff (i) and (ii) below.
 - (i) $\mathbf{M}$ *LimTxt-identifies* L .
 - (ii) For each text $t = (x_i)_{i \in \mathbb{N}} \in \text{Text}(L)$, and each $i \in \mathbb{N}$, $M(\text{content}(t[i])) = \mathbf{M}(t[i])$.
- (b) Let $\mathbf{M}$ be an IIM, and let $\mathcal{L}$ be a class of languages. $\mathbf{M}$ *SdrTxt-identifies* $\mathcal{L}$ iff there exists M such that, for each $L \in \mathcal{L}$, $\mathbf{M}$ *SdrTxt-identifies* L via M .
- (c) $\text{SdrTxt} = \{\mathcal{L} \mid (\exists \mathbf{M})[\mathbf{M} \text{ SdrTxt-identifies } \mathcal{L}]\}$.

Both of the preceding learning models allow that an IIM consider an *unbounded* number of elements when forming a new hypothesis. This does not seem practicable, in general, and motivates a desire for *memory limited* models of learning. Iterative (*It*) learning (Definition 3 below) is such a memory limited model. The *It*-model requires that an IIM consider *only* its most recently conjectured hypothesis, and the most recently occurring element of an initial segment of text. Thus, the IIM *cannot*, in general, consider previously conjectured hypotheses, *nor* previously occurring elements of an initial segment of text.

Definition 3 (Wiehagen [Wie76])

- (a) Let $\mathbf{M}$ be an IIM, let L be a language, let $M : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ be a partial computable function, and let $j_0 \in \mathbb{N}$. $\mathbf{M}$ *ItTxt-identifies* L via (M, j_0) iff (i) and (ii) below.
 - (i) $\mathbf{M}$ *LimTxt-identifies* L .
 - (ii) For each text $t = (x_i)_{i \in \mathbb{N}} \in \text{Text}(L)$, (α) through (γ) below.
 - (α) For each $i \in \mathbb{N}$, $\mathbf{M}(t[i]) \downarrow$.
 - (β) $\mathbf{M}(t[0]) = j_0$.
 - (γ) For each $i \in \mathbb{N}$, $\mathbf{M}(t[i+1]) = M(\mathbf{M}(t[i]), t(i))$.
- (b) Let $\mathbf{M}$ be an IIM, and let $\mathcal{L}$ be a class of languages. $\mathbf{M}$ *ItTxt-identifies* $\mathcal{L}$ iff there exists (M, j_0) such that, for each $L \in \mathcal{L}$, $\mathbf{M}$ *ItTxt-identifies* L via (M, j_0) .
- (c) $\text{ItTxt} = \{\mathcal{L} \mid (\exists \mathbf{M})[\mathbf{M} \text{ ItTxt-identifies } \mathcal{L}]\}$.

Note that, in Definition 3(b), the behavior of $\mathbf{M}$ on any text t for a language in $\mathcal{L}$ is completely determined by j_0 and the behavior of M on j_0 and t . Thus, when referring to an iterative (or iterative-like) learner, we will, in some cases, refer only to (M, j_0) and avoid mention of $\mathbf{M}$ altogether. We do so similarly for set-driven learners (Definition 2). For iterative-like learning criteria that we define below (Definitions 4 and 5), we do so in terms of such (M, j_0) directly. In all such cases, it will be evident how to construct an appropriate IIM $\mathbf{M}$ from (M, j_0) .

3 Bounded Example Memory (*Bem*) Learning

The following is a natural relaxation of *It*-learning called *k-bounded example-memory* (Bem_k) learning (Lange and Zeugmann [LZ96]). Recall that the *It*-learning model allows that an IIM consider the most recently occurring element of an initial segment of text, but *not* previously occurring elements. By contrast, the Bem_k -learning model allows that the IIM consider up to k such previously occurring elements, where $k \in \mathbb{N}^+$ is *a priori* fixed.

Definition 4 (Lange and Zeugmann [LZ96]). Let $k \in \mathbb{N}^+$ be fixed.

- (a) Let $M : (\mathbb{N} \times \text{Fin}(\mathbb{N})) \times \mathbb{N} \rightarrow \mathbb{N} \times \text{Fin}(\mathbb{N})$ be a partial computable function, let $j_0 \in \mathbb{N}$, and let L be a language. (M, j_0) *Bem_kTxt-identifies* L iff, for each text $t = (x_i)_{i \in \mathbb{N}} \in \text{Text}(L)$, (i) through (iii) below.
 - (i) For each $i \in \mathbb{N}$, $M_i(t) \downarrow$, where $M_0(t) = \langle j_0, \emptyset \rangle$ and $M_{i+1}(t) = M(M_i(t), x_i) = \langle j_{i+1}, X_{i+1} \rangle$.
 - (ii) There exists $n \in \mathbb{N}$ such that $W_{j_n} = L$ and $j_i = j_n$ for all $i \geq n$.
 - (iii) For each $i \in \mathbb{N}$, $X_{i+1} \subseteq X_i \cup \{x_i\}$ and $|X_{i+1}| \leq k$, where $X_0 = \emptyset$.
- (b) Let (M, j_0) be as in (a), and let $\mathcal{L}$ be a class of languages. (M, j_0) *Bem_kTxt-identifies* $\mathcal{L}$ iff, for each $L \in \mathcal{L}$, (M, j_0) *Bem_kTxt-identifies* L .
- (c) $Bem_k\text{Txt} = \{\mathcal{L} \mid (\exists M, j_0)[(M, j_0) \text{ Bem}_k\text{Txt-identifies } \mathcal{L}]\}$.

For the remainder, let $\pi_1^2(\langle j, X \rangle) = j$ and $\pi_2^2(\langle j, X \rangle) = X$, for each $j \in \mathbb{N}$ and $X \in \text{Fin}(\mathbb{N})$.

Note that Definition 4 allows an IIM to change the contents of its example memory infinitely often, even *after* it has converged to its final hypothesis. Thus, changing the contents of the example memory does *not* constitute a mind-change.

The classes $(Bem_kTxt)_{k \in \mathbb{N}^+}$ defined in Definition 4(d) above form a proper hierarchy, as stated in the following theorem.

Theorem 1 (Lange and Zeugmann [LZ96]). For each $k \in \mathbb{N}^+$, $Bem_kTxt \subset Bem_{k+1}Txt$.

A natural variation of Lange and Zeugmann's model is to eliminate the restriction on the number of examples that can be memorized, i.e., to allow that the IIM store an arbitrary number of examples in its memory. We call the resulting learning model Bem_* -learning.

The formal definition of Bem_* -learning is obtained from Definition 4 by replacing Bem_k by Bem_* and by dropping the condition $|X_{i+1}| \leq k$ in (a)(iii).¹ This definition immediately implies the following.

Proposition 1. For each $k \in \mathbb{N}^+$, $Bem_kTxt \subseteq Bem_*Txt$.

Kinber and Stephan [KS95] studied a flexible notion of memory limited learning that subsumes our definition of Bem_* -learning. As an immediate consequence of their results, one obtains a characterization of Bem_* -learning in terms of set-driven learning (Definition 2 above). Recall that, with set-driven learning, the IIM can consider *neither* the order of the elements in the text, *nor* the multiplicity with which they appeared. However, the full set of previously seen examples is always accessible. The similarity to the definition of Bem_* -learning is obvious; nonetheless, the proof of the characterization is not completely straightforward. The reader is referred to [KS95] for details.

Theorem 2 (Kinber and Stephan [KS95]). $SdrTxt = Bem_*Txt \subset LimTxt$.

4 Temporary Example Memory (*Tem*) Learning

This section introduces the temporary example memory (*Tem*) learning model. This model is a natural *restriction* of Bem -learning. It requires that, if a learner commits an example x to memory in some stage of the learning process, then there is some subsequent stage for which x *no longer* appears in the learner's memory.²

¹ N.B. The Bem_* -learning model does *not* afford the same capabilities to a learner as those provided by the Lim -learning model. Since the examples are stored in the learner's memory as a *set*, the learner *cannot* consider the *order* in which those elements appeared, *nor* the *multiplicity* with which they appeared.

² As noted by one anonymous referee, one might reasonably allow elements occurring *infinitely often* in the text to remain in the learner's memory indefinitely. However, such a weakened restriction leads to a model *equivalent* to the one considered herein. The proof of this fact is omitted.

$$\begin{array}{ccccccc}
Bem_1Txt & \subset & Bem_2Txt & \subset & Bem_3Txt & \subset & \cdots & Bem_*Txt \\
\cup & & \cup & & \cup & & & \cup \\
Tem_1Txt & \subset & Tem_2Txt & \subset & Tem_3Txt & \subset & \cdots & Tem_*Txt \\
\\
Bem_1Txt & \not\subseteq & Tem_*Txt
\end{array}$$

Fig. 1. Summary of the results of Section 4

Figure 1 summarizes the results of this section. The main results are the following. Theorem 3 says that there exists a class of languages that can be identified by memorizing $k + 1$ examples in the *Tem* sense, but that *cannot* be identified by memorizing k examples in the *Bem* sense. On the other hand, Theorem 4 says that there exists a class of languages that can be identified by memorizing *just 1 example* in the *Bem* sense, but that *cannot* be identified by memorizing *any number of examples* in the *Tem* sense.

The following is the formal definition of Tem_k -learning. Note the addition of part (a)(iv), as compared to Definition 4.³

Definition 5. Let $k \in \mathbb{N}^+$ be fixed.

- (a) Let $M : (\mathbb{N} \times \text{Fin}(\mathbb{N})) \times \mathbb{N} \rightarrow \mathbb{N} \times \text{Fin}(\mathbb{N})$ be a partial computable function, let $j_0 \in \mathbb{N}$, and let L be a language. (M, j_0) *Tem_kTxt-identifies* L iff, for each text $t = (x_i)_{i \in \mathbb{N}} \in \text{Text}(L)$, (i) through (iv) below.
 - (i) For each $i \in \mathbb{N}$, $M_i(t) \downarrow$, where $M_0(t) = \langle j_0, \emptyset \rangle$ and $M_{i+1}(t) = M(M_i(t), x_i) = \langle j_{i+1}, X_{i+1} \rangle$.
 - (ii) There exists $n \in \mathbb{N}$ such that $W_{j_n} = L$ and $j_i = j_n$ for all $i \geq n$.
 - (iii) For each $i \in \mathbb{N}$, $X_{i+1} \subseteq X_i \cup \{x_i\}$ and $|X_{i+1}| \leq k$, where $X_0 = \emptyset$.
 - (iv) For each $i \in \mathbb{N}$, there exists $i' \geq i$ such that $x_i \notin X_{i'+1}$.
- (b) Let (M, j_0) be as in (a), and let $\mathcal{L}$ be a class of languages. (M, j_0) *Tem_kTxt-identifies* $\mathcal{L}$ iff, for each $L \in \mathcal{L}$, (M, j_0) *Tem_kTxt-identifies* L .
- (c) $Tem_kTxt = \{\mathcal{L} \mid (\exists M, j_0)[(M, j_0) \text{ Tem}_k\text{Txt-identifies } \mathcal{L}]\}$.

The preceding definition immediately implies the following.

Proposition 2. For each $k \in \mathbb{N}^+$, $Tem_kTxt \subseteq Bem_kTxt$.

The formal definition of Tem_* -learning is obtained from Definition 5 by replacing Tem_k by Tem_* and by dropping the condition $|X_{i+1}| \leq k$ in (a)(iii). Again, a few observations follow immediately.

Proposition 3

- (a) For each $k \in \mathbb{N}^+$, $Tem_kTxt \subseteq Tem_*Txt$.
- (b) $Tem_*Txt \subseteq Bem_*Txt$.

³ For simplicity, Definition 5 allows that *when* an example is removed from memory be determined by the learner, as opposed to, say, by the environment. Technically, this gives the learner more control than absolutely necessary. However, this also makes the negative results obtained even more surprising (see, e.g., Theorem 4).

The following is the first main result of this section. Intuitively, it says that being able to store $k + 1$ examples temporarily, can, in some cases, allow one to learn more than being able to store k examples indefinitely.

Theorem 3. For each $k \in \mathbb{N}^+$, $Tem_{k+1}Txt - Bem_kTxt \neq \emptyset$.

Proof (Sketch). Let $k \in \mathbb{N}^+$. For separating Tem_{k+1} and Bem_k we use a class that was already used in [LZ96] for the separation of Bem_{k+1} and Bem_k . We set $\Sigma = \{\mathbf{a}, \mathbf{b}\}$. For every $j, \ell_0, \dots, \ell_k \in \mathbb{N}$, let

$$L_{(j, \ell_0, \dots, \ell_k)} = \{\mathbf{a}^{j+1}\} \cup \{\mathbf{b}^z \mid z \leq j\} \cup \{\mathbf{b}^{\ell_0}, \dots, \mathbf{b}^{\ell_k}\}. \quad (2)$$

By $\mathcal{L}_k$ we denote the class containing $L = \{\mathbf{b}\}^*$ and all the languages $L_{(j, \ell_0, \dots, \ell_k)}$ for $j, \ell_0, \dots, \ell_k \in \mathbb{N}$. The proof that $\mathcal{L}_k \in Tem_{k+1}Txt$ is omitted, due to space constraints. That $\mathcal{L}_k \notin Bem_kTxt$ is proven in [LZ96]. $\square$ (Theorem 3)

Theorem 3 has the following consequences.

Corollary 1

- (a) For each $k \in \mathbb{N}^+$, $Tem_kTxt \subset Tem_{k+1}Txt$.
- (b) $Tem_*Txt - \bigcup_{k \in \mathbb{N}^+} Bem_kTxt \neq \emptyset$.
- (c) $\bigcup_{k \in \mathbb{N}^+} Bem_kTxt \subset Bem_*Txt$.
- (d) $\bigcup_{k \in \mathbb{N}^+} Tem_kTxt \subset Tem_*Txt$.

In contrast to Theorem 3, restriction to temporary memory can have a significant effect upon a learner's capabilities, as demonstrated by our next main result. Intuitively, this result says that being able to store just 1 example indefinitely, can allow one to learn more than being able to store any number of examples temporarily. The proof involves an infinitary self-reference argument.

Theorem 4. $Bem_1Txt - Tem_*Txt \neq \emptyset$.

Proof (Sketch). Let $\mathcal{L} =$

$$\begin{aligned} & \{L \mid 0 \notin L \wedge (\forall e \in L)[W_e = L]\} \\ & \cup \{L \mid 0 \in L \wedge (\exists u \in L - \{0\})[W_u = \emptyset \\ & \quad \wedge (\forall e \in L - \{0, u\})[W_e = L - \{0, u\}]]\}. \end{aligned} \quad (3)$$

The proof that $\mathcal{L} \in Bem_1Txt$ is omitted, due to space constraints. The proof that $\mathcal{L} \notin Tem_*Txt$ follows. By way of contradiction, let $\mathbf{M}$ be such that $\mathbf{M} Tem_*Txt$ -identifies $\mathcal{L}$. By the Operator Recursion Theorem [Cas74, Cas94], there exist *distinct* φ -programs $(e_j)_{j \in \mathbb{N}}$, *none* of which are 0, and whose behavior is determined by the construction in Figure 2. In conjunction with $(e_j)_{j \in \mathbb{N}}$, a series of finite sequences $(\sigma^s)_{s \in \mathbb{N}}$ is constructed. Note that, in the construction of $(\sigma^s)_{s \in \mathbb{N}}$, σ^{s+1} is defined $\Leftrightarrow$ stage s is exited. So, if there is a *least* s_0 such that stage s_0 is *not* exited, then, for all $s' \geq s_0$, let $\sigma^{s'+1} = \sigma^{s_0}$.

Claim 1. (a) through (d) below.

- (a) $(\forall s \in \mathbb{N})[\sigma^s \subseteq \sigma^{s+1}]$.
- (b) $(\forall s \in \mathbb{N})[[s = 0 \vee \text{stage } s - 1 \text{ is exited}] \Rightarrow (\forall j < 2s)[\text{content}(\sigma^s) \subseteq W_{e_j}]]$.
- (c) $(\forall j \in \mathbb{N})[W_{e_j} \subseteq \bigcup_{s \in \mathbb{N}} \text{content}(\sigma^s)]$.
- (d) $(\forall s \in \mathbb{N})[\text{content}(\sigma^s) \subseteq \{e_j \mid j < 2s\}]$.

Proof of Claim. Clear by the construction of $(e_j)_{j \in \mathbb{N}}$ and $(\sigma^s)_{s \in \mathbb{N}}$. $\square$ (Claim 1)

Set $\sigma^0 = \lambda$, and execute stages $s = 0, 1, \dots$, successively, as follows.

STAGE s . Find the *least* $m \in \mathbb{N}$ (if any) for which one of the following conditions applies, and act accordingly.

COND. (i) $(\exists i \in \{0, 1\})[\mathbf{M}(\sigma^s \cdot e_{2s+i} \cdot \#^m) \uparrow]$. Go into an infinite loop.

COND. (ii) $[\neg(\text{i}) \wedge (\exists i \in \{0, 1\})[(\pi_1^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^m) \neq (\pi_1^2 \circ \mathbf{M})(\sigma^s)]]$.

(a) For the *least* $i \in \{0, 1\}$ satisfying the condition, set $\sigma^{s+1} = \sigma^s \cdot e_{2s+i} \cdot \#^m$.

(b) For each $j < 2s + 2$, list $\text{content}(\sigma^{s+1})$ into W_{e_j} .

(c) Proceed to stage $s + 1$.

COND. (iii) $[\neg(\text{i})\text{--}(\text{ii}) \wedge (\forall i \in \{0, 1\})[(\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^m) = \emptyset]]$.

(a) Set $\sigma^{s+1} = \sigma^s$.

(b) Terminate the construction.

COND. (iv) $[\neg(\text{i})\text{--}(\text{iii}) \wedge m > 0 \wedge (\exists i \in \{0, 1\})[(\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^m) = (\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^{m-1}) \neq \emptyset]]$.

(a) For the *least* $i \in \{0, 1\}$ satisfying the condition, set $\sigma^{s+1} = \sigma^s \cdot e_{2s+i} \cdot \#^m$.

(b) For each $j < 2s + 2$, list $\text{content}(\sigma^{s+1})$ into W_{e_j} .

(c) Terminate the construction.

Fig. 2. The construction of $(e_j)_{j \in \mathbb{N}}$ and $(\sigma^s)_{s \in \mathbb{N}}$ in the proof that $\mathcal{L} \notin \text{Tem}^* \text{Txt}$ (part of Theorem 4). Note: nothing is listed into W_{e_j} , for any j , aside from the above.

Consider the following cases.

CASE (I) $(\exists s \in \mathbb{N})(\forall m \in \mathbb{N})[\text{none of COND. (i)–(iv) apply for } m \text{ in stage } s]$. Then, for all $m \in \mathbb{N}$, (i) through (iv) below.

(i) $(\forall i \in \{0, 1\})[\mathbf{M}(\sigma^s \cdot e_{2s+i} \cdot \#^m) \downarrow]$.

(ii) $(\forall i \in \{0, 1\})[(\pi_1^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^m) = (\pi_1^2 \circ \mathbf{M})(\sigma^s)]$.

(iii) $(\exists i \in \{0, 1\})[(\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^m) \neq \emptyset]$.

(iv) $m > 0 \Rightarrow (\forall i \in \{0, 1\})[\begin{aligned} &(\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^m) \neq \\ &(\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^{m-1}) \\ &\vee (\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^{m-1}) = \emptyset \end{aligned}]$.

By (i) and (ii), clearly, for all $i \in \{0, 1\}$,

$$(\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i}) \supseteq (\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#) \supseteq (\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^2) \supseteq \dots \quad (4)$$

Since, for all σ , $(\pi_2^2 \circ \mathbf{M})(\sigma)$ is a *finite* set, both of the sequences corresponding to (4) must eventually reach a fixpoint. But, clearly, by (iii) and (iv), at least one such sequence does *not* reach a fixpoint (a contradiction).

CASE (II) $(\exists s, m \in \mathbb{N})[\text{COND. (i) applies for } m \text{ in stage } s]$. Then, clearly,

$$(\forall s')[\text{stage } s' \text{ is exited} \Leftrightarrow s' < s]. \quad (5)$$

Thus, for all $j < 2s$,

$$\begin{aligned} \text{content}(\sigma^s) &\subseteq W_{e_j} && \{\text{by (5) and Claim 1(b)}\} \\ &\subseteq \text{content}(\sigma^s) && \{\text{by (a) and (c) of Claim 1, and (5)}\} \\ &\subseteq \{e_j \mid j < 2s\} && \{\text{by Claim 1(d)}\}. \end{aligned} \quad (6)$$

Clearly, by the construction of $(e_j)_{j \in \mathbb{N}}$,

$$(\forall i \in \{0, 1\})[W_{e_{2s+i}} = \emptyset]. \quad (7)$$

Let $i \in \{0, 1\}$ be *least* such that

$$\mathbf{M}(\sigma^s \cdot e_{2s+i} \cdot \#^m) \uparrow. \quad (8)$$

Let t' be such that $t' = \sigma^s \cdot e_{2s+i} \cdot \#^m \cdot 0 \cdot \# \cdot \# \cdot \dots$. Let $L' = \text{content}(t')$. By (6) and (7), clearly, L' is a language in $\mathcal{L}$ of the second type in (3) (where, $u = e_{2s+i}$). But, by (8), $\mathbf{M}$ does *not* $\text{Tem}_* \text{Txt}$ -identify L' from t' (a contradiction).

CASE (III) $(\exists s, m \in \mathbb{N})[\text{COND. (iii) applies for } m \text{ in stage } s]$. Then, clearly,

$$(\forall s')[\text{stage } s' \text{ is exited} \Leftrightarrow s' \leq s]. \quad (9)$$

Thus, for all $j < 2s$,

$$\begin{aligned} \text{content}(\sigma^s) &= W_{e_j} && \{\text{by (9) and Claim 1(b)}\} \\ &\subseteq \text{content}(\sigma^{s+1}) && \{\text{by (a) and (c) of Claim 1, and (9)}\} \\ &= \text{content}(\sigma^s) && \{\text{by the case and the constr. of } (\sigma^s)_{s \in \mathbb{N}}\} \\ &\subseteq \{e_j \mid j < 2s\} && \{\text{by Claim 1(d)}\}. \end{aligned} \quad (10)$$

Clearly, by the construction of $(e_j)_{j \in \mathbb{N}}$,

$$(\forall i \in \{0, 1\})[W_{e_{2s+i}} = \emptyset]. \quad (11)$$

Note that part of COND. (iii) is that COND. (ii) does *not* apply. Thus,

$$(\forall i \in \{0, 1\})[\mathbf{M}(\sigma^s \cdot e_{2s+i} \cdot \#^m) = \langle (\pi_1^2 \circ \mathbf{M})(\sigma^s), \emptyset \rangle]. \quad (12)$$

For all $i \in \{0, 1\}$, let t'_i be such that $t'_i = \sigma^s \cdot e_{2s+i} \cdot \#^m \cdot 0 \cdot \# \cdot \# \cdot \dots$. For all $i \in \{0, 1\}$, let $L'_i = \text{content}(t'_i)$. By (10) and (11), clearly, L'_0 and L'_1 are *distinct* languages in $\mathcal{L}$ of the second type in (3) (where, $u = e_{2s}$ for L'_0 , and $u = e_{2s+1}$ for L'_1). But, by (12), $\mathbf{M}$ cannot distinguish L'_0 and L'_1 (a contradiction).

CASE (IV) $(\exists s, m \in \mathbb{N})[\text{COND. (iv) applies for } m \text{ in stage } s]$. Let $i \in \{0, 1\}$ be *least* such that

$$(\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^m) = (\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^{m-1}) \neq \emptyset. \quad (13)$$

Note that, part of COND. (iv) is that COND. (ii) does *not* apply. Furthermore, by the case, $m > 0$. Thus, it must also be that COND. (ii) does *not* apply for $m - 1$ (in stage s). Consequently,

$$\begin{aligned} &\mathbf{M}(\sigma^s \cdot e_{2s+i} \cdot \#^m) \\ &= \langle (\pi_1^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^m), (\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^m) \rangle && \{\text{immediate}\} \\ &= \langle (\pi_1^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^m), (\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^{m-1}) \rangle && \{\text{by (13)}\} \\ &= \langle (\pi_1^2 \circ \mathbf{M})(\sigma^s), (\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^{m-1}) \rangle && \{\text{by } \neg(\text{ii}) \text{ for } m\} \\ &= \langle (\pi_1^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^{m-1}), (\pi_2^2 \circ \mathbf{M})(\sigma^s \cdot e_{2s+i} \cdot \#^{m-1}) \rangle && \{\text{by } \neg(\text{ii}) \\ &\hspace{15em} \text{for } m - 1\} \\ &= \mathbf{M}(\sigma^s \cdot e_{2s+i} \cdot \#^{m-1}) && \{\text{immediate}\}. \end{aligned}$$

Clearly, then, for all $n \geq m$,

$$\mathbf{M}(\sigma^s \cdot e_{2s+i} \cdot \#^n) = \mathbf{M}(\sigma^s \cdot e_{2s+i} \cdot \#^{m-1}). \quad (14)$$

Next, note that, by the construction of $(\sigma^s)_{s \in \mathbb{N}}$,

$$\sigma^{s+1} = \sigma^s \cdot e_{2s+i} \cdot \#^m. \quad (15)$$

Clearly,

$$(\forall s')[\text{stage } s' \text{ is exited} \Leftrightarrow s' \leq s]. \quad (16)$$

Thus, for all $j < 2s + 2$,

$$\begin{aligned} \text{content}(\sigma^{s+1}) &\subseteq W_{e_j} && \{\text{by (16) and Claim 1(b)}\} \\ &\subseteq \text{content}(\sigma^{s+1}) && \{\text{by (a) and (c) of Claim 1, and (16)}\} \\ &\subseteq \{e_j \mid j < 2s + 2\} && \{\text{by Claim 1(d)}\}. \end{aligned} \quad (17)$$

Let t be such that $t = \sigma^{s+1} \cdot \# \cdot \# \cdot \dots$. Let $L = \text{content}(t)$. By (17), clearly, L is a language in $\mathcal{L}$ of the first type in (3). But, by (13), (14), and (15), $\mathbf{M}$ does *not* $\text{Tem}_* \text{Txt}$ -identify L from t (a contradiction).

CASE (V) $[\neg(\text{I})\text{-(IV)}]$. Then, clearly,

$$(\forall s \in \mathbb{N})(\exists m \in \mathbb{N})[\text{COND. (ii) applies for } m \text{ in stage } s]. \quad (18)$$

Let $t = \lim_{s \rightarrow \infty} \sigma^s$. By Claim 1(a), t is well-defined, and, by (18) and the construction of $(\sigma^s)_{s \in \mathbb{N}}$, t is total. Clearly,

$$(\forall s \in \mathbb{N})[\text{stage } s \text{ is exited}]. \quad (19)$$

Thus, for all $j \in \mathbb{N}$,

$$\begin{aligned} \text{content}(t) &= \bigcup_{s \in \mathbb{N}} \text{content}(\sigma^s) && \{\text{immediate}\} \\ &\subseteq W_{e_j} && \{\text{by (19), and (a) and (b) of Claim 1}\} \\ &\subseteq \bigcup_{s \in \mathbb{N}} \text{content}(\sigma^s) && \{\text{by Claim 1(c)}\} \\ &\subseteq \{e_j \mid j \in \mathbb{N}\} && \{\text{by Claim 1(d)}\}. \end{aligned} \quad (20)$$

By (20), $\text{content}(t)$ is a language in $\mathcal{L}$ of the first type in (3). But, by (18), $\mathbf{M}$ *never* reaches a final conjecture on t (a contradiction). $\square$ (*Theorem 4*)

The preceding result, along with Theorem 1 and Propositions 2 and 3, yields the following corollary.

Corollary 2

- (a) For each $k \in \mathbb{N}^+$, $\text{Tem}_k \text{Txt} \subset \text{Bem}_k \text{Txt}$.
- (b) $\text{Tem}_* \text{Txt} \subset \text{Bem}_* \text{Txt}$.

5 *Tem*-Learning of *Indexable* Classes of Languages

In this section, we consider the special case of *Tem*-learning of indexable classes of languages. A class of languages $\mathcal{L}$ is indexable iff (by definition) there exists a computable function $d : \mathbb{N} \times \mathbb{N} \rightarrow \{0, 1\}$ such that $\mathcal{L} = \{L_i \mid i \in \mathbb{N}\}$ where, for each $i \in \mathbb{N}$, $L_i = \{x \in \mathbb{N} \mid d(i, x) = 1\}$ [LZZ08]. Many interesting *and natural* classes of languages are indexable. For example, the classes of regular and context free languages [HMU01] are each indexable.

The next two results say that two of the important separation results obtained in Section 4 are witnessed by indexable classes of languages.

Corollary 3 (of the proof of Theorem 3). For each $k \in \mathbb{N}^+$, there is an indexable class of languages $\mathcal{L}_k$ such that $\mathcal{L}_k \in Tem_{k+1}Txt - Bem_kTxt$.

Proof of Corollary. One need only observe that each of the $\mathcal{L}_k$ constructed in the proof of Theorem 3 is an indexable class. $\square$ (Corollary 3)

Theorem 5. There is an indexable class of languages $\mathcal{L} \in Bem_*Txt - Tem_*Txt$.

Proof (Sketch). Let $\langle \cdot, \cdot \rangle : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ be any 1-1, onto, computable function. For all $A, B \subseteq \mathbb{N}$, let $A \times B = \{\langle a, b \rangle \mid a \in A \wedge b \in B\}$. Let $\mathcal{L}$ be such that

$$\mathcal{L} = \{\{e\} \times A \mid e \in \mathbb{N} \wedge 0 \in A \wedge A \in \text{Fin}(\mathbb{N})\} \cup \{L_e \mid e \in \mathbb{N}\}, \quad (21)$$

where, for each $e \in \mathbb{N}$, $L_e = \bigcup_{s \in \mathbb{N}} \text{content}(\sigma_e^s)$, and $(\sigma_e^s)_{s \in \mathbb{N}}$ is constructed as follows.

Set $\sigma_e^0 = \langle e, 1 \rangle$, and execute stages $s = 0, 1, \dots$, successively, as follows.

STAGE s . Act according to the following (decidable) conditions, then go to stage $s + 1$.

COND. (i) $(\exists n \leq s)[\Phi_e(\sigma_e^s \cdot \#^n) \leq s \wedge (\pi_2^2 \circ \varphi_e)(\sigma_e^s \cdot \#^n) = \emptyset]$. For the *least* $n \in \mathbb{N}$ satisfying the condition, set $\sigma_e^{s+1} = \sigma_e^s \cdot \#^n \cdot \langle e, s + 2 \rangle$.

COND. (ii) $[\neg(\text{i})]$. Set $\sigma_e^{s+1} = \sigma_e^s$.

Note that, for all $e, s \in \mathbb{N}$, $\langle e, s + 2 \rangle \in L_e \Leftrightarrow$ COND. (i) applies in stage s in the construction of $(\sigma_e^s)_{s \in \mathbb{N}}$. Furthermore, it is clearly the case that $\{\langle e, 1 \rangle\} \subseteq L_e \subseteq \{e\} \times (\mathbb{N} + 1)$. Thus, each L_e is computable. By only slightly more reasoning, it can be seen that $\mathcal{L}$ is an indexable class.

It is easily seen that $\mathcal{L} \in SdrTxt$. Thus, by Theorem 2, $\mathcal{L} \in Bem_*Txt$. The proof that $\mathcal{L} \notin Tem_*Txt$ is omitted, due to space constraints. $\square$ (Theorem 5)

It is currently open whether or not the remaining separation results of Section 4 can be witnessed by indexable classes of languages.

Problem 1. Let $k \in \mathbb{N}^+$, $\mathfrak{A} \in \{Bem_1Txt, \dots, Bem_kTxt\}$, and $\mathfrak{B} \in \{Tem_kTxt, Tem_{k+1}Txt, \dots, Tem_*Txt\}$. Is there an indexable class of languages $\mathcal{L} \in \mathfrak{A} - \mathfrak{B}$?

6 *Tem*-Learning of Classes of *Infinite* Languages

In this section, we consider the special case of *Tem*-learning of classes of infinite languages. Our main result of this section, Theorem 8, says that any class of infinite languages that can be identified by memorizing an arbitrary but finite number of examples in the *Bem* sense, can also be identified by memorizing an arbitrary but finite number of examples in the *Tem* sense.

Our first result of this section says that one of the important separation results obtained in Section 4 is witnessed by a class of infinite languages.

Theorem 6. For each $k \in \mathbb{N}^+$, there exists a class $\mathcal{L}_k$ of infinite languages such that $\mathcal{L}_k \in Tem_{k+1}Txt - Bem_kTxt$.

Proof (Sketch). Let $k \in \mathbb{N}^+$. Fix $\Sigma = \{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$. The witnessing class can be defined by taking the class $\mathcal{L}_k$ used in the proof of Theorem 3 and by adding the infinite set $\{\mathbf{c}\}^*$ to every language in this class. Further details are omitted. $\square$ (Theorem 6)

Before presenting our next main result, it is worth recalling the following.

Theorem 7 (Osherson, Stob, and Weinstein [OSW86]). Let $\mathcal{L}$ be any class of infinite languages. Then, $\mathcal{L} \in \text{LimTxt}$ iff $\mathcal{L} \in \text{SdrTxt}$.

Note that Theorems 2 and 7 have the following corollary.

Corollary 4 (of Theorems 2 and 7). Let $\mathcal{L}$ be any class of infinite languages. Then, $\mathcal{L} \in \text{LimTxt}$ iff $\mathcal{L} \in \text{Bem}_*\text{Txt}$.

Thus, Bem_* -learning is *not* a proper restriction when learning classes of infinite languages. This is in contrast to Theorem 2 which also says that Bem_* -learning is a proper restriction when learning classes of arbitrary languages.

Our next main result says that Tem_* -learning is equivalent to Bem_* -learning when learning classes of infinite languages. Thus, by Corollary 4, Tem_* -learning is similarly *not* a proper restriction when learning classes of infinite languages.

Theorem 8. Let $\mathcal{L}$ be any class of infinite languages. Then, $\mathcal{L} \in \text{Bem}_*\text{Txt}$ iff $\mathcal{L} \in \text{Tem}_*\text{Txt}$.

Proof (Sketch). By Proposition 3, it suffices to show that, for each class of infinite languages $\mathcal{L}$, if $\mathcal{L} \in \text{Bem}_*\text{Txt}$, then $\mathcal{L} \in \text{Tem}_*\text{Txt}$. So, let $\mathcal{L}$ be a class of infinite languages, and suppose that $\mathcal{L} \in \text{Bem}_*\text{Txt}$. An M' is constructed such that $M' \text{Tem}_*\text{Txt}$ -identifies $\mathcal{L}$. Due to space constraints, we give only the construction of M' , and not the proof of its correctness.

By Theorem 2, there exists M such that $M \text{SdrTxt}$ -identifies $\mathcal{L}$. Without loss of generality, suppose that $M(\emptyset) \downarrow$. Let p_M be such that, for all finite $A \subseteq \mathbb{N}$, $\varphi_{p_M}(A) = M(A)$. By 1-1 s-m-n [Rog67], there exists a 1-1 computable function f such that, for all finite $A, B \subseteq \mathbb{N}$, and all $k \in \{0, 1\}$, $W_{f(A, B, k)} = W_{M(A)}$.

For all $L \in \mathcal{L}$, all $t = (x_i)_{i \in \mathbb{N}} \in \text{Text}(L)$, and all $i \in \mathbb{N}$, let M' be as follows. $M'_0(t) = \langle f(\emptyset, \emptyset, 0), \emptyset \rangle$ and $M'_{i+1}(t) = M'(\langle f(A_i, B_i, k_i), X_i \rangle, x_i) = \langle f(A_{i+1}, B_{i+1}, k_{i+1}), X_{i+1} \rangle$, where A_{i+1} , B_{i+1} , k_{i+1} , and X_{i+1} are determined as in Figure 3.

The remainder of the proof is omitted. $\square$ (Theorem 8)

Corollary 5. Let $\mathcal{L}$ be any class of infinite languages. Then, $\mathcal{L} \in \text{LimTxt}$ iff $\mathcal{L} \in \text{Tem}_*\text{Txt}$.

It is currently open whether or not the remaining separation results of Section 4 can be witnessed by classes of infinite languages.

Problem 2. Let $k \in \mathbb{N}^+$, $\mathfrak{A} \in \{\text{Bem}_1\text{Txt}, \dots, \text{Bem}_k\text{Txt}\}$, and $\mathfrak{B} \in \{\text{Tem}_k\text{Txt}, \text{Tem}_{k+1}\text{Txt}, \dots, \text{Tem}_*\text{Txt}\}$. Is there a class of infinite languages $\mathcal{L} \in \mathfrak{A} - \mathfrak{B}$?

$A_0 = B_0 = X_0 = \emptyset$ and $k_0 = 0$. For each $i \in \mathbb{N}$, $A_{i+1} = A_i$, $B_{i+1} = B_i$, $k_{i+1} = k_i$, and $X_{i+1} = X_i$, unless stated otherwise.

if $x_i \neq \#$ **then**

let $B_i^* = \begin{cases} B_i, & \text{if } k_i = 0; \\ W_{M(A_i)}^{s_i^{\min}}, & \text{if } k_i = 1, \text{ where } s_i^{\min} = \min\{s \mid (W_{M(A_i)}^{s+1} \cap X_i) \neq \emptyset\}; \end{cases}$

 /* For the latter case, it can be shown that $s_i^{\min} < \infty$. */

let $X_i^+ = (X_i \cup \{x_i\})$;

let $C_i = (B_i^* \cup X_i^+)$;

let $S_i = \{s \leq \max(C_i) \mid B_i^* \subseteq W_{M(A_i)}^s \subseteq C_i \wedge (W_{M(A_i)}^{s+1} \cap (X_i^+ - W_{M(A_i)}^s)) \neq \emptyset\}$;

if $(\exists A') [B_i \subseteq A' \subseteq C_i \wedge \varphi_{PM}^{\max(C_i)}(A') \downarrow \neq \varphi_{PM}^{\max(C_i)}(A_i) \downarrow]$ **then**

$A_{i+1} \leftarrow A'$; $B_{i+1} \leftarrow C_i$; $k_{i+1} \leftarrow 0$; $X_{i+1} \leftarrow \emptyset$;

else if $S_i \neq \emptyset$ **then**

$k_{i+1} \leftarrow 1$; $X_{i+1} \leftarrow (X_i^+ - W_{M(A_i)}^{s_i^{\max}})$, where $s_i^{\max} = \max(S_i)$;

else

$X_{i+1} \leftarrow X_i^+$;

end if;

end if.

Fig. 3. The behavior of M' in the proof of Theorem 8

7 Conclusion

We introduced a new model of language learning called *temporary example memory* (*Tem*) learning. This model is a natural *restriction* of bounded example memory (*Bem*) learning. In particular, it requires that, if a learner commits an example x to memory in some stage of the learning process, then there is some subsequent stage for which x *no longer* appears in the learner's memory. In some sense, this model captures the idea that *memories fade*.

Aside from the open questions mentioned in Sections 5 and 6, the following would constitute an interesting line of research. In some sense, an IIM can memorize examples that it has seen by *coding* them into its hypotheses, i.e., by exploiting redundancy in the hypothesis space. This “memory” is, in principle, unbounded in the number of examples that it can retain, and in how long it can retain them.⁴ From a practical point of view, the option to memorize examples in this way probably does not meet the *intuitive* requirements of a model of incremental learning. Thus, it would be interesting to consider the *Bem* and *Tem*-learning models in conjunction with hypothesis spaces *that have no redundancy*, i.e., Friedberg numberings. Note that such numberings have already been considered as hypothesis spaces in the context of *It*-learning [JS07].

⁴ Of course, since the IIM must eventually converge to a single hypothesis, the IIM can memorize examples in this way only finitely often.

References

- [Bak02] Bakker, B.: Reinforcement learning with long short-term memory. *Advances in Neural Inform. Processing Systems* 14, 1475–1482 (2002)
- [Cas74] Case, J.: Periodicity in generations of automata. *Mathematical Systems Theory* 8(1), 15–32 (1974)
- [Cas94] Case, J.: Infinitary self-reference in learning theory. *Journal of Experimental and Theoretical Artificial Intelligence* 6(1), 3–16 (1994)
- [CCJS07] Carlucci, L., Case, J., Jain, S., Stephan, F.: Results on memory-limited U-shaped learning. *Inform. Comput.* 205(10), 1551–1573 (2007)
- [CJLZ99] Case, J., Jain, S., Lange, S., Zeugmann, T.: Incremental concept learning for bounded data mining. *Inform. Comput.* 152(1), 74–110 (1999)
- [Gol67] Gold, E.M.: Language identification in the limit. *Inform. Control* 10(5), 447–474 (1967)
- [HMU01] Hopcroft, J.E., Motwani, R., Ullman, J.D.: *Introduction to automata theory, languages, and computation*, 2nd edn. Addison Wesley, Reading (2001)
- [HS97] Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Computation* 9(8), 1735–1780 (1997)
- [JS07] Jain, S., Stephan, F.: Learning in Friedberg numberings. In: Hutter, M., Servedio, R.A., Takimoto, E. (eds.) *ALT 2007. LNCS (LNAI)*, vol. 4754, pp. 79–93. Springer, Heidelberg (2007)
- [KS95] Kinber, E., Stephan, F.: Language learning from texts: mind changes, limited memory, and monotonicity. *Inform. Comput.* 123(2), 224–241 (1995)
- [LM92] Lin, L.J., Mitchell, T.: Reinforcement learning with hidden states. In: *Proc. of 2nd Intl. Conf. on Simulation of Adaptive Behavior*, pp. 271–280 (1992)
- [LMZ08] Lange, S., Moelius, S.E., Zilles, S.: Learning with temporary memory (expanded version). Technical report, University of Delaware (2008), <http://www.cis.udel.edu/~moelius/publications>
- [LZ96] Lange, S., Zeugmann, T.: Incremental learning from positive data. *Journal of Computer and System Sciences* 53(1), 88–103 (1996)
- [LZZ08] Lange, S., Zeugmann, T., Zilles, S.: Learning indexed families of recursive languages from positive data: A survey. *Theor. Comput. Sci.* 397(1-3), 194–232 (2008)
- [McC96] McCallum, R.A.: Learning to use selective attention and short-term memory in sequential tasks. In: *Proc. of 4th Intl. Conf. on Simulation of Adaptive Behavior*, pp. 315–324 (1996)
- [Mit97] Mitchell, T.M.: *Machine Learning*. McGraw-Hill Higher Education, New York (1997)
- [OSW86] Osherson, D., Stob, M., Weinstein, S.: *Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists*, 1st edn. MIT Press, Cambridge (1986)
- [RCN03] Rabaey, J.M., Chandrakasan, A., Nikolic, B.: *Digital Integrated Circuits: A Design Perspective*, 2nd edn. Prentice-Hall, Inc., Englewood Cliffs (2003)
- [Rog67] Rogers, H.: *Theory of Recursive Functions and Effective Computability*. McGraw Hill, New York (1967) (Reprinted, MIT Press, 1987)
- [WC80] Wexler, K., Culicover, P.W.: *Formal Principles of Language Acquisition*. MIT Press, Cambridge (1980)
- [Wie76] Wiehagen, R.: Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. *Elektron. Inform. Kybernetik* 12(1/2), 93–99 (1976)

Erratum: Constructing Multiclass Learners from Binary Learners: A Simple Black-Box Analysis of the Generalization Errors

Jittat Fakcharoenphol¹ and Boonserm Kijsirikul²

¹ Department of Computer Engineering, Kasetsart University,
Bangkok, Thailand
jtf@ku.ac.th

² Department of Computer Engineering, Chulalongkorn University,
Bangkok, Thailand
Boonserm.K@chula.ac.th

Abstract. There are errors in our paper “Constructing Multiclass Learners from Binary Learners: A Simple Black-Box Analysis of the Generalization Errors,” which appeared in ALT’05 [3]. The errors are related to our uses of union bounds. We briefly describe the problem and discuss which of our results can be shown to hold. We also provide a counter example for our previous claim in the full version of the erratum.

1 Background

In [3], we analyzed various multiclass learning algorithms that use binary classifications as subroutines. We viewed binary classifiers as black-boxes and analyzed the error rate of the multiclass construction as a function of binary error rates. This approach is mainly known as learning reductions [1].

In what follows, we assume that the readers are familiar with the pair-wise reductions such as Decision Directed Acyclic Graphs [5] (DDAG), Adaptive Directed Acyclic Graphs [4] (ADAG).

2 The Errors

The problems in that paper is in our analysis of adaptive constructions, i.e., those whose set of invoked binary classifiers changes over the input. They include DDAG, ADAG, and Randomized Decision Directed Acyclic Graphs (R-DDAG). More specifically, the errors are regarding our use of the union bound.

To see the problem, we consider our analysis of ADAG for the problem with k classes. We start with the setting. Let D be the distribution over X , set of all data points. There are $\binom{k}{2}$ classifiers: there is a binary classifier $A_{i,j}$ for each pair $i \neq j$. Error rate of classifier $A_{i,j}$, $\epsilon_{i,j}$ is defined to be

$$\Pr_{x \sim D} [A_{i,j}(x) \text{ gives a wrong prediction} | x \text{ belongs to class } i \text{ or class } j].$$

In the ADAG reduction, we have a full binary tree T , with k leaves initially labelled with all classes, while all internal nodes are unlabelled. Call the leaf labelled with i , L_i . Given a data point x , the prediction algorithm picks any unlabelled node u whose children are all labelled. Suppose that they are labelled with i and j . We then call $A_{i,j}(x)$ and assign the result of the classifier as the label of u . The algorithm iterates until there is no unlabelled node left. The multiclass prediction is the label of the root node.

Our analysis first assume that the data point belongs to class i . Let $\mathcal{I}(i)$ denote the set of internal nodes of T on the path from L_i to the root. The algorithm makes a wrong prediction if any classifier called on these nodes make mistakes. Denote by $\mathcal{E}(u)$ the event that the classifier on node u makes a wrong prediction; thus, the multiclass error rate is $\Pr\left[\bigcup_{u \in \mathcal{I}(i)} \mathcal{E}(u)\right] \leq \sum_{u \in \mathcal{I}(i)} \Pr[\mathcal{E}(u)]$.

For any node u , let $\mathcal{L}(i)$ denote the set of leaf labels in the subtree rooted at u . We claim, erroneously, that $\max_{j \in \mathcal{L}(u)} \epsilon_{i,j}$ is an upperbound on $\Pr[\mathcal{E}(u)]$.

If this were true, we would have that, since $|\mathcal{I}(i)| \leq \lceil \log k \rceil$, the error rate is at most $\sum_{j=1}^{\lceil \log k \rceil} \epsilon_{i,r_j}$, when r_j is the class c with the j -th largest error rate $\epsilon_{i,c}$.

The above claim would have work if each classifiers $A_{i,j}$ is randomized and for any data point x , it makes a mistake with probability $\epsilon_{i,j}$. However, usually for a fixed x , the error is not random.

The correct analysis of $\Pr[\mathcal{E}(u)]$ must consider all binary classifiers $A_{i,j}$ for $j \in \mathcal{L}(u)$. Let event $\mathcal{F}(i,j)$ denote the event that the classifier $A_{i,j}$ is used at node u and makes a wrong prediction. Thus, $\Pr[\mathcal{E}(u)] = \Pr[\bigcup_{j \in \mathcal{L}(u)} \mathcal{F}(i,j)] = \sum_{j \in \mathcal{L}(u)} \Pr[\mathcal{F}(i,j)]$. With no further assumption, we can only bound this with $\sum_{j \in \mathcal{L}(u)} \epsilon_{i,j}$, using the union bound. Thus, the probability of making mistake is at most $\sum_{j \neq i} \epsilon_{i,j}$, using again the union bound. In the case of uniform error rate, this only gives the bound of $(k-1)\epsilon$. This analysis is tight (see an example in [2]).

The erroneous theorems are Theorems 2, 3, and 4. In Theorem 2, we claim an upper bound δ_i for input from class i to be at most $\max\{\sum_{j < i} \epsilon_{i,j}, \sum_{j > i} \epsilon_{i,j}\}$. The correct upperbound is $\sum_{j \neq i} \epsilon_{i,j}$. This is the correct bounds for Theorems 3 and 4 as well.

Our analyses of non-adaptive constructions (Theorems 1, 5, and 6) remain correct.

3 A Tight Example

In the full version of the erratum [2], we describe the probability space of the input with k classes and a set of binary classifiers such that the binary error rate is $1/(k-1)$ while multiclass error rate of the constructions is 1, for $k > 2$.

Acknowledgment. We would like to thank John Langford for a useful discussion.

References

- [1] Beygelzimer, A., Dani, V., Hayes, T., Langford, J., Zadrozny, B.: Error limiting reductions between classification tasks. In: ICML 2005, pp. 49–56 (2005)
- [2] Fakcharoenphol, J., Kijsirikul, B.: Erratum: Constructing multiclass learners from binary learners: A simple black-box analysis of the generalization errors, <http://www.cpe.ku.ac.th/~jtf/papers/blackbox-erratum.pdf>
- [3] Fakcharoenphol, J., Kijsirikul, B.: Constructing multiclass learners from binary learners: A simple black-box analysis of the generalization errors. In: Jain, S., Simon, H.U., Tomita, E. (eds.) ALT 2005. LNCS (LNAI), vol. 3734, pp. 135–147. Springer, Heidelberg (2005)
- [4] Kijsirikul, B., Ussivakul, N., Meknavin, S.: Adaptive directed acyclic graphs for multiclass classification. In: Ishizuka, M., Sattar, A. (eds.) PRICAI 2002. LNCS (LNAI), vol. 2417, pp. 158–168. Springer, Heidelberg (2002)
- [5] Platt, J., Cristianini, N., Shawe-Taylor, J.: Large margin DAGs for multiclass classification. In: Advance in Neural Information Processing System, vol. 12. MIT Press, Cambridge (2000)

Author Index

- Agarwal, Shivani 7
Angluin, Dana 272
Antos, András 287
Arias, M. 303
Aspnes, James 272

Balcan, Maria-Florina 316
Balcázar, J.L. 303
Bartók, Gábor 329
Becerra-Bonache, Leonor 359
Blum, Avrim 316

Case, John 359, 389, 404, 419
Chernov, Alexey 138, 199
Cléménçon, Stéphan 22
Cortes, Corinna 38
Csiszár, Imre 1

Dashevskiy, Mikhail 214
de Brecht, Matthew 374

Fakcharoenphol, Jittat 464

Glocer, Karen A. 256
Gretton, Arthur 183
Grover, Varun 287
Györfi, László 108, 183

Hatano, Kohei 227
Herbster, Mark 54
Hoory, Shlomo 344

Jain, Sanjay 359, 434

Kalnishkan, Yuri 199
Keim, Daniel A. 2
Kijssirikul, Boonserm 464
Kötzing, Timo 389, 404

Lange, Steffen 449
Lovász, László 3

Mannila, Heikki 4
Mansmann, Florian 2

Margalit, Oded 344
Maurer, Andreas 70, 79
Mitchell, Tom M. 5
Moelius III, Samuel E. 419, 449
Mohri, Mehryar 38
Moribe, Jun-ichi 227
Mukherjee, Indraneel 240

Oelke, Daniela 2
Ortner, Ronald 123

Pontil, Massimiliano 70, 79

Reyzin, Lev 272
Riley, Michael 38
Rostamizadeh, Afshin 38
Ryabko, Daniil 169

Sabato, Sivan 92
Schapire, Robert E. 240
Shamir, Ohad 92
Shen, Alexander 138, 154
Stephan, Frank 359, 434
Szepesvári, Csaba 287, 329

Takeda, Masayuki 227
Takimoto, Eiji 227
Tishby, Naftali 92

Vajda, István 108
Vayatis, Nicolas 22
Vereshchagin, Nikolai 138
Vishwanathan, S.V.N. 256
Vovk, Vladimir 138, 154, 199

Warmuth, Manfred K. 256

Yamamoto, Akihiro 374

Zhdanov, Fedor 199
Ziegler, Hartmut 2
Zilles, Sandra 329, 449