Marcus Hutter
Frank Stephan
Vladimir Vovk
Thomas Zeugmann (Eds.)

LNAI 6331

Algorithmic Learning Theory

21st International Conference, ALT 2010
Canberra, Australia, October 2010
Proceedings

Springer

Lecture Notes in Artificial Intelligence 6331

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

Subseries of Lecture Notes in Computer Science

Marcus Hutter   Frank Stephan
Vladimir Vovk   Thomas Zeugmann (Eds.)

Algorithmic Learning Theory

21st International Conference, ALT 2010
Canberra, Australia, October 6-8, 2010
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

Marcus Hutter
Australian National University and NICTA
Research School of Information Sciences and Engineering
Canberra, ACT 0200, Australia
E-mail: marcus.hutter@anu.edu.au

Frank Stephan
National University of Singapore, Department of Mathematics
Block S17, 10 Lower Kent Ridge Road, Singapore 119076, Republic of Singapore
E-mail: fstephan@comp.nus.edu.sg

Vladimir Vovk
University of London, Department of Computer Science
Royal Holloway, Egham, Surrey TW20 0EX, UK
E-mail: vovk@cs.rhul.ac.uk

Thomas Zeugmann
Hokkaido University, Division of Computer Science
N-14, W-9, Sapporo 060-0814, Japan
E-mail: thomas@ist.hokudai.ac.jp

Library of Congress Control Number: 2010934948

CR Subject Classification (1998): I.2, F.4.1, F.1, F.2, I.2.3, I.2.6

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN	0302-9743
ISBN-10	3-642-16107-3 Springer Berlin Heidelberg New York
ISBN-13	978-3-642-16107-0 Springer Berlin Heidelberg New York

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

springer.com

© Springer-Verlag Berlin Heidelberg 2010
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper   06/3180

Preface

This volume contains the papers presented at the 21st International Conference on Algorithmic Learning Theory (ALT 2010), which was held in Canberra, Australia, October 6–8, 2010. The conference was co-located with the 13th International Conference on Discovery Science (DS 2010) and with the Machine Learning Summer School, which was held just before ALT 2010. The technical program of ALT 2010, contained 26 papers selected from 44 submissions and five invited talks. The invited talks were presented in joint sessions of both conferences.

ALT 2010 was dedicated to the theoretical foundations of machine learning and took place on the campus of the Australian National University, Canberra, Australia. ALT provides a forum for high-quality talks with a strong theoretical background and scientific interchange in areas such as inductive inference, universal prediction, teaching models, grammatical inference, formal languages, inductive logic programming, query learning, complexity of learning, on-line learning and relative loss bounds, semi-supervised and unsupervised learning, clustering, active learning, statistical learning, support vector machines, Vapnik-Chervonenkis dimension, probably approximately correct learning, Bayesian and causal networks, boosting and bagging, information-based methods, minimum description length, Kolmogorov complexity, kernels, graph learning, decision tree methods, Markov decision processes, reinforcement learning, and real-world applications of algorithmic learning theory.

DS 2010 was the 13th International Conference on Discovery Science and focused on the development and analysis of methods for intelligent data analysis, knowledge discovery and machine learning, as well as their application to scientific knowledge discovery. As is the tradition, it was co-located and held in parallel with Algorithmic Learning Theory.

In addition to these two conferences, the Machine Learning Summer School taught fundamental knowledge and recent results to PhD students and other interested researchers.

The present volume contains the texts of the 26 papers presented at ALT 2010, divided into groups of papers on statistical learning, grammatical inference and graph learning, probably approximately correct learning, query learning and algorithmic teaching, on-line learning, inductive inference, reinforcement learning, and kernel methods. The volume also contains the texts or abstracts of the invited talks:

– Alexander Clark (Royal Holloway, University of London, UK), "Towards General Algorithms for Grammatical Inference" (invited speaker for ALT 2010)
– Manfred K. Warmuth (University of California Santa Cruz, USA), "The Blessing and the Curse of the Multiplicative Updates" (invited speaker for ALT 2010)

– Ivan Bratko (University of Ljubljana, Slovenia), "Discovery of Abstract Concepts by a Robot" (invited speaker for DS 2010)
– Kotagiri Ramamohanarao (University of Melbourne, Australia), "Contrast Pattern Mining and Its Application for Building Robust Classifiers" (invited speaker for DS 2010)
– Peter L. Bartlett (University of California Berkeley, USA), "Optimal Online Prediction in Adversarial Environments" (joint invited speaker for ALT 2010 and DS 2010).

Papers presented at DS 2010 are contained in the DS 2010 proceedings.

Since 1999, ALT has been awarding the E. M. Gold Award for the most outstanding student contribution. This year, the award was given to Gábor Bartók for his paper "Toward a Classification of Finite Partial-Monitoring Games," co-authored by Dávid Pál and Csaba Szepesvári.

ALT 2010 was the 21st in the ALT conference series, established in Japan in 1990. A second root is the conference series Analogical and Inductive Inference previously held in 1986, 1989 and 1992 which merged with the conference series ALT after a co-location in the year 1994. From then on, ALT became an international conference series which kept its strong links to Japan but also was regularly held at overseas destinations including Australia, Germany, Hungary, Italy, Portugal, Singapore, Spain and the USA.

The ALT series is supervised by its Steering Committee: Naoki Abe (IBM Thomas J. Watson Research Center, Yorktown, USA), Shai Ben-David (University of Waterloo, Canada), Philip M. 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), Eiji Takimoto (Kyushu University, Fukuoka, Japan), György Turán (University of Illinois at Chicago, USA and University of Szeged, Hungary), Osamu Watanabe (Tokyo Institute of Technology, Japan), Thomas Zeugmann (Chair, Hokkaido University, Japan), and Sandra Zilles (Publicity Chair, University of Regina, Saskatchewan, Canada).

We would like to thank the many people and institutions who contributed to the success of the conference. In particular, we want to thank our authors for contributing to the conference and for coming to Canberra in October 2010. Without their efforts and willingness to choose ALT 2010 as a forum to report on their research, this conference would not have been possible.

We would like to thank the National ICT Australia for generously sponsoring the conference ALT 2010; NICTA is an Australian research institute dedicated to information and communications technology, and its founding members are the University of New South Wales, the Australian National University, the NSW Government and the ACT Government; later The University of Sydney, the Victorian Government, the University of Melbourne, the Queensland Government, the University of Queensland, the Griffith University and the Queensland University of Technologies became partners. We are furthermore grateful to the

Australian National University (ANU) for hosting the event; ANU is a leading public teaching and research university in Canberra, Australia. The support of ANU and NICTA was a great help, organisationally and financially, for the conferences ALT 2010 and DS 2010. We are also grateful that we could use the excellent conference management system EasyChair for putting together the programme for ALT 2010; EasyChair was developed mainly by Andrei Voronkov and is hosted at the University of Manchester. The system is cost-free.

As already mentioned the conference series ALT has in this years, as in many previous years, been co-located with the series Discovery Science. We are grateful for this continuous collaboration. In particular, we would like to thank the Conference Chair, Achim Hoffmann and the Programme Committee Chairs, Geoffrey Holmes and Bernhard Pfahringer, of DS 2010.

We would like to thank Mark Reid for organising the conference and putting in the tremendous amount of work he has dedicated to making ALT 2010 a success. We want to extend our thanks to the other members of the local organisation committee, who were there to organise the reception, to sit at the information desk and to do the other duties which are connected to organising and hosting a conference.

We are grateful to the members of the Programme Committee for ALT 2010 and the subreferees for their hard work to select a good programme for ALT 2010. Reviewing papers and checking the correctness of results is demanding in time and skills and we very much appreciate this contribution to the conference. Last but not least we thank Springer for their support in preparing and publishing this volume of the Lecture Notes in Artificial Intelligence series.

July 2010

Marcus Hutter
Frank Stephan
Vladimir Vovk
Thomas Zeugmann

Organisation

Conference Chair

Marcus Hutter	Australian National University and National ICT Australia

Program Committee

Marta Arias	Universitat Politècnica de Catalunya
Maria Florina Balcan	Georgia Institute of Technology
Alina Beygelzimer	IBM Research
Alexey Chernov	Royal Holloway, University of London
Corinna Cortes	Google Research New York
Łukasz Dębowski	Institute of Computer Science, Polish Academy of Sciences
Henning Fernau	Universität Trier
Matthias Hein	Saarland University
Kouichi Hirata	Kyushu Institute of Technology
Sanjay Jain	National University of Singapore
Jyrki Kivinen	University of Helsinki
Gábor Lugosi	Pompeu Fabra University
Eric Martin	The University of New South Wales
Rémi Munos	Institut National de Recherche en Informatique et en Automatique Lille
Ronald Ortner	University of Leoben
Mark Reid	Australian National University
Steven de Rooij	University of Cambridge
Frank Stephan (chair)	National University of Singapore
Nicolas Vayatis	Ecole Normale Supérieure de Cachan
Kenji Yamanishi	The University of Tokyo
Vladimir Vovk (chair)	Royal Holloway, University of London
Sandra Zilles	University of Regina

Local Arrangements Chair

Mark Reid	Australian National University

Subreferees

Nir Ailon
Nicolas Baskiotis
Avrim Blum
Sébastien Bubeck
Joseph Defretin
Nicolás Della Penna
Christos Dimitrakakis
Mohammad Ghavamzadeh
Daniel Golovin
Steve Hanneke
Eiju Hirowatari
Yuri Kalnishkan
Varun Kanade
Anna Kasprzik
Alessandro Lazaric
Odalric-Ambrym Maillard
Daniel Meister
Tetsuhiro Miyahara
Novi Quadrianto
Afshin Rostamizadeh
Cynthia Rudin
Hiroshi Sakamoto
Ravi Sastry
Rocco Servedio
Jinwoo Shin
Kilho Shin
Gilles Stoltz
Mahito Sugiyama
Peter Sunehag
Ameet Talwalkar
Franck Thollard
Ryota Tomioka
Fedor Zhdanov

Sponsoring Institutions

National ICT Australia

Australian National University

Table of Contents

Editors' Introduction 1
Marcus Hutter, Frank Stephan, Vladimir Vovk, and Thomas Zeugmann

Invited Papers

Towards General Algorithms for Grammatical Inference 11
Alexander Clark

The Blessing and the Curse of the Multiplicative Updates (Abstract) . . . 31
Manfred K. Warmuth

Discovery of Abstract Concepts by a Robot (Abstract) 32
Ivan Bratko

Contrast Pattern Mining and Its Application for Building Robust Classifiers (Abstract) 33
Kotagiri Ramamohanarao

Optimal Online Prediction in Adversarial Environments 34
Peter L. Bartlett

Regular Contributions

Statistical Learning

An Algorithm for Iterative Selection of Blocks of Features 35
Pierre Alquier

Bayesian Active Learning Using Arbitrary Binary Valued Queries 50
Liu Yang, Steve Hanneke, and Jaime Carbonell

Approximation Stability and Boosting 59
Wei Gao and Zhi-Hua Zhou

Grammatical Inference and Graph Learning

A Spectral Approach for Probabilistic Grammatical Inference on Trees 74
Raphaël Bailly, Amaury Habrard, and François Denis

PageRank Optimization in Polynomial Time by Stochastic Shortest Path Reformulation 89
Balázs Csanád Csáji, Raphaël M. Jungers, and Vincent D. Blondel

Inferring Social Networks from Outbreaks 104
Dana Angluin, James Aspnes, and Lev Reyzin

Probably Approximately Correct Learning

Distribution-Dependent PAC-Bayes Priors 119
Guy Lever, François Laviolette, and John Shawe-Taylor

PAC Learnability of a Concept Class under Non-atomic Measures: A Problem by Vidyasagar 134
Vladimir Pestov

A PAC-Bayes Bound for Tailored Density Estimation 148
Matthew Higgs and John Shawe-Taylor

Compressed Learning with Regular Concept 163
Jiawei Lv, Jianwen Zhang, Fei Wang, Zheng Wang, and Changshui Zhang

Query Learning and Algorithmic Teaching

A Lower Bound for Learning Distributions Generated by Probabilistic Automata 179
Borja Balle, Jorge Castro, and Ricard Gavaldà

Lower Bounds on Learning Random Structures with Statistical Queries 194
Dana Angluin, David Eisenstat, Leonid (Aryeh) Kontorovich, and Lev Reyzin

Recursive Teaching Dimension, Learning Complexity, and Maximum Classes 209
Thorsten Doliwa, Hans Ulrich Simon, and Sandra Zilles

On-line Learning

Toward a Classification of Finite Partial-Monitoring Games 224
Gábor Bartók, Dávid Pál, and Csaba Szepesvári

Switching Investments 239
Wouter M. Koolen and Steven de Rooij

Prediction with Expert Advice under Discounted Loss 255
Alexey Chernov and Fedor Zhdanov

A Regularization Approach to Metrical Task Systems 270
Jacob Abernethy, Peter L. Bartlett, Niv Buchbinder, and Isabelle Stanton

Inductive Inference

Solutions to Open Questions for Non-U-Shaped Learning with Memory Limitations 285
John Case and Timo Kötzing

Learning without Coding 300
Samuel E. Moelius III and Sandra Zilles

Learning Figures with the Hausdorff Metric by Fractals 315
Mahito Sugiyama, Eiju Hirowatari, Hideki Tsuiki, and Akihiro Yamamoto

Inductive Inference of Languages from Samplings 330
Sanjay Jain and Efim Kinber

Reinforcement Learning

Optimality Issues of Universal Greedy Agents with Static Priors 345
Laurent Orseau

Consistency of Feature Markov Processes 360
Peter Sunehag and Marcus Hutter

Algorithms for Adversarial Bandit Problems with Multiple Plays 375
Taishi Uchiya, Atsuyoshi Nakamura, and Mineichi Kudo

On-line Learning and Kernel Methods

Online Multiple Kernel Learning: Algorithms and Mistake Bounds 390
Rong Jin, Steven C.H. Hoi, and Tianbao Yang

An Identity for Kernel Ridge Regression 405
Fedor Zhdanov and Yuri Kalnishkan

Author Index 421

Editors' Introduction

Marcus Hutter, Frank Stephan, Vladimir Vovk, and Thomas Zeugmann

The conference "Algorithmic Learning Theory 2010" is dedicated to studies of learning from a mathematical and algorithmic perspective. Researchers consider various abstract models of the problem of learning and investigate how the learning goal in such a setting can be formulated and achieved. These models describe ways to define

- the goal of learning,
- how the learner retrieves information about its environment,
- how to form of the learner's models of the world (in some cases).

Retrieving information in some models is passive where the learner just views a stream of data. In other models, the learner is more active, asking questions or learning from its actions. Besides explicit formulation of hypotheses in an abstract language with respect to some indexing system, there are also more implicit methods like making predictions according to the current hypothesis on some arguments which then are evaluated with respect to their correctness, and wrong predictions (coming from wrong hypotheses) incur some loss on the learner. In the following, a more detailed introduction is given to the five invited talks and then to the regular contributions.

Invited Talks. The five joint invited speakers of the conferences ALT 2010 and DS 2010 are eminent researchers in their fields and give either an introduction to their specific research area or talk about a topic of wide general interest.

Alexander Clark (Royal Holloway University of London) received his bachelor degree from Trinity College, Cambridge, and his Ph.D. from the University of Sussex. He has throughout his career put a large emphasis on applying theoretical insights to solve the corresponding practical problems. In particular, he studied the unsupervised learning of natural languages; his findings are also relevant to first language acquisition in humans. His work included finding the right definition of learnability in various linguistic contexts, designing learning algorithms and implementing them. These algorithms were tested on both synthetic and natural examples. He also studied the learnability of regular languages and context-free languages; a sample result, obtained in collaboration with Franck Thollard, is that the class of regular languages can be PAC-learned using a polynomial amount of data and processing time, provided that the distributions of the samples are restricted to be generated by one of a large family of related probabilistic deterministic finite state automata. In his invited talk *Towards General Algorithms for Grammatical Inference*, Alexander Clark deals with the learning of context-free languages and multiple context-free languages. He formulates a general framework for a large class of learning algorithms for such languages and, using this framework, he reviews Angluin's classical LSTAR algorithm and compares it with various contemporary approaches.

© Springer-Verlag Berlin Heidelberg 2010

Manfred K. Warmuth (University of California at Santa Cruz) is a leading expert in computational learning theory. In his groundbreaking article "Learnability and the Vapnik-Chervonenkis dimension", he showed, jointly with Anselm Blumer, Andrzej Ehrenfeucht and David Haussler, that a class is PAC learnable if and only if it has finite Vapnik-Chervonenkis dimension, which established a close connection between these two fields. He further showed, jointly with Leonard Pitt, that there is no approximation algorithm to find the minimum consistent DFA within a polynomial bound. He made important contributions to the boosting algorithm, a very successful method to construct powerful learners. One of his main interests is on-line learning, where he originated several novel approaches and obtained fundamental theoretical results. His most recent work applies theoretical insights to the study of evolution. In his invited talk *The Blessings and the Curse of the Multiplicative Updates*, Manfred K. Warmuth considers learning settings in which parameters are updated in a multiplicative way; the advantage is that the importance of major patterns might grow exponentially, the disadvantage is that the importance of some minor pattern might go down too fast so that this pattern is wiped out although the information it contains might be needed later. The talk describes how modern machine learning algorithms try to preserve relevant information and compares this to the strategies nature has to preserve relevant genetic information during evolution. In his conclusion, the author states that there are still various strategies which one can take over from nature in order to use them in learning algorithms.

Ivan Bratko (University of Ljubljana) received his bachelor, masters and doctoral degrees from the University of Ljubljana. He is an eminent researcher in machine learning, knowledge-based systems, heuristic programming, qualitative modelling, intelligent robotics and computer chess. He is the author of the standard reference "Prolog Programming for Artificial Intelligence", which has been translated into German, Italian, French, Slovenian, Japanese and Russian; he has furthermore authored about 200 research articles. In his invited talk *Discovery of Abstract Concepts by a Robot*, Ivan Bratko reviews experiments in which a robot is experimenting in its environment. In these experiments the robot is not only required to discover laws that enable predictions, but also to discover abstract concepts that are not explicitly observable in the data measured such as the notions of a tool or stability. The approach reported is based on Inductive Logic Programming.

Kotagiri Ramamohanarao (University of Melbourne) received the Bachelor of Engineering from Andhra University, the Master of Engineering from the Indian Institute of Science and the Ph.D. from Monash University. He is a leading expert in machine learning and data mining, robust agent systems, information retrieval, intrusion detection, logic programming and deductive databases, distributed systems, bioinformatics and medical imaging. His invited talk *Contrast Pattern Mining and its Application for Building Robust Classifiers* studies the problem to distinguish, differentiate and contrast between different data sets and introduces the techniques for contrasting data sets.

Peter L. Bartlett (University of California at Berkeley) is an outstanding researcher in the areas of machine learning and statistical learning theory. Jointly with Martin Anthony, he co-authored the excellent textbook "Learning in Neural Networks: Theoretical Foundations". For his work in statistical learning theory, in 2001 he was awarded the Malcolm McIntosh Prize for the Physical Scientist of the Year in Australia. His research interests include, besides neural networks and statistical learning theory, also privacy and security aspects of learning, online learning algorithms and kernel methods. In his invited talk *Optimal Online Prediction in Adversarial Environments*, Peter L. Bartlett deals with prediction in statistical settings and investigates strategies to minimise the regret in a prediction game against an adversarial environment. He explains that, although not every environment is adversarial, it is often the mathematically most elegant way to model the prediction situation.

Statistical Learning. Statistical learning theory studies methods of assigning labels to data vectors under the statistical assumption that the labelled data vectors are generated independently by an arbitrary unknown probability distribution. Sometimes, the labels depend only on a small fraction of the variables present while most components of the data vectors are irrelevant; the area of feature selection studies methods of selecting the relevant variables (vector components) so that in the future the learning algorithm can concentrate on the data vectors compressed in this way and so easier to label. In active learning, the learner does not get the labels together with the training data; instead the learner has to request the labels from a teacher. Naturally this can be done only with a small fraction of the data presented. Boosting is a method which improves the properties of a learner by combining various primitive learners into a better one.

Pierre Alquier's paper *An Algorithm for Iterative Selection of Blocks and Features* is about selecting variables from very long vectors of variables where in the data, almost all variables are 0 and neighbouring variables are with high probability equal. This topic has been studied previously, but its theoretical treatment so far has been insufficient. To obtain better results in the area, the author proposes an alternative approach, based on the Iterative Feature Selection method. This method is based on an iterative algorithm which takes the general form of the vector to be learnt into account, but does not know the positions where the blocks start and end. The algorithm improves the statistical performance of its current guess (estimator) at each step. The obtained results are justified both theoretically and through simulations on practically relevant data.

Liu Yang, Steve Hanneke, and Jaime Carbonell study in their paper *Bayesian Active Learning Using Arbitrary Binary Valued Queries* how to learn a concept to precision ϵ using as few binary queries as possible. The authors provide an upper and lower bound on how many queries may be required to learn successfully. The model is generalised from the usual one in the sense that arbitrary binary valued queries are taken into consideration and not only membership queries. The analysis is Bayesian in the sense that the bound depends on a prior distribution on the concept space.

In their paper *Approximation Stability and Boosting*, Wei Gao and Zhi-Hua Zhou revisit the notion of stability for boosting algorithms. It is known that algorithms like AdaBoost have almost-everywhere uniform stability when the base algorithm has L_1 stability. The latter is however too restrictive: the authors show that AdaBoost using such a learner becomes a constant learner unless the underlying algorithm is a real-valued learner. Therefore the authors dedicate themselves to the question on what can be said when the base learner is not real-valued. For this analysis, they introduce a property called "approximation stability". They show that AdaBoost has this property and prove that this property is sufficient for generalisation and for the learnability of asymptotic empirical risk minimisation in the general learning setting.

Grammatical Inference and Graph Learning. This section is dedicated to learning specific structures such as formal languages, trees or graphs. These types of structures are important in mathematics and computer science and also play an important role in learning theory. Most prominent graphs occur in the internet and social networks; for example, search machines collect a lot of information on the graph structure of the internet where nodes are given by webpages and edges by the hyperlinks.

In their paper *A Spectral Approach for Probabilistic Grammatical Inference of Trees*, Raphaël Bailly, Amaury Habrard, and François Denis consider distributions over the set of trees which are computed by weighted automata. This is a quite natural class of distributions which has an algebraic characterisation. By concentrating on the finite dimensional subspace containing all the residuals of such a distribution, the authors find an approach which allows them to define a global solution for their inference problem, so that they can avoid to construct the automaton to be built iteratively step by step.

Balázs Cs. Csáji, Raphaël M. Jungers, and Vincent D. Blondel dedicate their paper *PageRank Optimization in Polynomial Time by Stochastic Shortest Path Reformulation* to the question on how a member-node of a network can increase its importance and visibility by small modifications of the network. The underlying idea is that the nodes in the network are evaluated using the well-known PageRank algorithm which, roughly speaking, assigns to every node the expected time which one spends on the node during a random walk. The task is now the following: Given a set of possible new edges, one has to select a fixed number of them and add them to the network in order to increase the PageRank. Csáji, Jungers and Blondel show that the general problem on how to select these edges is polynomial time solvable; they do this by reformulating the algorithm as a stochastic shortest path problem and they then show that this new problem is well-suited for reinforcement learning methods.

Dana Angluin, James Aspnes, and Lev Reyzin explore in their paper *Inferring Social Networks from Outbreaks* a learning setting which stems from the study of diseases. In a network, a disease might have travelled along the edges of the network. Hence whenever there is an outbreak of the disease, the disease is tracked down at the locations of its appearance and the observed locations can be considered to be connected through the network. The learning task is to build a model of the network which explains how during an outbreak the illness propagated in the network. Formally, an outbreak is a set S_i of nodes, called constraints, and the goal of the learner is to find a subset E of edges, called connections, such that each constraint S_i is connected by those members of E which are between nodes from S_i; then it is assumed that diseases can travel along the so selected edges. Here the choice of E should be optimised with respect to its minimum loglikelihood cost. In the off-line learning problem, the learner receives all the constraints S_i at the start of the algorithm; in the on-line learning problem, the learner reads the constraints one by one and each time has to add some edges in order to meet the constraint immediately. Angluin, Aspnes and Reyzin obtain a lower bound of $\Omega(\log(n))$ for the off-line version of the problem and an upper $O(n\log(n))$ bound for the on-line version. Better bounds are obtained for various special cases.

Probably Approximately Correct Learning. The basic idea of PAC learning is that the learner observes the data according to a distribution, but it does not need to figure out aspects of the concept to be learnt which are unlikely to be observed. In other words, when learning a concept L, the learner observes randomly drawn data according to some unknown probability distribution D and the learner has to find with high probability a hypothesis H such that H is similar to L with respect to the distribution D, that is, $D(\{x : H(x) \neq L(x)\})$ is below a bound given to the algorithm as a parameter.

Guy Lever, François Laviolette, and John Shawe-Taylor derive in their paper *Distribution-Dependent PAC-Bayes Priors* a number of PAC-Bayes bounds for Gibbs classifiers using prior and posterior distributions which are defined, respectively, in terms of regularised empirical and true risks for a problem. The results rely on a key bound on the Kolmogorov-Loveland divergence between distributions of this form; this bound introduces a new complexity measure.

The purpose of Vladimir Pestov's work is explained already in its title, *PAC Learnability of a Concept Class under Non-Atomic Measures: A Problem by Vidyasagar*. The characterisation of PAC learnability under the class of all non-atomic measures is achieved by introducing an appropriate combinatorial parameter modifying the Vapnik-Chervonenkis dimension. This is a natural problem (in fact it was asked by Vidyasagar 13 years ago) and the solution is non-trivial, involving techniques from set theory and measure theory.

In *A PAC-Bayes Bound for Tailored Density Estimation*, Matthew Higgs and John Shawe-Taylor consider the problem of density estimation with an unusual twist: they want their solution to be tailored to the larger inference process of which this problem is part. Formalization of this idea involves the theory of reproducing kernel Hilbert spaces. Error bounds are stated in the framework of PAC-Bayes theory, which is standard in the case of classification but rarely used in density estimation.

In their paper *Compressed Learning with Regular Concept*, Jiawei Lv, Jianwen Zhang, Fei Wang, Zheng Wang, and Changshui Zhang study the PAC learnability of half spaces where, for any given distribution, only those half spaces are considered where the measure of the bounding hyperplane is 0. The learning is called compressed as some of the components of the data presented are replaced by a random value; such compression is done for two reasons: (a) privacy as parts of the data might otherwise reveal information which should not be compromised; (b) efficiency reasons in order to reduce the overfitting in the learning process. The algorithm used is the voted-perceptron algorithm invented by Freund and Schapire.

Query Learning and Algorithmic Teaching. The basic scenario is that a learner wants to learn a concept which a teacher is teaching; in that scenario the learner has to be successful whenever the teacher satisfies the minimum requirements, that is, gives correct answers although those need not to be more helpful than required. In most settings of query learning, the queries are of a fixed form, for example the learner can ask equivalence queries to which the teacher provides a counter example in the case that the hypothesis does not match the concept to be learnt and also membership queries where the teacher answers "yes" or "no," depending on whether the queried item is an element of the concept to be learnt or not. More recent research also looks at statistical queries where an underlying distribution is assumed and the teacher returns a polynomial-time program which has — with respect to the underlying distribution — an error probability below a parameter given in the query.

Borja Balle, Jorge Castro, and Ricard Gavaldà investigate in their paper *A Lower Bound for Learning Distributions Generated by Probabilistic Automata* the limitations on the learnability of certain distributions. These distributions are generated by probabilistic deterministic finite automata (PDFA). The authors show that the learnability of such distributions using statistical queries depends on a parameter μ which is quite frequently studied in the literature, and they show that this parameter cannot be omitted without losing polynomial time learnability for various important classes; in other words, the number of queries needed depends on this parameter μ. For their results, they use in addition to statistical queries also a new variant of these called L_∞-queries.

Dana Angluin, David Eisenstat, Leonid (Aryeh) Kontorovich, and Lev Reyzin study *Lower Bounds on Learning Random Structures with Statistical Queries.* The researchers consider randomly composed DNF formulas, randomly selected decision trees of logarithmic depth and randomly constructed deterministic finite automata. They show that each of these three concept cannot be weakly learned with a polynomial number of statistical queries, where the underlying distribution on the examples is arbitrary.

The paper *Recursive Teaching Dimension, Learning Complexity and Maximum Classes* by Thorsten Doliwa, Hans Ulrich Simon, and Sandra Zilles deals with the recursive teaching dimension, which is the smallest number n such that for each concept C in the class to be learnt there are n examples $x_1, x_2, \ldots, x_n$ such that C is the only concept D in the class satisfying $C(y) = D(y)$ for all $y \in \{x_1, x_2, ..., x_n\}$. The authors show, among other results, that for maximum classes the recursive teaching dimension equals the Vapnik-Chervonenkis dimension. In addition the authors show that the sequences defining the recursive teaching dimension also arise from various famous algorithms.

On-line Learning. The basic idea of on-line algorithms is that a learner combines expert advice in the process of decision making. In each round, the experts are asked which action to take, and then the learner makes its own decision based on this advice. Experts can be free agents or just decision or prediction strategies. Typical results in this areas are relative loss bounds: the goal is to design prediction algorithms that are guaranteed to suffer a loss that is not much worse than the loss suffered by the best experts. To achieve this goal, the learner keeps some statistics on the reliability of each expert, which is taken into account when making decisions.

Student Gábor Bartók received the *E. Mark Gold Award* for his paper *Toward a Classification of Finite Partial-Monitoring Games* which is joint work with Dávid Pál and Csaba Szepesvári. A finite partial-monitoring game is a two player game; the two players are called Learner and Nature in order to express that a learner explores and studies its natural environment which reacts to the learner's actions. In this game, Learner repeatedly chooses one of finitely many actions and Nature reacts to the learner by choosing one of finitely many possible outcomes. Depending on the action and outcome, the learner receives a feedback signal and suffers a loss; the goal of the learner is to choose the actions such that the overall loss is minimised. The authors make significant progress in classifying the games with two outcomes.

The paper *Switching Investments* by Wouter M. Koolen and Steven de Rooij is devoted to mathematical finance. As usual in on-line learning, the authors do not make any statistical assumptions about the financial market, and design investment algorithms competitive with a wide class of investment strategies that "buy low and sell high". One of their algorithms, in addition, possesses linear time and space complexity.

Alexey Chernov and Fedor Zhdanov explore in their paper *Prediction with Expert Advice under Discounted Loss* a relatively new type of performance guarantees in on-line learning. In the standard approach, the learner's goal is to be competitive with the best experts according to the learner's and experts' cumulative losses. Chernov and Zhdanov, following earlier work by Freund and Hsu, establish similar results for cumulative discounted losses, where more recent losses are taken with greater weights. The framework of discounted losses provides an elegant alternative to Herbster and Warmuth's framework of "tracking the best expert".

Jacob Abernethy, Peter L. Bartlett, Niv Buchbinder, and Isabelle Stanton address in their paper *A Regularization Approach to Metrical Task Systems* the construction of randomised on-line algorithms for metrical task systems, where the learner always follows one expert and where it incurs a cost for switching from one expert to another. In the general case, the costs are an arbitrary metric among states. The authors restrict themselves to the class of "work-based" algorithms and obtain for this special case various algorithms.

Inductive Inference. The basic scenario of inductive inference is that a class $\mathcal{C}$ of recursively enumerable languages is called learnable from positive data if there is a recursive learner which can identify every language $L \in \mathcal{C}$ in the following sense: when reading the elements of L in arbitrary order from an infinite list, the learner outputs in parallel finitely many hypotheses such that the last of these generates L. Many variants of this notion of learning have been introduced and compared with each other, and the topic remains active more than 40 years after Gold's papers started it.

John Case and Timo Kötzing investigate in their paper *Solutions to Open Questions for Non-U-Shaped Learning with Memory Limitations* the question when U-shaped learning behaviour can be avoided without losing learning power. Here a learner is said to be *U-shaped* on a text for a language L in the class to be learnt if at some time it conjectures L, later conjectures a language different from L and at the end returns to conjecturing L. For basic learning criteria it is known whether U-shaped learning behaviour can be avoided: in the case of explanatory learning the answer is "yes" and in the case of behaviourally correct learning the answer is "no". But for various other learning criteria, in particular those with limitations of the long term memory, this question remained open. Case and Kötzing solve several of these open questions. One sample result is that there are classes which are learnable with three memory states such that every learner using only finitely many memory states for these classes has U-shaped learning behaviour on some text for some language to be learnt.

Samuel E. Moelius III and Sandra Zilles study in their paper *Learning Without Coding* notions of iterative learning which hinder or reduce the abilities of the learner to code. Here an iterative learner is a learner which starts with a default hypothesis and maps each datum plus the old hypothesis to the new hypothesis; the hypothesis itself is the only memory the learner has of the previously observed data. As there is some temptation for the learner to code observed data into the hypothesis, the authors look for learning models which minimise such coding by the learners. The authors investigate to which extent one can overcome such behaviour by requiring that the learner uses a one-to-one hypothesis space. Furthermore, they generalise learnability by considering learners which are coded as enumeration operators and which do not need hypothesis spaces. One sample result of the authors is that such learners can infer various classes which cannot be learnt iteratively; conversely there are also classes learnable using a one-to-one hypothesis space which are not learnable under this new model.

Mahito Sugiyama, Eiju Hirowatari, Hideki Tsuiki, and Akihiro Yamamoto give in their paper *Learning Figures with the Hausdorff Metric by Fractals* a theoretical foundation in the framework of inductive inference for learning with discretisation of analog data. They study the learnability of geometric figures, that is, fractals. The two main learnability notions employed are identification in the limit as well as closer and closer approximations of the object to be learnt where the approximations are measured with a Hausdorff metric.

Sanjay Jain and Efim Kimber analyse in their paper *Inductive Inference of Languages from Samplings* the scenario where the learner is not given all the data about the set to be learnt but only some part of it. In prior work they studied the scenario where for every language L to be learnt and every subset

X of L, the learner has to converge on a text for X to a target language L' satisfying $X \subseteq L' \subseteq L$. In this paper, rather than considering all subsets of the target language, they consider A-sampling of the target language, where A is some fixed set and A-sampling of a language L is the set formed by taking the ith least-elements of L, for $i \in A$. Results show that the choice of A has a large influence on the learnability of the class. Furthermore, the authors consider when such a learner can be constructed independently of or uniformly in A, for a collection of such sets A.

Reinforcement Learning. In reinforcement learning, a decision maker (agent) interacts with an environment (world) by an alternating sequence of actions and observations, including (occasional) rewards that should be maximised in the long run. The environment is stochastic and unknown and has to be learned. This setting encompasses most other learning scenarios, including active and passive learning.

It has been argued that the AIXI theory represents the first general and formal "optimal" but incomputable "solution" to this problem. Laurent Orseau in his paper *Optimality Issues of Universal Greedy Agents with Static Priors* challenges the optimality of AIXI. Unlike passive Solomonoff induction it is quite non-trivial to come up with notions of optimality that are simultaneously strong enough to be interesting and weak enough to be satisfiable by any agent at all. One suggested notion is to require that the probability of a suboptimal actions tends to zero, where an action is called suboptimal if it differs from the optimal action of the informed agent on the same history. Environments with this property are called asymptotically learnable. Orseau shows that there exist histories and environments that AIXI cannot learn asymptotically, hence establishing that this optimality notion is too strong.

At the other end of the spectrum are efficient but limited reinforcement learning algorithms. In particular, efficient learning and planning algorithms exist for completely observable finite state Markov decision processes (MDPs). Real-world problems can often be approximately modeled or reduced to finite MDPs. A natural idea is to formally define the quality of a reduction and to automatically learn good reductions by optimizing the quality criterion. Peter Sunehag and Marcus Hutter in their paper *Consistency of Feature Markov Processes* investigate a recently introduced such criterion. They show asymptotic consistency in the sequence prediction case, and extend their result to prediction with side information and to the active case.

Multi-armed bandit problems can be regarded as (reinforcement) learning problems with a single state. Despite their apparent simplicity, they constitute prototypical active learning problems that already require trading off exploration and exploitation. Taishi Uchiya, Atsuyoshi Nakamura, and Mineichi Kudo in their paper *Algorithms for Adversarial Bandit Problems with Multiple Plays* consider the non-stochastic / online / adversarial setting and the case where multiple arms are played simultaneously, which is relevant, e.g., for multiple advertisement placement. They analyze and present bounds for extensions of the

Exp3 and CompBand algorithms in terms of time and space efficiency and regret for the best fixed action set.

On-line Learning and Kernel Methods. The papers in this part of the proceedings are in the intersection of two areas: on-line learning and kernel methods. Kernel methods were popularised in machine learning by the work of Vladimir Vapnik on support vector machines, but later became a powerful tool used in many other areas of learning theory. The basic idea is the so-called "kernel trick": the instances (typically low-dimensional) are mapped to a high-dimensional (often infinite-dimensional) *feature space*, where the prediction is done and its analysis is performed. Popular methods used in the feature space are separating positive and negative examples with a large-margin hyperplane (in the case of classification) and fitting a linear function to the data (in the case of regression). Even conventional linear methods become a powerful tool when applied to high-dimensional feature spaces, and when mapped back to the original instance space, they may become highly non-linear and yet computationally efficient.

The paper *Online Multiple Kernel Learning: Algorithms and Mistake Bounds* by Rong Jin, Steven C. H. Hoi, and Tianbao Yang constructs a number of on-line kernel algorithms for classification and provides relative loss bounds for them. Its goal is to merge classifiers based on several different kernels. The performance of the resulting algorithms should be comparable with that of the algorithm based on the best kernel; this is a non-trivial problem since which kernel is best becomes known only after we see the data. The authors construct both deterministic and randomised versions of such algorithms, the latter achieving computational efficiency by applying ideas from the popular area of "multi-armed bandits" (see above).

Fedor Zhdanov and Yuri Kalnishkan analyze in their work *An Identity for Kernel Ridge Regression* properties of the popular method of kernel ridge regression as an on-line prediction algorithm. The main result of the paper is the equality between the quadratic loss (suitably reduced) of the kernel ridge regression algorithm applied in the on-line mode and the quadratic loss of the best regressor (suitably penalised). This new identity makes it possible to derive, in an elegant way, upper bounds for the cumulative quadratic loss of online kernel ridge regression.

Towards General Algorithms for Grammatical Inference

Alexander Clark

Department of Computer Science
Royal Holloway, University of London
alexc@cs.rhul.ac.uk

Abstract. Many algorithms for grammatical inference can be viewed as instances of a more general algorithm which maintains a set of primitive elements, which distributionally define sets of strings, and a set of features or tests that constrain various inference rules. Using this general framework, which we cast as a process of logical inference, we re-analyse Angluin's famous LSTAR algorithm and several recent algorithms for the inference of context-free grammars and multiple context-free grammars. Finally, to illustrate the advantages of this approach, we extend it to the inference of functional transductions from positive data only, and we present a new algorithm for the inference of finite state transducers.

1 Introduction

Grammatical inference (GI) is concerned with learning various types of formal languages. In this paper, we consider two classic problems: the first is where we are learning *languages*, that is to say sets of strings over a finite alphabet — subsets of Σ^*; and the second is where we are learning transductions or functions – relations between pairs of strings — a subset of $\Sigma^* \times \Delta^*$. In the first case we will receive information about the language, typically in the form of a sequence of positive examples drawn from the language; in the second we will receive input-output pairs, and in both cases we wish to construct a representation of the language or function from this information. At a certain point, as the amount of information we have increases, we wish our representation to converge exactly to a correct hypothesis; that is to say, a hypothesis that exactly defines the target concept.

We will work here in the Gold paradigm [1], which is mathematically convenient although unrealistic as a model of learning in the real world. We assume that the learner is provided with a sequence of examples from the target language L subject only to the constraint that every string in the language must occur at least once. We will denote such a sequence $w_1, w_2, \ldots$. After processing each example, the learner must produce a representation — $G_1, G_2, \ldots$. We require that for each such sequence, or presentation, there must be some finite point N after which the learner no longer changes the hypothesis, and such that the hypothesis is correct: $\forall n > N$, $G_n = G_N$ and $L(G_N) = L$.

There are two main problems with learning the sorts of richly structured representations that are required to model natural languages. The first are the sorts

© Springer-Verlag Berlin Heidelberg 2010

of information theoretic problems that have been well-studied in many areas of learning – these problems concern whether the learner has enough information about the target to produce an accurate enough hypothesis. Concepts such as finite elasticity, VC-dimension and so on have been developed in attempts, largely successful, to characterise those cases where learning is impossible simply because there is insufficient information to allow a learner to pick out the right hypothesis. Such results consider the learner primarily as a *function* rather than as an *algorithm*: if there is no suitable function then there can of course not be an algorithm that implements that function.

The second class of problems are caused by the complexity involved in constructing a hypothesis given an adequate source of information about the language [2, 3]. These are two rather different types of problems, and in our opinion it is appropriate to try to solve them separately.

In this paper we will focus almost exclusively on the algorithmic aspects of learning — in an attempt to overcome these complexity problems — and we will therefore give ourselves a rich source of information. In addition to the positive examples we will assume that the learner has access to an oracle that can answer membership queries: the learner can construct a string w and query the oracle with this string; the oracle will return *true* if $w \in L$ and false if it is not. This is an extremely powerful source of information. It is easy to see that if we put no constraints on the amount of computation that we use, there are trivial enumerative algorithms that the learner could use to learn any enumerable class of recursive languages.

2 Objective Representations

We will now consider various algorithms for grammatical inference. These algorithms all start by constructing the representation based on objectively defined sets of strings; we discuss the methodological and representational basis of this approach in more detail in [4].

We define representations where the symbols of the representation – non-terminals, states etc – have well defined referents as sets of strings, or sets of tuples of strings. Once we have fixed what each of these symbols represents, we can think of the derivation rules of the grammar, the transitions, or productions etc – as being logical deductive relationships between these sets. Some rules will be valid – in the sense that the deductive relationships will be correct— and others will be incorrect, in the sense that we may deduce that a string is in a set when in fact it is not.

The crucial point is that once we have fixed what each representational primitive is meant to do – that is to say defined what set of strings it should generate or produce – each decision about the model becomes a local decision rather than a global one. In the classic representations of the Chomsky hierarchy, the primitive symbols are arbitrary. There is no fixed definition for what each symbol need represent. In a normal CFG, if there is a rule $N \rightarrow PQ$, there is no way of evaluating that rule in isolation from the rest of the grammar. It is only as part of a global inference about the rules that generate N, and the strings that are generated from P and from Q that we can decide whether this is a good rule or not. This global inference is generally intractable. In the models we consider here, we define in advance what the symbols should "mean". Once we have done this, we can determine for each indivdual production in the grammar whether it is valid or not. This local inference procedure is tractable and indeed is often quite trivial.

We define now the notation we will use in the rest of the paper. We will use Σ (and Δ) to refer to non-empty finite alphabets. We will use Σ^* to refer to the set of all strings over Σ; λ is the empty string. Given two strings u, v we denote their concatenation by uv. A context (l, r) is an element of $\Sigma^* \times \Sigma^*$; we can combine a context with a string by wrapping it around the string: we denote this by $(l, r) \odot u = lur$. A language L is any subset of Σ^*. Given two subsets of Σ^*, X and Y, we define their concatenation in the normal way as $XY = \{uv|u \in X, v \in Y\}$. Given a language L we define the distribution of u in L as $C_L(u) = \{(l, r)|lur \in L\}$. For a string w we define $Sub(w)$ to be the set of all substrings of w, $\{u|\exists l, r \in \Sigma^*, lur = w\}$, and $Con(w) = \{(l, r)|\exists u \in \Sigma^*, lur = w\}$. We extend this to sets of strings in the natural way.

The models maintain two classes of objects: the first is a set of primitive elements; we will denote these by Q. These correspond to the states or non-terminals in standard representations. The second are a class of tests or experiments, which we denote X. These are used to restrict and eliminate incorrect rules.

Each primitive element from Q will define a set of strings given a language L; or more generally a tuple of strings. For a given element $p \in Q$, we will denote by $\mathbf{D}(p)$ the set of strings defined by p; to avoid confusion we shall write $[[p]]$ to refer to the symbol as used in the representation. The definition of $\mathbf{D}(p)$ will determine what the representation class is; different representational decisions will give rise to different representation classes. In many cases, the elements of p will be strings, and then some of the possibilities for $\mathbf{D}(p)$ are as follows:

$$\begin{array}{rl}
\{w|wp \in L\} & \text{left quotient of } p \\
\{w|pw \in L\} & \text{right quotient of } p \\
\{w|C_L(w) = C_L(p)\} & \text{congruence class of } p \\
\{w|C_L(w) \supseteq C_L(p)\} &
\end{array}$$

For each algorithm, we will pick one of these; different decisions will give rise to different representation classes. For example, we might pick the first of these; $\mathbf{D}(p) = \{w|wp \in L\}$. This will, as we shall see, lead us naturally to a representation for regular languages. Clearly $\mathbf{D}(\lambda) = L$ no matter what L is. If the language $L = (ab)^*$, then $\mathbf{D}(a) = b(ab)^*$, $\mathbf{D}(b) = \emptyset$, $\mathbf{D}(ab) = L$ and so on.

We might also define Q to be a set of pairs of strings. If we do this, and we write an element as $p = (u, v)$ we might have

$$\mathbf{D}((u, v)) = \{w|uwv \in L\} \qquad (1)$$

Given these sets of strings, we can then define a logical derivation relation. We wish to define a set of deductive relations between these sets of strings. Given a language L and a set Q, we will have the family of sets $\mathbf{D}(Q) = \{\mathbf{D}(p) | p \in Q\}$. The simplest deductive relation we could have is this: suppose that $\mathbf{D}(p) \subseteq \mathbf{D}(q)$. Then if we know that a string u is in $\mathbf{D}(p)$, we can deduce that it is in $\mathbf{D}(q)$.

Similarly if we know that u is in $\mathbf{D}(p)$ and we know that v is in $\mathbf{D}(q)$, and we know that $\mathbf{D}(p)\mathbf{D}(q) \subseteq \mathbf{D}(r)$ then we can deduce that $uv \in \mathbf{D}(r)$.

Finally if we know for example, that $\mathbf{D}(p)$ is a subset of L, or in particular if $\mathbf{D}(p) = L$, then if we have deduced that $u \in \mathbf{D}(p)$, then we can deduce that $u \in L$. Obviously these deductions must start somewhere – there must be a few base facts where we know that $u \in \mathbf{D}(p)$; typically we will know these for a few short strings, at least the elements of Σ.

Thus the derived language representations will work by trying to predict, on the basis of these deductive relationships, which elements of $\mathbf{D}(Q)$ a particular string is in. In some cases the elements of this class may be a partition, and a string can only occur in a single element, but in general they may overlap, and a string may be in more than one of the classes. A derivation is therefore a deduction; bringing to mind the "Parsing as deduction" slogan [5]. We assume for the moment that we know which of these deductions are valid.

We will have various different rule schemas. We will now list some of these, though these by no means exhaust all of the possibilities. The notation we use is that p is an element of Q, $[[p]]$ represents the corresponding symbol, and $\mathbf{D}(p)$ represents the set of strings defined by p.

Type L (Lexical)
$[[p]] \to u$. This allows us to deduce that a string u is in $\mathbf{D}(p)$. It is valid iff $u \in \mathbf{D}(p)$. We will consider two special cases:
L0 $[[p]] \to \lambda$
L1 $[[p]] \to a$, where $a \in \Sigma$

Type R (Regular) $[[p]] \to u[[q]]$. This is valid if $u\mathbf{D}(q) \subseteq \mathbf{D}(p)$. We have the special case:
R1 $[[p]] \to a[[q]]$, where $a \in \Sigma$

Type LIN (Linear) $[[p]] \to u[[q]]v$. This is valid if $u\mathbf{D}(q)v \subseteq \mathbf{D}(p)$. We will consider also the following special cases:
Type EL (Even Linear) $[[p]] \to u[[q]]v$ where $|u| = |v|$
Type EL1 $[[p]] \to a[[q]]b$ where $a, b \in \Sigma$

Type B (Binary Branching) $[[p]] \to [[q]][[r]]$. This allows us to deduce that if a string $u \in \mathbf{D}(q)$ and $v \in \mathbf{D}(r)$, then $uv \in \mathbf{D}(p)$. This is valid iff $\mathbf{D}(q)\mathbf{D}(r) \subseteq \mathbf{D}(p)$.

Type S (Subset) $[[p]] \to [[q]]$. This allows us to deduce that if a string $u \in \mathbf{D}(q)$ then $u \in \mathbf{D}(p)$. This is valid iff $\mathbf{D}(q) \subseteq \mathbf{D}(p)$.

Type E (Equality) $[[p]] \leftrightarrow [[q]]$ or both $[[p]] \leftarrow [[q]]$ and $[[p]] \to [[q]]$. This allows us to deduce that if a string $u \in \mathbf{D}(q)$ then $u \in \mathbf{D}(p)$ and vice-versa. This is valid iff $\mathbf{D}(q) = \mathbf{D}(p)$.

Type C (Conjunction) $[[p]] \to [[q]] \wedge [[r]]$. This allows us to deduce that if a string $u \in \mathbf{D}(q)$ and u is also in $\mathbf{D}(r)$ then $u \in \mathbf{D}(p)$. This is valid iff $\mathbf{D}(q) \cap \mathbf{D}(r) \subseteq \mathbf{D}(p)$.

We will also have one final class of rules, where we use the distinguished start symbol S, just as we do in CFGs. In some cases we may have an element i of Q that such that $\mathbf{D}(i) = L$ by definition, in which case we could dispense with these initial rules and the separate symbols.

Type I (Initial) $S \rightarrow [[p]]$. This is valid iff $\mathbf{D}(p) \subseteq L$.

We will have a collection of these rules which we call G: this collection will consist of the representation for the language. Given a collection of primitive elements, Q, we can consider the set of all valid rules that relate these primitive elements. Some of these rule schemas may give rise to unbounded numbers of rules; in particular the first three schemas (**L,R,LIN**) will need to be bounded in some way. Typically we will restrict these rules to those where u has length at most 1.

An important aspect of this model is the use of the **E** rules. In many of these models we will have a set of primitive elements Q, where several different elements of Q will define the same sets; that is to say we might have $p, q \in Q$, and $\mathbf{D}(p) = \mathbf{D}(q)$. A natural way to handle this is to consider the primitive elements to be equivalence classes of Q, under equality of $\mathbf{D}(q)$; or equivalently considering them to be the elements of $\mathbf{D}(Q)$. This certainly gives some efficiency gains when implementing them. However, Yoshinaka [6] introduced the idea of using "chain rules" – these equality or **E** productions — to link distinct elements of Q that define the same strings. This greatly simplifies the analysis of the algorithms, and while it causes some decrease in efficiency it does not change the polynomial nature of the algorithms. Accordingly we will adopt this refinement.

For a given decision about what $\mathbf{D}(p)$ is we can divide these rule schemas into three types. There are those that we can be sure are correct, as a result of the way $\mathbf{D}(p)$ has been defined – we will call these rules *a priori*. There are those that we are certain are correct as a result of information that has already been received — we will call these rules *certain*. Finally there are those which we believe to be correct but might later turn out to be incorrect on the basis of further information – we will call these *defeasible*. Defeasible rules will be assumed to be correct until we receive a piece of information that tells us that the rule is incorrect. Once we have seen such a piece of information, we will be certain that the rule is incorrect, and no further information will cause us to change our mind – once we know that a rule is incorrect then it is definitely wrong, and until that point we will consider it to be true, though uncertain.

For example, if we have a rule of the type $[[p]] \rightarrow u$, then it might be the case that $\mathbf{D}(p)$ is defined to include u; for example p might be u, and $\mathbf{D}(p)$ might be an equivalence class that includes u. This would be a case where the rule is *a priori*. Alternatively, we might have received information that tells us definitely that $u \in \mathbf{D}(p)$; for example, $\mathbf{D}(p)$ might be defined to be the set of all strings that occur in a given context, in which case, if we know that that string occurs in a given context then we will be certain, as a result of this single piece of information, that the rule is correct.

Finally, and most importantly, there may be rules that no finite amount of information can make certain – the defeasible rules. In general, checking these rules will involve checking potentially infinite sets. For example, a rule of the form $[[p]] \rightarrow [[q]]$, is only valid if $\mathbf{D}(q) \subseteq \mathbf{D}(p)$ and these two sets will in general be infinite. In general given only a finite amount of information it will always be possible that the language is different from what we would expect. There are an infinite number of possible languages, and it might be that the language does not include some very long strings that we would expect it to contain based on the examples we have seen. This is a possibility that we can never conclusively exclude, at least in the learning models that we consider here.

We therefore use a finite set of experiments that we denote by X, and that we will gradually increase during the learning process. Typically, X will be a set of strings or contexts that we will use to test the validity of rules. In one formalisation, X is a set of strings and we can test whether $\mathbf{D}(q) \subseteq \mathbf{D}(p)$ by testing $X \cap \mathbf{D}(q) \subseteq X \cap \mathbf{D}(p)$. Clearly if the inference is invalid, then when X is sufficiently large we will detect this fact: if it is not the case that $\mathbf{D}(q) \subseteq \mathbf{D}(p)$, then there must be some $x \in \mathbf{D}(q) \setminus \mathbf{D}(p)$, and if $x \in X$ we will detect it by testing whether $X \cap \mathbf{D}(q) \subseteq X \cap \mathbf{D}(p)$.

Initially, we assume that all possible defeasible rules are valid unless they are explicitly contradicted by a piece of information; typically an element or elements of X. We will start with X being either empty or consisting of a small set of elements, often only one. We will monotonically increase X based on the examples that we observe from the language we are trying to learn. During the course of the algorithm X will generally be increased without limit.

As we increase X we will remove incorrect defeasible rules; correct defeasible rules will never be removed. For each incorrect defeasible rule it will suffice to find a single element of X that will remove that rule; therefore in the best case we only need to have an X that is of the same cardinality as the set of possible rules, which will typically be bounded by a polynomial in the size of Q. Whether this is possible or not depends on the learning model; under the Minimally Adequate Teacher (MAT) model we will receive counter-examples [7] and generally we can construct a suitable element of X from the counter-example. If we have only positive examples, we can increase X without limit. The larger X is the more incorrect rules we will remove. The only limit is that the size of X be bounded appropriately so that the overall algorithm is efficient.

We therefore have a deductive system or grammar G that we construct from information about a language L using a defined set of primitive elements Q and a set of tests or experiments X. We will write $G(Q, X, L)$ for this system. Typically L is fixed, and so we will write this as $G(Q, X)$, but it is implicitly used in the definition since we will use an oracle for L when constructing the system G.

3 Derivation

Having constructed this inference system, we are clearly interested in using it to infer properties of novel strings that we have not observed before. Given any string w we wish to be able to tell whether it is in a particular class, and indeed whether it is in L or not; therefore we wish to be able to say for any primitive element q and any string w whether $w \in \mathbf{D}(q)$ or not.

We now consider a derivation relation or proof. Clearly we can chain these inferences together in the natural way. The inference rules allow us to deduce that a long string is in a particular class from the fact that its substrings are in another class; using this, together with a set of rules that tell us which classes the strings of length 1 are will allow us to construct efficient inference procedures. This is a standard insight from logical grammar formalisms; [5]. Thus we consider the grammatical formalisms here to be inference systems between distributionally defined sets of strings. We will not formalise the inference system using a sequent calculus; this seems unnecessarily complex.

These deductive procedures turn out to be the same as the derivation procedures in various forms of grammars, such as context free grammars, conjunctive grammars and so on. Standard techniques for dynamic programming can be used to compute these efficiently. In particular, for a given string w, we will construct a table which maintains for each string u which is a substring of w, a list or set of the elements of Q, p such that we have a proof that u is in $\mathbf{D}(p)$. This can be done in time polynomial in the length of the string and the size of Q.

We will write $[[p]] \overset{*}{\Rightarrow}_G w$ if there is a proof that $w \in \mathbf{D}(p)$ using steps in the set of productions or rules of the grammar G. If all of the inference steps are valid then it is clear that we will only deduce that $[[p]] \overset{*}{\Rightarrow}_G w$ if w is in fact in $\mathbf{D}(p)$. As a special case, we will only have a proof $S \overset{*}{\Rightarrow}_G w$ if w is in L.

Lemma 1. *If all of the inference steps in P are valid, then if $[[p]] \overset{*}{\Rightarrow} w$ then $w \in \mathbf{D}(p)$.*

Proof. Immediate by induction on the length of the proof; if each inference is valid.

Clearly as we increase Q, and we assume that all of the rules are valid, then the language will only increase, as the set of rules will increase.

The next step is to establish that the rules are in fact valid. First of all note that as we increase the size of X, the set of tests, the set of rules will monotonically decrease as we will remove defeasible rules. The following lemma is thus immediate.

Lemma 2. *If $X_1 \subseteq X_2$ then $L(G(Q_1, X_1)) \supseteq L(G(Q_1, X_2))$*

Morover, given that there are only a finite number of defeasible rules, at some point we will have removed all incorrect rule. We formalise this as a property of the set of experiments which we call *fiduciality.*

Definition 1. *A set of experiments X is fiducial for a set of primitive elements Q iff every rule is valid; i.e. all incorrect defeasible rules have been removed.*

As we increase the number of primitive elements, the set of rules will monotonically increase, even if some of them are not valid.

Lemma 3. *If $Q_1 \subseteq Q_2$ then $L(G(Q_1, X)) \subseteq L(G(Q_2, X))$*

Moreover if we have X_1 fiducial for Q_1 and X_2 fiducial for Q_2 then $L(G(Q_1, X_1)) \subseteq L(G(Q_2, X_2))$.

We now define a very fundamental idea; at some point the set of primitive elements may be large enough, so that the set of correct rules will define the language. When we reach this point, if we have some additional incorrect defeasible rules then we will overgenerate; indeed no matter how large or small X is, we will always define a language which includes the target language. We formulate this idea as follows:

Definition 2. *A finite set of primitive elements Q is a kernel for the target language L, if the set of valid rules derived from Q is sufficient that for every string $w \in L$, we have a proof using these rules that $S \overset{*}{\Rightarrow} w$.*

An easy consequence of the definition is that given any language L, then any $G(Q, X)$ where Q is a kernel for L will define a language that includes the target language L – since we will have enough correct rules, and possibly some defeasible incorrect rules if X is too small.

For any specific algorithm, the learnable class that we will have will be defined as the set of languages which have a finite kernel. This is, broadly speaking, the set of languages that can be finitely defined under the representational assumptions that we make. As we shall see, for the case of regular grammars, the class corresponds exactly to the class of regular languages, but for other representation classes, the classes of languages do not correspond precisely to existing language classes.

We will consider various specific models: but they all satisfy the following criteria.

– As we increase X, the set of rules will monotonically decrease.
– No correct rules will be removed by increasing X
– Every incorrect rule will be removed.
– As we increase Q, the set of rules will monotonically increase.
– We can perform all of the computations in polynomial time. In particular we can compute the derivation relations $S \overset{*}{\Rightarrow}_G w$ in time polynomial in the size of the rule set and the length of w.

4 Generic Algorithms

We can now define a generic algorithm for inferring these representations. We will use the paradigm of identification in the limit from positive data and (optionally) membership queries which is easy to handle and quite permissive. Algorithm 1 receives a stream of positive examples, and may use a membership oracle O. It calls several functions:

– **InitQ** returns an initial set of primitive elements.
– **InitX** returns an initial set of experiments.

– **Make** constructs a representation from the available data. We will consider this to be an inference system and thus a collection of rules.
– **IncreaseQ** returns a set of primitive elements
– **IncreaseX** returns a set of experiments.

These must satisfy the following conditions: if there is a kernel for a language, then **IncreaseQ** must eventually return a set that includes a kernel. Secondly, for any incorrect defeasible rule, **IncreaseX** must eventually produce an element that will remove it. **Make** simply produces all possible rules from Q and then removes those that are contradicted by elements of X.

```
1  $E \leftarrow \emptyset$ ;
2  $Q \leftarrow$ **InitQ** ;
3  $X \leftarrow$ **InitX** ;
4  $G \leftarrow$ **Make**$(Q, X, O, E)$ ;
5  while $w_i$ is a positive example do
6      $E \leftarrow E \cup \{w_i\}$ ;
7      for $w \in E$ do
8          if not $S \overset{*}{\Rightarrow}_G w$ then
9              $Q \leftarrow Q \cup$ **IncreaseQ**$(E)$ ;
10     $X \leftarrow X \cup$ **IncreaseX**$(w_i)$ ;
11     $G =$ **Make**$(Q, X, O, E)$ ;
```

Algorithm 1. Generic meta-algorithm for learning from positive data and membership queries. O is a membership oracle for the language.

We can now see that given a specific set of representational assumptions that the algorithm defined here will identify in the limit any language which has a finite kernel. We state the theorem given some set of definitional assumptions, and given definitions of the subroutines called by the algorithm.

Theorem 1. *Algorithm 1 will identify in the limit any language with a finite kernel.*

Proof. We will use E_n, Q_n, X_n, G_n to refer to the state of the variables at iteration n. Note first that if there is some n such that Q_n is a kernel it will never change, and the grammar will always define a superset of the target language. If at some point n there is a kernel then at some point $n' > n$, all incorrect defeasible rules will have been removed, and at that point $G_{n'}$ will be correct and will never change. So we merely need to show that at some point we will get a kernel. If L has a finite kernel, then let N be the smallest number such that **IncreaseQ**$(\{w_1, \ldots w_N\})$ is a kernel. Suppose Q_N is not a kernel. Suppose $L(G_N)$ does not include L, then at some point $n \geq N$ we must find a w_n which will call line 1, and after that point Q is a kernel. Alternatively, suppose $L(G_N)$ does include L, and since it is not a kernel it must include some incorrect rules.

Then either there is some point when all incorrect rules will be removed, at which point the hypothesis will under-generate and we will observe a string which will trigger line 1, or line 1 will be triggered before that point, and in either case we end up with a kernel.

5 Regular Languages

We will now make this rather abstract discussion concrete by producing a reconstruction of Angluin's celebrated LSTAR algorithm in this model. We will not try to make this algorithm efficient; it will be polynomial, but we will not constrain the representation to be deterministic and as such the algorithm will be much less efficient that the LSTAR algorithm and will not have the elegant algorithmics of that approach.

Our representational primitives will be strings, and so Q will be a finite set of strings that will correspond to prefixes of the language. Each prefix w will define a quotient or residual language as follows:

$$\mathbf{D}(w) = \{v | wv \in L\} \tag{2}$$

In terms of a DFA, each element of Q will therefore correspond to a state q. If we let Q_* denote the set of states of a minimal DFA that generates the languages L_*, then for each $w \in Q$, we will have a corresponding state in the DFA which will be the state $\delta(q_0, w)$, using standard notation.

We will use the following rule schemas:

I $S \rightarrow [[\lambda]]$
R $[[w]] \rightarrow v[[wv]]$ if $w, wv \in Q$
L0 $[[w]] \rightarrow \lambda$ if $w \in L$
E $[[w]] \rightarrow [[v]]$ iff $\mathbf{D}(w) \cap X = \mathbf{D}(v) \cap X$

Note that each of these four rules have slightly different properties. The first two rule schemas are universally valid – we know that they are correct *a priori* without using any evidence from the oracle. The first schema is clearly correct since by definition $\mathbf{D}(\lambda) = L$. The second schema is correct since $v\mathbf{D}(wv) \subseteq \mathbf{D}(w)$. The "proof" is trivial: suppose $u \in \mathbf{D}(wv)$; this means that $wvu \in L$. Therefore $vu \in \mathbf{D}(w)$.

The third rule schema is also certain, but uses information from the oracle. If $w \in L$ then $\lambda \in \mathbf{D}(w)$, but it is only when we have tested the example w for membership that we will know that the rule is valid.

The final **E** schema is non-trivial: it uses information from the oracle but is defeasible. Given two strings in Q, w and v we will assume that they are equivalent, (i.e. $\mathbf{D}(w) = \mathbf{D}(v)$) unless we observe some string $s \in X$ such that $ws \in L$ and $vs \notin L$ or vice-versa. Once we observe such a string then we remove this equality rule, as we know it is not valid.

Given a membership oracle for a language L, a finite set of strings Q and a finite set of test suffixes X, we can construct a regular grammar based on these rules schemas in time polynomial in the size of Q and X.

Let us now consider the notions of *kernel* and *fiduciality* in this concrete case.

Lemma 4. *The class of languages that have a finite kernel in this case is equal to the class of regular languages.*

Proof. Clearly if it has a finite kernel then it is regular since it will be correctly defined by a regular grammar. Conversely if L is a regular language, then consider a minimal DFA for L. A finite set of strings Q is a kernel for L if Q contains one string for each state in the minimal DFA and a string for each transition. That is to say for each transition $q \rightarrow^a q'$ there is a string u and a string ua in Q such that $\delta(q_0, u) = q$ and $\delta(q_0, ua) = q'$.

Thus the idea of a kernel is closely related to that of a structurally complete sample as defined in for example [8]. Indeed, the set of prefixes of a structurally complete sample for an automaton will be a kernel for the language defined by that automaton.

Lemma 5. *A set X is fiducial for a set of primitives Q iff for every pair of strings that are not congruent there is an element of X that is in the symmetric difference of their quotient languages.*

5.1 Even Linear Grammars

Recall that an even linear grammar (ELG) is a CFG where all of the productions are either of the form $X \rightarrow uYv$ where $u, v \in \Sigma^+$ and $|u| = |v|$, or of the form $X \rightarrow u$ where $|u|$ is even. We can clearly convert these to regular grammars by "folding" them over and mapping them to automata over an alphabet consisting of pairs of letters [9].

We can also can model them directly in this approach by considering the primitive elements Q to be pairs of strings (u, v) where $|u| = |v|$, and considering the experiments X to be strings of even length.

6 Congruence Based Approaches

Let us now move onto the theory of context free grammatical inference, in particular the theory of congruence based approaches as explored in [10, 11, 6, 12].

The most basic of these models, presented in [12] makes the representational assumption that the non-terminals of the gramar generate congruence classes of the language.

Recall that the syntactic congruence is defined as the relation $u \equiv_L v$ iff $C_L(u) = C_L(v)$. We will define Q as a set of strings, and for each $u \in Q$, we define $\mathbf{D}(u) = \{w : C_L(u) = C_L(w)\}$. These are the congruence classes of the language L. The set of experiments X will be a finite set of contexts; i.e. $X \subset \Sigma^* \times \Sigma^*$.

We will therefore have the following families of rules

B $[[uv]] \rightarrow [[u]][[v]]$
L0 $[[a]] \rightarrow a$

L1 $[[\lambda]] \rightarrow \lambda$
E $[[u]] \rightarrow [[v]]$ iff $C_L(u) \cap X = C_L(v) \cap X$
I $S \rightarrow [[u]]$ iff $u \in L$

In terms of the meta-algorithm 1 we will define the following functions:

InitQ returns $\Sigma \cup \{\lambda\}$
InitX returns $\{(\lambda, \lambda)\}$
Make Returns a CFG with the productions defined above
IncreaseQ returns $Sub(E)$
IncreaseX returns $Con(E)$

In [12], a similar algorithm was shown to polynomially learn the class of congruential CFGs from a minimally adequate teacher (MAT).

7 Dual CFG Representations

In regular inference we are concerned with the relation between the prefix and the suffix. Given a language L, we define a relation $u \sim_L v$ iff $uv \in L$: the dual relation is basically the same except that we swap prefixes and suffixes. This leads to representations where we have finite automata that read from right to left – this is uninteresting.

In distributional learning we can find that there is a partial duality between the context and the substring. We will now consider a dual representation, where the primitive elements are contexts, and the set of experiments X is a set of substrings.

We will consider Q as a finite set of contexts, i.e. a subset of $\Sigma^* \times \Sigma^*$, and we shall assume that $(\lambda, \lambda) \in Q$. We now define, for a context $p = (l_p, r_p)$ in Q

$$\mathbf{D}((l_p, r_p)) = \{u | l_p u r_p \in L\} \tag{3}$$

Note that $\mathbf{D}((\lambda, \lambda)) = L$. We will have as before various classes of rules. The defeasible class of rules is thus the class of binary rules of the form $[[p]] \rightarrow [[q]][[r]]$. We will test these using the following criterion.

$(\mathbf{D}(q) \cap X)(\mathbf{D}(r) \cap X) \subseteq \mathbf{D}(p)$

We can test this simply using a membership oracle by checking for each $u, v \in X$ such that $q \odot u \in L$ and $r \odot v \in L$, and if they are then we test whether $p \odot (uv) \in L$; if this is not the case then we remove the rule. Otherwise we include the rule.

The basic rules are thus

I $S \rightarrow [[(\lambda, \lambda)]]$
L1 $[[p]] \rightarrow a$ iff $p \odot a \in L$
L0 $[[p]] \rightarrow \lambda$ iff $p \odot \lambda \in L$
B $[[p]] \rightarrow [[q]][[r]]$ iff $(\mathbf{D}(q) \cap X)(\mathbf{D}(r) \cap X) \subseteq \mathbf{D}(p)$

Only the final rule is defeasible. we need to show that all and only the incorrect rules will be removed. Suppose that the rule $[[p]] \to [[q]][[r]]$ is incorrect. Then this means that $\mathbf{D}(q)\mathbf{D}(r)$ is not a subset of $\mathbf{D}(p)$. Therefore there are strings u, v such that $u \in \mathbf{D}(q)$ and $v \in \mathbf{D}(r)$ but $uv \notin \mathbf{D}(p)$.

If we define $\mathbf{IncreaseX}(E)$ to return the set of all substrings in E, then clearly once E includes $q \odot u$ and $r \odot v$, this incorrect rule will be removed. The converse is obvious; if the rule is valid then $(\mathbf{D}(q) \cap X)(\mathbf{D}(r) \cap X) \subseteq \mathbf{D}(p)$ is always true even when $X = \Sigma^*$. Note that the class of languages learnable is rather different as it will include non-deterministic and inherently ambiguous ones, whereas the algorithm of Section 6 appears to only include deterministic languages.

This algorithm corresponds to the algorithm defined in [13] restricted to the case where we consider only single contexts. It is instructive to contrast the primal, congruence-based algorithm with this dual algorithm for context-free inference. For the primal representation, the **B** rules are *a priori* and the **E** rules are defeasible and the **L** and **I** rules are certain; for the dual representation, the **B** rules are defeasible, the **L** rules are certain, the **I** rules are *a priori*, and we do not use **E** rules.

Table 1. Table showing the basic representational assumptions of the models. All models also have **I** rules, so we omit them. In Yoshinaka's algorithm for MCFGs, the range of **B** rules used is much wider. The final column gives the class of representation that is used. OSST stands for onward subsequential transducer.

Model	Q	X	$\mathbf{D}(p)$	Rules	Class
Angluin [7]	Σ^*	Σ^*	$\{w \mid pw \in L\}$	**L0, R,E**	DFA
Sempere [9]	$\Sigma^k \times \Sigma^k$	Σ^*	$\{w \mid p \odot w \in L\}$	**L0, EL1, E**	ELG
Clark et al. [14]	Σ^*	$\Sigma^* \times \Sigma^*$	$\{w \mid C_L(w) \supseteq C_L(p)\}$	**L, S,B,C**	BFG
Clark [12]	Σ^*	$\Sigma^* \times \Sigma^*$	$\{w \mid C_L(w) = C_L(p)\}$	**L1,L0,B,E**	CFG
Clark [13]	$(\Sigma^* \times \Sigma^*)^{\leq k}$	Σ^*	$\{w \mid C_L(w) \supseteq p\}$	**L1,L0,B**	CFG
Clark [15]	$(\Sigma^* \times \Sigma^*)^*$	Σ^*	$\{w \mid C_L(w) \supseteq p\}$	**L1,L0,B,S,C**	DLG
Yoshinaka [16]	$(\Sigma^*)^{\leq k}$	$(\Sigma^*)^{\leq (k+1)}$	$\{\mathbf{w} \mid C_L^k(\mathbf{w}) \supseteq C_L^k(\mathbf{p})\}$	**E, L, B+**	MCFG
Oncina [17]	Σ^*	$\Sigma^* \times \Delta^*$	$(p, lcp(\tau_L(p\Sigma^*)))^{-1}L$		OSST
This paper	$\Sigma^* \times \Delta^*$	$\Sigma^* \times \Delta^*$	$(u, v)^{-1}L$		FST

8 Distributional Lattice Grammars

Distributional Lattice Grammars [15] are an algorithmically more refined version of these approaches, which allow efficient learning and inference even when we have an exponentially large set of primitive elements Q. The starting point is the dual CFG approach; we start with a finite set F of contexts that includes the empty context (λ, λ). The primitive elements are not, however, these individual contexts but rather the set of all subsets of F. Thus rather than taking for a context $(l, r) \in F$ the set of strings $\{u \mid lur \in L\}$, we take as our primitive element f a subset of F, say $f = \{(l_1, r_1), \dots (l_k, r_k)\}$, and define

$$\mathbf{D}(f) = \{u \mid l_1 u r_1 \in L \wedge \dots \wedge l_k u r_k \in L\} \tag{4}$$

In other words, Q is just the power set of F; each element of Q is a subset of F. This clearly allows a much greater representational power: the sets of strings that we define thus correspond to finite intersections of the sets defined by individual contexts. This allows us to represent a larger class of languages. Consider for example the language $\{a^nb^n|n \geq 0\} \cup \{a^nb^{2n}|n \geq 0\}$; this language cannot be represented using sets that are defined by a single context, because the relevant sets of strings, such as $\{a^nb^n|n \geq 0\}$ are not defined by a single context. For example, the context (a, b) defines a set of strings that includes $\{a^nb^n|n \geq 0\}$ but also includes many other strings such as $\{abbb, aabbbbb \dots\}$. However if we allow our primitives sets to be defined by pairs of contexts, then the pair $(a, b), (aa, bb)$ will succesfully pick out, "triangulate" in a sense, the relevant set of strings. One approach to this, taken in [13] is simply to stipulate a maximum cardinality for the sets of contexts, and consider all sets of contexts of cardinality less than this. Considering this upper bound as fixed, the set of primitive elements becomes polynomial.

This avoids the problem rather than solves it; for natural language, it is essential to recognise that the syntactically and linguistically relevant sets of strings may require a large number of contexts to pick them out precisely.

An important insight of this approach is that there is a natural lattice structure that arises in these forms of learning. Since each primitive element p in Q defines a *set*, $\mathbf{D}(p)$ we can see that there will inevitably be a lattice structure generated by this set. Once we realise this, then it is natural to extend the set of primitive elements, by looking at the meet semi-lattice generated by the set $\{\mathbf{D}(p)|p \in Q\}$, and augmenting the inference system with conjunctive rules (those of type $\mathbf{C}$ above).

DLGs take this path and for computational reasons it turns out to be essential to add conjunctive rules. Given these rules, we can compute for every string w, the set of all sets that it must be a member of $Y(w) = \{\mathbf{D}(p)|w \in \mathbf{D}(p)\}$. The crucial observation is this: given that this is a lattice, rather than considering all of the exponentially many elements of this set of sets, we can sum it up in a single element; namely $\bigcap_{s \in Y(w)} s$. If w lies in all of the sets in $Y(w)$ then it must lie in their intersection. Since we have extended our set of primitive elements so it is a meet semi-lattice, (in fact a full lattice in the case of DLGs), then this intersection element will be in our set Q. Thus though w may be a member of very many sets defined by elements of Q, computationally we can consider just this unique one: the smallest set that we can prove it is in.

The addition of the conjunctive rules increases the power of the formalism to that of Conjunctive Grammars [18], or more precisely to a subclass of the languages definable by conjunctive grammars. This insight, though it has so far only been applied to the theory of CFG inference, is of more general application, and we think it can potentially be applied to all of the models discussed here.

9 MCFGs

Yoshinaka [19, 16] shows how we can extend this model to the inference of Multiple Context-Free Grammars [20]. We fix a natural number constant p and define the set Q to be a set of tuples of strings $(u_1, \ldots, u_m)$ where $1 \leq m \leq p$. X is then a set of generalised contexts which again are tuples of strings, this time of arity up to $p + 1$.

Given a tuple $\mathbf{w} = (u_1, \ldots, u_m)$ we can define the generalised distribution with respect to a language L to be the set $C^m_L(\mathbf{w}) = \{(v_1, \ldots, v_{m+1}) | v_1 u_1 v_2 \ldots v_m u_m v_{m+1} \in L\}$. The representational assumption of Yoshinaka's algorithm is thus

$$\mathbf{D}(\mathbf{w}) = \{\mathbf{u} | C^m_L(\mathbf{u}) \supseteq C^m_L(\mathbf{w})\} \tag{5}$$

Since the elements are no longer strings, but rather tuples of strings, the ways in which they can be combined are significantly more complex. Rather than one **B** rule, we have a whole family of such rules, each corresponding to a different combination operation.

It is an open question whether the class of DLGs is sufficiently expressive to represent the class of natural languages, or whether it will be necessary to move into the MCFL hierarchy. It might be that even if DLGs are sufficiently expressive, one might still want to use MCFGs because of the slightly richer notion of dependency that they allow, which might permit a more principled modeling of certain movement phenomena in natural languages.

10 Transductions

We now turn our attention to the study of transductions or bilanguages. We assume that we have two alphabets Σ and Δ which may or may not be disjoint, and rather than a language we are interested in *bilanguages* or *transductions* which are subsets of $\Sigma^* \times \Delta^*$; we will write an element of a bilanguage T as an ordered pair (u, v) where $u \in \Sigma^*$ and $v \in \Delta^*$. As defined like this, there is a symmetry but we will often be interested in the cases where L considered as a relation between Σ^* and Δ^* is a function. We say that a transduction T is *functional* if $(u, v), (u, w) \in T$ implies that $v = w$. We say that a transduction is total if for all $u \in \Sigma^*$ there is a $v \in \Delta^*$ such that $(u, v) \in T$.

If we are not interested in functional transductions, then this reduces to a special case of the learnability of multiple context free languages, subclasses of which can be learned directly using results already published [16]. However, as is demonstrated by the well-known OSTIA algorithm [17], if we assume that the data is functional, then we do not need to have membership queries or access to negative evidence, as the positive examples are restricted enough to learn the function.

We will consider now the case where we wish to infer a representation for a total function T. We will define the function $\tau_T : \Sigma^* \to \Delta^*$ as $\tau(u) = v$ where $(u, v) \in T$.

We will start by considering a basic model analogous to that of regular languages. We start by noting that our model above assumed only that the language was a subset of a monoid. Note that we clearly have a natural monoid structure over $\Sigma^* \times \Delta^*$, where $(u, v) \circ (u', v') = (uu', vv')$, and (λ, λ) is the identity. Thus

```
E ← ∅ ;
Q ← **InitQ** ;
X ← **InitX** ;
G ← **Make**(Q, X, E) ;
**while** $(u_i, v_i)$ *is a positive example* **do**
    $E \leftarrow E \cup \{(u_i, v_i)\}$ ;
    **for** $(u, v) \in E$ **do**
        **if** *not* $S \overset{*}{\Rightarrow}_G (u, v)$ **then**
            $Q \leftarrow Q \cup$ **IncreaseQ**$(E)$ ;
    $X \leftarrow X \cup$ **IncreaseX**$(w_i)$ ;
    $G =$ **Make**$(Q, X, E)$ ;
```

Algorithm 2. Generic meta-algorithm for learning functional transductions from positive data

we can now immediatelt lift our analysis to the case where $\mathbf{D}(p)$ is defined to be a subset of $\Sigma^* \times \Delta^*$.

We start by defining our sets of primitive elements Q to be a finite set of pairs $(u, v) \in \Sigma^* \times \Delta^*$, and assume further that $(\lambda, \lambda) \in Q$. We define for a given element $p = (u_p, v_p) \in Q$

$$\mathbf{D}((u_p, v_p)) = \{(u', v') | (u_p u', v_p v') \in T\} \tag{6}$$

We define the following rule schemas:

I $S \rightarrow [[(\lambda, \lambda)]]$ which is *a priori*
R $[[(u, v)]] \rightarrow (u', v')[[(uu', vv')]]$, also *a priori*
L0 $[[(u, v)]] \rightarrow (\lambda, \lambda)$ iff $(u, v) \in T$. This is certain.

The defeasible rule schema will be **E** rules. We will only use positive data here which is sufficient, since if we observe a pair (u, v) then we know that for all $v' \neq v$ that $(u, v') \notin T$, since T is a function. We will therefore have X being a set of pairs that are a subset of T. We will say that an equality rule $[[(u, v)]] \rightarrow [[(u', v')]]$ is *incorrect* with respect to X if there is a pair of elements of X of the form $(ux, vy), (u'x, w')$ such that $w' \neq v'y$. Note that these two elements of X need not be distinct, as we shall see below. If it is not incorrect w.r.t X then we say it is correct w.r.t. X. The **E** rule schema is thus:

E $[[(u, v)]] \rightarrow [[(u', v')]]$ iff it is correct w.r.t. X.

If the rule is correct, and $(ux, w), (u'x, w') \in X$, and suppose that $w = vy$, then $(x, y) \in \mathbf{D}(u, v)$ and therefore $(x, y) \in \mathbf{D}(u', v')$ and so $(u'x, v'y) \in T$ and so $w' = v'y$ since it is functional, and therefore it will be correct w.r.t any X On the other hand, if a rule is incorrect, then there must be some $(x, y) \in \mathbf{D}(u, v) \setminus \mathbf{D}(u', v')$, or $\mathbf{D}(u', v') \setminus \mathbf{D}(u, v)$. If we assume w.l.o.g. the first, then we know that $\tau(ux) = vy$. Let $w = vy$ and $w' = \tau(u'x)$; w' cannot be equal to $v'y$ since this would mean that $(x, y) \in \mathbf{D}(u', v')$ which would be a contradiction. Therefore if X contains the two pairs $(ux, \tau(ux)), (u'x, \tau(u'x))$ then this rule will be incorrect w.r.t. X.

We are only interested in those cases where $(u, v)^{-1}T$ is non-empty. This set could be empty, if for example all of the strings that start with u are mapped to strings that start with a, and v starts with b. The algorithm we consider will only add (u, v) when we have observed them as a prefix of a string, and so we will assume in what follows that there is always at least one element of $(u, v)^{-1}T$. There are is a special cases that we should note: when $u = u'$ and $v \neq v'$ clearly these pairs will not be congruent. In this case, let (x, y) be some element of $(u, v)^{-1}T$; then clearly (ux, vy) is a suitable element of X to show that these pairs are not congruent.

We have implicitly defined **Make**; we now define the other subroutines. **IncreaseX** just returns the current data E, and **IncreaseQ** will return the set of all prefixes of E. We say that (u, v) is a prefix of (u', v') if there is some (x, y) such that $(u, v) \circ (x, y) = (u', v')$. The initialisation functions just set X to the empty set and Q to $\{(\lambda, \lambda)\}$. There is one detail we neglect in this informal presentation: we also need to deal with the **L0** rules. Since we are not using an oracle, we set them when we observe the relevant pair (u, v) in the data.

Therefore Algorithm 2 will learn the class of all such transductions with a finite kernel. Let us pause and consider the class of transductions that have a finite kernel; these will clearly be a class of rational functions. These clearly include all subsequential functions, which are those where the underlying automaton is deterministic, and there is a final output function σ. The role of $\sigma(q)$ is played by λ-transitions leading to accepting states. However this clearly also includes non-deterministic transductions. We consider now the classic example of such a transduction (c.f. [21]).

Example 1. Suppose we have $\Sigma = \{a\}, \Delta = \{b, c\}$, and $T = \{(a^{2n}, b^{2n}) | n \geq 0\} \cup \{(a^{2n+1}, c^{2n+1}) | n \geq 0\}$. This is clearly a total function; a^n is mapped to b^n if n is even and to c^n if n is odd.

A kernel for this transduction is the following set Q:

$$\{(\lambda, \lambda), (a, b), (aa, bb), (aaa, bbb), (a, c), (aa, cc), (aaa, ccc)\}$$

We can easily verify that $(a, b) \equiv_T (aaa, bbb)$ and that $(a, c) \equiv_T (aaa, ccc)$. It is easy to convert this to a finite state transducer (FST). Each element of Q corresponds to a state; accepting states are those with a rule of type **L0**. The initial state is $[[(\lambda, \lambda)]]$. Rules of type $\mathbf{R}$ such as $[[(u, v)]] \to (u', v')[[(uu', vv')]]$ are written as a transition $[[(u, v)]] \overset{u':v'}{\to} [[(uu', vv')]]$. Figure 1 shows a minimal set of primitive elements and transitions that define this transduction. The actual output from the algorithm would contain many more states and transitions, but would nonetheless not over-generate.

The relation to the OSTIA algorithm is not entirely clear. The OSTIA algorithm infers a class of subsequential transducers; these are deterministic on the input string: thus we can define an equivalence relation of finite index on the set of pairs. If we define the longest common prefix of a set of strings to be lcp, the

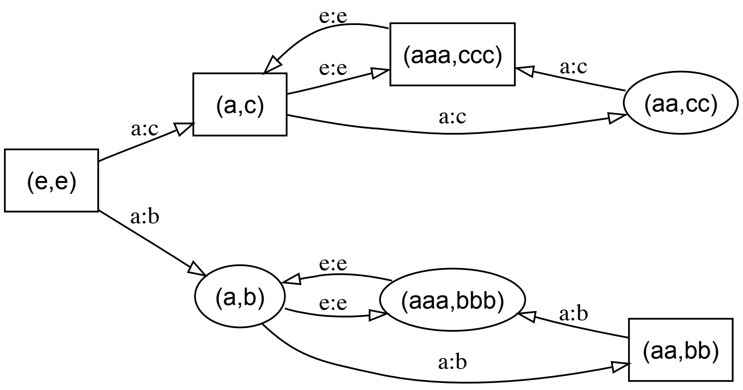


Fig. 1. State diagram for $T = \{(a^{2n}, b^{2n})|n \geq 0\} \cup \{(a^{2n+1}, c^{2n+1})|n \geq 0\}$. The "e" stands for the empty string λ; accepting states are drawn as rectangles. We write this as a finite state transducer rather than a rewriting system.

primitives of the OSTIA algorithm are of the form $(u, lcp(\tau(u\Sigma^*)))$, extending τ to sets of strings in the standard way. As can be seen from Figure 1, the output of the algorithm here is not deterministic on the input string. It appears that the class of transductions that can be learned includes all rational functions, but this must remain a conjecture at a moment.

11 Discussion

We have presented a common framework which allows us to see many different models and algorithms, at a suitable level of abstraction, as instances of the same meta-algorithm. In Table 1 we lay out, somewhat crudely the range of representational assumptions of the models that we have looked at in this paper. We have presented a meta-algorithm quite generally. As a result the specific algorithms are significantly less efficient that they could be. Compare, for example, the elegant algorithmics of the LSTAR or OSTIA algorithms with the very blunt approach taken in this paper. Nonetheless, this rather abstract presentation has allowed us to see that many classic and recent algorithms for GI are variants of the same algorithm. Using these methods allows us to see the range of possible new algorithms and GI techniques that result from combinations of different representational assumptions and sets of rules.

The example of a transduction learning algorithm is meant to show the advantages of this approach – applying this to the problems of learning transductions or functions immediately gives us a new and powerful algorithm for learning regular functions that extends previous results. There is also a natural extension to context-free transductions that we will present elsewhere.

The resulting algorithms are polynomial in the sense that each iteration requires only polynomial time. This is, perhaps, not strict enough a criterion on its own, but in some cases (e.g. [12]) we can get a result under the MAT model.

The general approach we advocate is ultimately very simple – a decision about what each representational element should mean; given this, we can define a set of valid inferences; invalid inferences can be removed based on testing an increasingly large set of experiments. The overall effect is a large and growing family of efficient algorithms for many classic problems in grammatical inference.

Acknowledgments

I am very grateful to Shalom Lappin and Ryo Yoshinaka for comments and corrections on a draft of this paper; any remaining errors are of course my own.

References

[1] Gold, E.M.: Language identification in the limit. Information and control 10(5), 447–474 (1967)

[2] Kearns, M., Valiant, G.: Cryptographic limitations on learning boolean formulae and finite automata. JACM 41(1), 67–95 (1994)

[3] Angluin, D., Kharitonov, M.: When won't membership queries help? J. Comput. Syst. Sci. 50, 336–355 (1995)

[4] Clark, A.: Three learnable models for the description of language. In: Dediu, A.-H., Fernau, H., Martín-Vide, C. (eds.) Language and Automata Theory and Applications. LNCS, vol. 6031, pp. 16–31. Springer, Heidelberg (2010)

[5] Pereira, F., Warren, D.: Parsing as deduction. In: Proceedings of the 21st annual meeting of the Association for Computational Linguistics, Association for Computational Linguistics, pp. 137–144 (1983)

[6] Yoshinaka, R.: Identification in the limit of k, l-substitutable context-free languages. In: Proceedings of the 9th International colloquium on Grammatical Inference, pp. 266–279. Springer, Heidelberg (2008)

[7] Angluin, D.: Learning regular sets from queries and counterexamples. Information and Computation 75(2), 87–106 (1987)

[8] Dupont, P., Miclet, L., Vidal, E.: What is the search space of the regular inference? In: Carrasco, R.C., Oncina, J. (eds.) ICGI 1994. LNCS, vol. 862, pp. 25–37. Springer, Heidelberg (1994)

[9] Sempere, J., Garcia, P.: A Characterization of Even Linear Languages and its Application to the Learning Problem. In: Proceedings of the Second International Colloquium on Grammatical Inference and Applications, pp. 38–44. Springer, Heidelberg (1994)

[10] Clark, A., Eyraud, R.: Polynomial identification in the limit of substitutable context-free languages. Journal of Machine Learning Research 8, 1725–1745 (2007)

[11] Clark, A.: PAC-learning unambiguous NTS languages. In: Proceedings of the 8th International Colloquium on Grammatical Inference (ICGI), 59–71 (2006)

[12] Clark, A.: Distributional learning of some context-free languages with a minimally adequate teacher. In: Proceedings of the ICGI, Valencia, Spain (September 2010)

[13] Clark, A.: Learning context free grammars with the syntactic concept lattice. In: Proceedings of the ICGI, Valencia, Spain (September 2010)

[14] Clark, A., Eyraud, R., Habrard, A.: A polynomial algorithm for the inference of context free languages. In: Proceedings of the International Colloquium on Grammatical Inference, September 2008, pp. 29–42. Springer, Heidelberg (2008)
[15] Clark, A.: A learnable representation for syntax using residuated lattices. In: Proceedings of the 14th Conference on Formal Grammar, Bordeaux, France (2009)
[16] Yoshinaka, R.: Polynomial-time identification of multiple context-free languages from positive data and membership queries. In: Proceedings of the International Colloquium on Grammatical Inference (2010)
[17] Oncina, J., García, P., Vidal, E.: Learning subsequential transducers for pattern recognition interpretation tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence 15, 448–458 (1993)
[18] Okhotin, A.: Conjunctive grammars. Journal of Automata, Languages and Combinatorics 6(4), 519–535 (2001)
[19] Yoshinaka, R.: Learning mildly context-sensitive languages with multidimensional substitutability from positive data. In: Gavaldà, R., Lugosi, G., Zeugmann, T., Zilles, S. (eds.) ALT 2009. LNCS, vol. 5809, pp. 278–292. Springer, Heidelberg (2009)
[20] Seki, H., Matsumura, T., Fujii, M., Kasami, T.: On multiple context-free grammars. Theoretical Computer Science 88(2), 229 (1991)
[21] Mohri, M.: Finite-state transducers in language and speech processing. Computational Linguistics 23(2), 269–311 (1997)

The Blessing and the Curse of the Multiplicative Updates

Manfred K. Warmuth[⋆]

Computer Science Department
University of California, Santa Cruz
CA 95064, U.S.A.
manfred@cse.ucsc.edu

Abstract. Multiplicative updates multiply the parameters by nonnegative factors. These updates are motivated by a Maximum Entropy Principle and they are prevalent in evolutionary processes where the parameters are for example concentrations of species and the factors are survival rates. The simplest such update is Bayes rule and we give an in vitro selection algorithm for RNA strands that implements this rule in the test tube where each RNA strand represents a different model. In one liter of the RNA "soup" there are approximately 10^{20} different strands and therefore this is a rather high-dimensional implementation of Bayes rule. We investigate multiplicative updates for the purpose of learning online while processing a stream of examples. The "blessing" of these updates is that they learn very fast because the good parameters grow exponentially. However their "curse" is that they learn too fast and wipe out parameters too quickly. We describe a number of methods developed in the realm of online learning that ameliorate the curse of these updates. The methods make the algorithm robust against data that changes over time and prevent the currently good parameters from taking over. We also discuss how the curse is circumvented by nature. Some of nature's methods parallel the ones developed in Machine Learning, but nature also has some additional tricks.

[⋆] Supported by NSF grant IIS-0917397.

© Springer-Verlag Berlin Heidelberg 2010

Discovery of Abstract Concepts by a Robot

Ivan Bratko

University of Ljubljana, Faculty of Computer and Information Science
Tržaška 25, 1000 Ljubljana, Slovenia
ivan.bratko@fri.uni-lj.si

Abstract. This paper reviews experiments with an approach to discovery through robot's experimentation in its environment. In addition to discovering laws that enable predictions, we are particularly interested in the mechanisms that enable the discovery of abstract concepts that are not explicitly observable in the measured data, such as the notions of a tool or stability. The approach is based on the use of Inductive Logic Programming. Examples of actually discovered abstract concepts in the experiments include the concepts of a movable object, an obstacle and a tool.

Keywords: Autonomous discovery, robot learning, discovery of abstract concepts, inductive logic programming.

© Springer-Verlag Berlin Heidelberg 2010

Contrast Pattern Mining and Its Application for Building Robust Classifiers

Kotagiri Ramamohanarao

Department of Computer Science and Software Engineering
The University of Melbourne
Kotagiri@unimelb.edu.au

Abstract. The ability to distinguish, differentiate and contrast between different data sets is a key objective in data mining. Such ability can assist domain experts to understand their data and can help in building classification models. This presentation will introduce the techniques for contrasting data sets. It will also focus on some important real world applications that illustrate how contrast patterns can be applied effectively for building robust classifiers.

© Springer-Verlag Berlin Heidelberg 2010

Optimal Online Prediction in Adversarial Environments

Peter L. Bartlett

Computer Science Division and Department of Statistics,
University of California at Berkeley, Berkeley CA 94720, USA
bartlett@cs.berkeley.edu

In many prediction problems, including those that arise in computer security and computational finance, the process generating the data is best modelled as an adversary with whom the predictor competes. Even decision problems that are not inherently adversarial can be usefully modeled in this way, since the assumptions are sufficiently weak that effective prediction strategies for adversarial settings are very widely applicable.

The first part of the talk is concerned with the regret of an optimal strategy for a general online repeated decision problem: At round t, the strategy chooses an action (possibly random) a_t from a set $\mathcal{A}$, then the world reveals a function ℓ_t from a set $\mathcal{L}$, and the strategy incurs a loss $\mathbf{E}\ell_t(a_t)$. The aim of the strategy is to ensure that the regret, that is, $\mathbf{E}\sum_t \ell_t(a_t) - \inf_{a\in\mathcal{A}} \sum_t \ell_t(a)$ is small. The results we present [1] are closely related to finite sample analyses of prediction strategies for probabilistic settings, where the data are chosen iid from an unknown probability distribution. In particular, we relate the optimal regret to a measure of complexity of the comparison class that is a generalization of the Rademacher averages that have been studied in the iid setting.

Many learning problems can be cast as online convex optimization, a special case of online repeated decision problems in which the action set $\mathcal{A}$ and the loss functions ℓ are convex. The second part of the talk considers optimal strategies for online convex optimization [2, 3]. We present the explicit minimax strategy for several games of this kind, under a variety of constraints on the convexity of the loss functions and the action set $\mathcal{A}$. The key factor is the convexity of the loss functions: curved loss functions make the decision problem easier. We also demonstrate a strategy that can adapt to the difficulty of the game, that is, the strength of the convexity of the loss functions, achieving almost the same regret that would be possible if the strategy had known this in advance.

References

[1] Abernethy, J., Agarwal, A., Bartlett, P.L., Rakhlin, A.: A stochastic view of optimal regret through minimax duality. arXiv:0903.5328v1 [cs.LG] (2009)
[2] Abernethy, J., Bartlett, P.L., Rakhlin, A., Tewari, A.: Optimal strategies and minimax lower bounds for online convex games. UC Berkeley EECS Technical Report EECS-2008-19 (2008)
[3] Bartlett, P.L., Hazan, E., Rakhlin, A.: Adaptive online gradient descent. UC Berkeley EECS Technical Report EECS-2007-82 (2007)

© Springer-Verlag Berlin Heidelberg 2010

An Algorithm for Iterative Selection of Blocks of Features

Pierre Alquier

[1] LPMA (University Paris 7)
175, rue du Chevaleret
75013 Paris France
alquier@math.jussieu.fr
http://alquier.ensae.net/
[2] CREST (ENSAE)

Abstract. We focus on the problem of linear regression estimation in high dimension, when the parameter β is "sparse" (most of its coordinates are 0) and "blocky" (β_i and β_{i+1} are likely to be equal). Recently, some authors defined estimators taking into account this information, such as the Fused-LASSO [19] or the S-LASSO [10] among others. However, there are no theoretical results about the obtained estimators in the general design matrix case. Here, we propose an alternative point of view, based on the Iterative Feature Selection method [1]. We propose an iterative algorithm that takes into account the fact that β is sparse *and* blocky, with no prior knowledge on the position of the blocks. Moreover, we give a theoretical result that ensures that every step of our algorithm actually improves the statistical performance of the obtained estimator. We provide some simulations, where our method outperforms LASSO-type methods in the cases where the parameter is sparse and blocky. Moreover, we give an application to real data (CGH arrays), that shows that our estimator can be used on large datasets.

Keywords: Feature Selection, Sparsity, Linear Regression, Grouped Variables, ArrayCGH.

1 Introduction

1.1 Setting of the Problem

We assume that we are in the gaussian linear regression setting

$$Y = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix} \sim \mathcal{N}\left(X\beta, \begin{pmatrix} \sigma^2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma^2 \end{pmatrix} \right) \tag{1}$$

where X is some (deterministic) real-valued matrix $n \times p$ and $\beta = (\beta_1, \dots, \beta_p)' \in \mathbb{R}^p$, with possibly $p > n$ (we use the convention that any $u \in \mathbb{R}^p$ is a column vector and we use the notation u' for its transpose). Let $X_1, \dots, X_p$ be the columns

© Springer-Verlag Berlin Heidelberg 2010

of X, and let $\mathbf{P}$ denote the probability distribution of Y given by Equation 1 (for the sake of simplicity, we assume that the data are normalized in such a way that $X_j'X_j = n$). Our purpose is to estimate β, based on the observation of both X and Y. We use as a criterion the quadratic loss, for any $b \in \mathbb{R}^p$, we put

$$L(b) = \|X(b-\beta)\|_2^2 .$$

It is well known that, as soon as $p > n$, it is not possible to build an estimator with a small loss $L(\cdot)$ - unless some additional condition is satisfied. For example, one may assume that β is sparse, which means that the number of non-zero coefficients in β (usually refered as $\|\beta\|_0$) is small. If $\|\beta\|_0 \ll n$, the LASSO estimator [18], the Dantzig selector [6] or the nonnegative garrote [4], among others, may achieve good performance: see for example the simulations in [18], and the theoretical results in [5] and [3]. Also note that [21] provides a nice survey of the theoretical results for the LASSO and the conditions needed to prove these theoretical results.

Here, we focus on the case where β is sparse *and blocky*. By this, we mean that the non-zero coefficients of β are "grouped" in blocks where they have the same value. For example, a typical parameter β could be

$$\beta = (0,0,0,5,5,5,0,0,0,0,0,1,1,1,2,2,0,0)' .$$

An example of application is given in genomics, where models like the one in (1) may appear. The observation Y_i represents a characteristic of a tumor and depends on some lesions appearing on the chromosomes of cancer cells $X_{j,i}$, the index j representing the localization on the chromosome. Since a lesion generally affects a whole part of a chromosome (duplication or deletion of a region), one may expect that two consecutive parameters β_j and β_{j+1} have some strong relationship. For example, [15] use such an assumption - the main difference is that [15] considers the context of classification, and not of regression. They use a Support Vector Machine with a penalization of the type

$$\lambda \sum_{j=1}^{p} |\beta_j| + \mu \sum_{j=2}^{p} |\beta_j - \beta_{j-1}|,$$

called Fused-SVM in the paper. Using such a penalisation, we expect to find a sparse and blocky solution: the first term ensures sparsity when λ is large, the second term ensures that for most j, $\beta_j = \beta_{j-1}$ and so there are blocks of similar values in the solution. Such a penalization is also used in [12] for genomic applications, with a logistic regression model.

Some estimators have been proposed for the regression case. Let us mention the Fused-LASSO [19], say $\tilde{\beta}^F_{s,t}$, given by

$$\min_b \left\{ \|Y - Xb\|_2^2 + 2ns \sum_{j=1}^{p} |\beta_j| + 2nt \sum_{j=2}^{p} |\beta_j - \beta_{j-1}| \right\}$$

with $s, t > 0$, or the S-LASSO estimator, say $\tilde{\beta}^S_{s,t}$, by

$$\min_b \left\{ \|Y - Xb\|_2^2 + 2ns \sum_{j=1}^{p} |\beta_j| + 2nt \sum_{j=2}^{p} (\beta_j - \beta_{j-1})^2 \right\}$$

proposed in [9] and studied in [10] (the S-LASSO does not actually lead to blocky solutions, but to smooth solutions: most $\beta_j - \beta_{j-1}$ are small, but not necessarily 0). See also the structured feature selection in [17]. These estimators can be approximated in practice even for large p: for example the Pathwise Coordinate Optimization algorithm [8] can be used for the S-LASSO, and is used for the Fused-LASSO in [8] when $X = I_n$ (from now I_k will denote the indentity matrix of size k). The Fused-LASSO can be approximated in the general case using for example the algorithms in [11] and [20]. However, note that no general theoretical results were provided in order to estimate a sparse blockwise β. Hebiri [9] provides good guarantees for the S-LASSO under the sparsity assumption but does not take any advantage of the smooth aspect of β, if any. Rinaldo [16] gives theoretical guarantees for the Fused-LASSO that can be applied only in the case where $X = I_n$, and so $n = p$.

Finally, the Group-LASSO introduced by [22] has very interesting practical performance, as well as good theoretical properties studied in [7]. See also [23]. However, this procedure requires the prior knowledge of the location of the groups. The S-LASSO and Fused-LASSO do not require such a prior, and in this paper, we are interested in the case where we do not have this knowledge. Our setting is then:

– we know that most of the $\beta_j = 0$ but we do not know the j's such that $\beta_j \neq 0$;
– we know that most of the $\beta_j = \beta_{j+1}$ but we do not know the j's such that $\beta_j \neq \beta_{j+1}$.

1.2 Overview of the Paper

In this paper, we propose an algorithm to estimate a sparse and blockwise β in the model given by (1), without prior knowledge on the location of the groups. This algorithm is an iterative algorithm. It starts from the initial value $\hat{\beta}^{(0)} = (0, \ldots, 0)$. Then, at each step m, we compute $\hat{\beta}^{(m+1)}$ from $\hat{\beta}^{(m)}$ in the following way: a particular coordinate or group of consecutive coordinates is selected, and then updated. The way to update this coordinate, or group of coordinates, is described in Section 2. The way to select the coordinate, or the group, is postponed to Section 3.

Actually, in Section 3, we give the following result: with large probability, at each step, whatever the choice of the coordinate (or group) to be updated, $L(\hat{\beta}^{(m+1)}) \leq L(\hat{\beta}^{(m)})$. This is Theorem 1. A refinement of this result, Theorem 2, allows to choose the particular coordinate (or group) that ensures the largest possible decrease from $L(\hat{\beta}^{(m)})$ to $L(\hat{\beta}^{(m+1)})$.

We then provide a simulation study in Section 4 with a comparison to the S-LASSO and the Fused-LASSO estimators. Our method outperforms the LASSO-type estimators when the parameter is sparse and blocky. We also provide an application to real data (arrayCGH) in Section 5.

Finally, the proof of Theorems 1 and 2 are given in Section 6.

2 Construction of Our Estimator

We now describe our algorithm. It is an iterative algorithm, that starts from $\hat{\beta}^{(0)} = (0, \ldots, 0)$. At each step, we are going to compute $\hat{\beta}^{(m+1)}$ from $\hat{\beta}^{(m)}$. Let us define the soft thresholding function

$$\forall x \in \mathbb{R}, \forall u \geq 0, \quad \gamma(x,u) = \text{sign}(x)(|x| - u)_+$$

where $\text{sign}(x)$ is the sign of x. For any given $j \in \{1, \ldots, p\}$ and $k \in \{1, ..., p-j+1\}$ let us define $\mathbf{1}_{j,k} \in \mathbb{R}^p$ by $(\mathbf{1}_{j,k})_i = 1$ if $i \in \{j, \ldots, j+k-1\}$, and $(\mathbf{1}_{j,k})_i = 0$ otherwise (in other words, $\mathbf{1}_{j,k} \in \mathbb{R}^p$ is the pattern of the group of variables $(j, j+1, \ldots, j+k-1)$, of length k).

At each step, several possible moves are considered in our algorithm: a possible move is to update a coordinate or a group of coordinates of maximal size K, for a given $K \in \{1, \ldots, p\}$. Let us now describe the move for a chosen coordinate or group of coordinates. The way to choose what group of coordinates is to be updated is discussed in the next section, in view of some theoretical results. Let us choose $s > 0$ (the value of s is also discussed in the next section).

Parameter update. Let us assume that we are going to update the group of coordinates $(j, j+1, \ldots, j+k-1)$ (a single coordinate is a particular case with $k = 1$). We define $\hat{\beta}^{(m+1)}$ by

$$\begin{cases} \hat{\beta}_j^{(m+1)} = \hat{\beta}_j^{(m)} + \gamma\left(\frac{(Y - X\hat{\beta}^{(m)})' \sum_{h=j}^{j+k-1} X_h}{\mathbf{1}'_{j,k} X' X \mathbf{1}_{j,k}}, s\right) \\ \vdots \\ \hat{\beta}_{j+k-1}^{(m+1)} = \hat{\beta}_{j+k-1}^{(m)} + \gamma\left(\frac{(Y - X\hat{\beta}^{(m)})' \sum_{h=j}^{j+k-1} X_h}{\mathbf{1}'_{j,k} X' X \mathbf{1}_{j,k}}, s\right) \end{cases} \tag{2}$$

and $\hat{\beta}_\ell^{(m+1)} = \hat{\beta}_\ell^{(m)}$ for any $\ell \notin \{j, \ldots, j+k-1\}$.

Observe that in the case where $k = 1$, this expression is very natural:

$$\hat{\beta}_j^{(m+1)} = \hat{\beta}_j^{(m)} + \gamma\left(\frac{(Y - X\hat{\beta}^{(m)})' X_j}{n}, s\right)$$

and $\hat{\beta}_\ell^{(m+1)} = \hat{\beta}_\ell^{(m)}$ for any $\ell \neq j$. If we take $K = 1$, so we never consider groups of coordinates, we obtain the Iterative Feature Selection algorithm studied in [1].

3 Main Result and Comments

We now give some properties satisfied by any algorithm using only the kind of moves described in the previous section. These results will also provide us a rule to choose the group of coordinates that have to be updated at step m, to calibrate the parameters s and K and, finally, to decide at which step m to stop our iterations.

Note that the proof of the following result is given at the end of the paper, in Section 6, page 44.

Theorem 1. *Let us assume that, as described previously, we start from* $\hat{\beta}^{(0)} = (0, \ldots, 0)$*, that at each step m we update one group of coordinates following Equation 2 (the choice of the group* **may** *be data-driven). We assume that we stop after M steps. We have*

$$\mathbf{P}\left[L(\hat{\beta}^{(M)}) \leq L(\hat{\beta}^{(M-1)}) \leq \cdots \leq L(\hat{\beta}^{(0)})\right] \geq 1 - Kpe^{-\frac{ns^2}{2\sigma^2}}.$$

In a way, this result states that every strategy using only the kind of moves described in Section 2 cannot result in overfitting: we are "almost certain" to get closer to our (unknown) objective β at each step. We can quantify this "almost certain": it means "with probability at least $1 - \varepsilon$", if we choose s such that $Kp\exp(-ns^2/(2\sigma^2)) \leq \varepsilon$. For example,

$$s = \sqrt{\frac{2\sigma^2 \log(\frac{pK}{\varepsilon})}{n}}$$

gives a confidence level $1 - \varepsilon$. Note that it is very similar to the theoretical value for the regularization parameter in the LASSO proposed in [3]. This link will be made clearer in the conclusion of the paper.

Note that if we allow $K = p$ (to test every group of size 1 to p), this will not make a big difference because the threshold s will become

$$s = \sqrt{\frac{2\sigma^2 \log(\frac{p^2}{\varepsilon})}{n}} \leq \sqrt{\frac{4\sigma^2 \log(\frac{p}{\varepsilon})}{n}}.$$

However, when p is large, it could be more convenient to choose a smaller K for algorithmic reasons. Let us stress this conclusion for K: $K = p$ is probably almost optimal in theory, but if p is large, we may want to choose a smaller K to keep the computation time small (we will shortly discuss this point in Subsection 3.1).

Also, note that this choice for s is not necessarily the best choice in practice: it requires the knowledge of σ^2, and it is usually too large (see the experiments in [2] for the case $K = 1$). Moreover it is not data-driven. Cross-validation will provide better results in practice.

Finally, we still do not know "how much closer to β" we can move at every step, and we still have no idea of how to choose if we are going to update $1, 2 \ldots$ or K coordinates, and wich coordinates we are going to update - we just know that in some sense, every choice is allowed. We now make some propositions.

Sequential strategy. Just sequentially update $j = 1$, $j = 2$, ..., $j = p$ with 1 coordinates moves, then $(j, j+1) = (1, 2)$, ..., $(j, j+1) = (p-1, p)$ with 2 coordinates moves, then $(j, j+1, j+2) = (1, 2, 3)$ ... with 3 coordinates moves, ... up to K coordinates moves, and start again. This strategy (with $K = 2$) is in some way similar to the one proposed for example in [8] for an algorithm computing the Fused-LASSO estimate.

Now, a finer version of Theorem 1 will give us an opportunity to make a data-driven choice of the group of coordinates to update at each step.

Theorem 2. *In the same setting as for Theorem 1,*

$$\mathbf{P}\left[\forall m \in \{1, \ldots, M\} : L(\hat{\beta}^{(m)}) \leq L(\hat{\beta}^{(m-1)}) - \|X(\hat{\beta}^{(m)} - \hat{\beta}^{(m-1)})\|_2^2\right] \geq 1 - Kpe^{-\frac{ns^2}{2\sigma^2}}.$$

Note that the proof of Theorem 2 will be included in the proof of Theorem 1.

"Best single move" strategy. We propose, at each step, to choose the move that maximises $\|X(\hat{\beta}^{(m)} - \hat{\beta}^{(m-1)})\|_2^2$. Moreover, this allows to define a stopping criterion: we stop when all the possible moves give an improvement $\|X(\hat{\beta}^{(m)} - \hat{\beta}^{(m-1)})\|_2^2 < 1/n^2$ for example.

Note that in any case we do not claim that this strategy is the best possible one. It is possible that iterations gets stuck in some regions of $\mathbb{R}^p$ that are not the most interesting ones (in the same way that sequential optimization algorithms may be trapped in some local minimum). A way to avoid this risk is not clear yet. This is actually the object of an alternative estimator proposed in the conclusion of this paper: see Equation 5 page 47. This estimator will be the object of a future work. However, there are two things that we know for sure about this strategy:

1. Theorem 2 ensures that, after m steps, with probability at least $1 - Kpe^{-\frac{ns^2}{2\sigma^2}}$,
$$L(\hat{\beta}^{(m)}) \leq L(\hat{\beta}^{(0)}) - \sum_{k=1}^{m} \|X(\hat{\beta}^{(k)} - \hat{\beta}^{(k-1)})\|_2^2;$$
2. after m steps, only m coordinates, or groups of coordinates, have been updated so we know that the solution is sparse and blocky, at least when m is small. In our experimental study, very often, only a few moves are necessary, i.e. $\|X(\hat{\beta}^{(m)} - \hat{\beta}^{(m-1)})\|_2^2$ becomes very small after only a few steps.

3.1 A Note on the Computational Complexity of Our Method

Let us remark that we can easily upper bound the computational complexity of the "best single move" strategy with a limited number of steps M (like in Theorems 1 and 2).

First, we need to explore all the possible groups of variables, there are $p + (p-1) \cdots + (p-K+1) \leq Kp$ such groups. Then for each group we have to compute the quantities in (2), this costs roughly $\mathcal{O}(nk) = \mathcal{O}(nK)$ operations.

This means that the computation time is roughly $\mathcal{O}(npK^2M)$. So, of course, if we know that the blocks in β have a limited size, this is reasonable, but if we do not have such an information and we take $K = p$, we can have trouble in the case where p is large.

4 Simulations

4.1 Description of the Experiments

A toy example is proposed in the seminal paper of Tibshirani [18]. We slightly modify this example to have

$$\forall i \in \{1, \ldots, 50\}, \quad Y_i = \beta' X_i + \varepsilon_i$$

with $X_i \in \mathbb{R}^p$, $\beta \in \mathbb{R}^p$ and the ε_i are i. i. d. from a gaussian $\mathcal{N}(0, \sigma^2)$. The X_i's are i.i.d., and each X_i is drawn from a Gaussian distribution with mean $(0, \ldots, 0)'$ and with variance matrix:

$$\varSigma(\rho) = \left(\rho^{|i-j|}\right)_{\substack{i \in \{1,\ldots,p\} \\ j \in \{1,\ldots,p\}}}$$

for $\rho \in [0, 1[$. Note that Tibshirani's toy example is set with $p = 8$. Here we will consider $p > 8$.

We will use two values for β:

$$\beta^* = (5, 5, 5, 5, 5, 0, 0, 0, 0, 0, 2, 2, 2, 2, 0, \ldots, 0) \in \mathbb{R}^p,$$
$$\beta^{**} = (5, 4.5, 4, 3.5, 3, 2.5, 2, 1.5, 1, 0.5, 0, \ldots, 0) \in \mathbb{R}^p,$$

β^* is sparse and blocky while β^{**} is sparse and, in some way, smooth (a context that should be in favor of the S-LASSO estimator). We use two values for σ: 1 ("low noise") and 3 ("noisy case"); one value for ρ: $\rho = 0.5$, and finally two values for p: $p = 15 < n$ and $p = 100 > n$.

The S-LASSO and Fused-LASSO estimators are computed via the Pathwise Coordinate Optimization procedure described in [8]. Our procedure will be called in the experiments ISBF (Iterative Selection of Blocks of Features). It is used with the "best single move" strategy, and with $K = 10$.

We will finally use the parameters s and t in a grid:

$$s, t \in \mathcal{G} = \left\{ 10^{-i/8} \sqrt{\frac{\sigma^2 \log(p)}{n}}; \quad i = 0, \ldots, 20 \right\}. \tag{3}$$

4.2 Results

In a first time, we present in detail the results for one particular experiment. We then give an overview of the results in the whole set of experiments.

First, we focus on an experiment realized in the case $\beta = \beta^*$, $\sigma = 1$, $\rho = 0.5$ and $p = 15$. We define, for a given $s \in \mathcal{G}$, $L_{\mathrm{ISBF},s}$ as the loss obtained using the ISBF method with threshold value s, and

$$L_{\mathrm{F},s} = \arg\min_{t \in \mathcal{G}} L(\tilde{\beta}^F_{s,t})$$

the oracle for the Fused-LASSO with s fixed. We define in the same way $L_{\mathrm{S},s}$ for the S-LASSO. Figure 1 gives a plot of $L_{\mathrm{ISBF},s}$, $L_{\mathrm{F},s}$ and $L_{\mathrm{S},s}$ as a function of s.

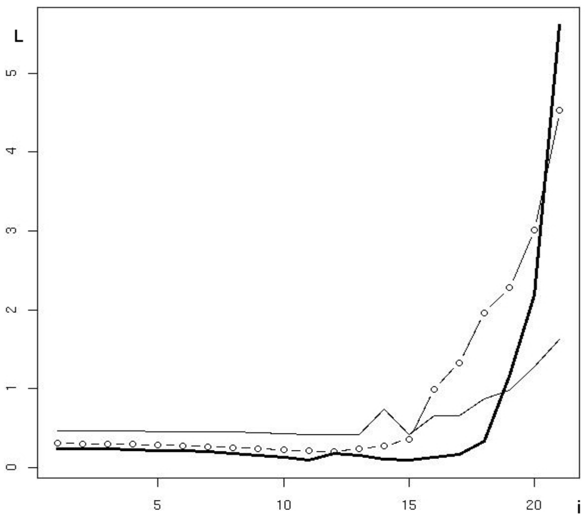


Fig. 1. The quantities $L_{\mathrm{ISBF},s}$ (thick line), $L_{\mathrm{F},s}$ (thin line) and $L_{\mathrm{S},s}$ (dotted line) as a function of s. The horizontal axis gives $i \in \{0, \ldots, 20\}$, the vertical axis is the value of the risk $L_{\mathrm{ISBF},s}$, $L_{\mathrm{F},s}$ and $L_{\mathrm{S},s}$ with $s = s(i)$ as defined in Equation 3.

Note that $L_{\mathrm{ISBF},s}$ is almost always under $L_{\mathrm{F},s}$ and $L_{\mathrm{S},s}$. So, for any reasonable s, the ISBF procedure reaches a better performance than the *oracle* of the Fused-LASSO and the S-LASSO – note that the oracles are not even available to the practitioners. In practice, we use some data-driven method to choose s for ISBF and (s, t) for the LASSO-type procedures, as cross-validation. Note that ISBF is more easy to deal with as it involves only one parameter to tune.

Now, let us have a look at the whole set of experiments (for each pair (σ, p) we run 20 experiments). For the Fused-LASSO and the S-LASSO, we report the performance of the oracle: namely, for each simulation, we define

$$L_{\mathrm{F}} = \arg\min_{(s,t) \in \mathcal{G}^2} L(\tilde{\beta}^F_{s,t})$$

the oracle loss for the Fused-LASSO, and we do the same to define the oracle loss of the S-LASSO, L_{S}, and for the ISBF, L_{ISBF}. The results for the estimation of β^* are reported in Table 1.

Table 1. Results for the estimation of β^* (sparse and blocky)

σ	p		L_F	L_S	L_{ISBF}
1	15	median	0.41	0.18	**0.10**
		mean	0.52	0.24	**0.14**
		s.d.	0.28	0.15	0.11
3	15	median	0.49	0.70	**0.41**
		mean	0.75	0.71	**0.46**
		s.d.	0.58	0.39	0.28
1	100	median	0.51	0.55	**0.35**
		mean	0.64	0.55	**0.38**
		s.d.	0.32	0.13	0.16
3	100	median	0.92	1.32	**0.75**
		mean	0.95	1.25	**0.77**
		s.d.	0.46	0.29	0.32

We can see that ISBF clearly outperforms the other methods on this particular set of experiments. The results for the estimation of β^{**} are reported in Table 2. Here, the S-LASSO is better.

Table 2. Results for the estimation of β^{**} (sparse and smooth)

σ	p		L_F	L_S	L_{ISBF}
1	15	median	0.51	**0.12**	0.21
1	15	mean	0.61	**0.13**	0.24
		s.d.	0.52	0.05	0.08
3	15	median	0.85	**0.36**	0.63
		mean	0.91	**0.39**	0.76
		s.d.	0.37	0.28	0.44
1	100	median	0.44	**0.31**	0.66
		mean	0.47	**0.35**	0.64
		s.d.	0.12	0.12	0.08
3	100	median	0.81	**0.70**	1.21
		mean	1.08	**0.71**	1.22
		s.d.	0.80	0.22	0.22

A conclusion to the very short experimental study is that the IFSF seems better in the case of a sparse and blocky parameter, while the S-LASSO is better in the case of a sparse and smooth parameter.

5 Application on CGH Data

We now present an application to the detection of amplification or deletion of DNA using CGH arrays. The data we have are partial CGH arrays (only for chromosome 17) of cancer cells. The model (for one patient) is the following:

$$Y_i = \beta_i + \varepsilon_i$$

for $1 \leq i \leq n$ with $n = 7728$, here i represents ordered positions of measurements on the chromosome. Remark that this is a particular case of Model (1) with $p = n$ and $X = I_n$ (this context is the one studied by Rinaldo [16]), here i represents ordered positions of measurements on the chromosome of the patient.

The data we use are on a logarithmic scale, so, $\beta_i = 0$ means no amplification or deletion of DNA at position i, $\beta_i < 0$ means deletion and $\beta_i > 0$ means amplification. We use ISBF to estimate β, the results are given in Figures 2 and 3 (for two different patients, with $K = 100$ and $s \simeq 0.01$). Note that there is work left to compare this method to the already available ones in this context. As an illustration we also give here the results of the estimation using the S-LASSO procedure with $s = t = 0.4$, see Figures 4 and 5. However, it is a good point to see that our method works in a reasonable time on a large dataset and gives reasonable results, probably more suited for interpretation by a practitioner.

Remark 1. *All simulations and experiments were performed with the R software [14]. The code is available from the author.*

6 Proof of Theorem 1

We now give the proof of our main result. First, we give some preliminary definitions and result.

Definition 1. *For the sake of simplicity let us put* $K_j = \min(K, p - j + 1)$.

Actually K_j is the maximal length for a group $\{j, j+1, \ldots, j+k-1\}$ constrained by $j + k - 1 \leq p$ and $k \leq K$.

Definition 2. *Let us put, for any* $s > 0$, $j \in \{1, \ldots, p\}$ *and* $k \in \{1, \ldots, K_j\}$*:*

$$R_s(j,k) = \left\{ b \in \mathbb{R}^p : \left| \frac{\sum_{h=j}^{j+k-1} X_h'(Y - Xb)}{\mathbf{1}_{j,k}' X' X \mathbf{1}_{j,k}} \right| \leq s \right\}$$

and, for any $s > 0$ *and* $b \in \mathbb{R}^p$ *let us define the event*

$$A_s(b) = \bigcap_{j=1}^{p} \bigcap_{k=1}^{K_j} \left\{ b \in R_s(j,k) \right\}.$$

Lemma 1. *For any* $s > 0$ *we have*

$$P\left[A_s(\beta)\right] \geq 1 - K\left(p - \frac{K-1}{2}\right) e^{-\frac{s^2 n}{2n\sigma^2}}.$$

Proof. First, let us note that (1) implies that

$$X'(Y - X\beta) \sim \mathcal{N}(0, \sigma^2 X'X)$$

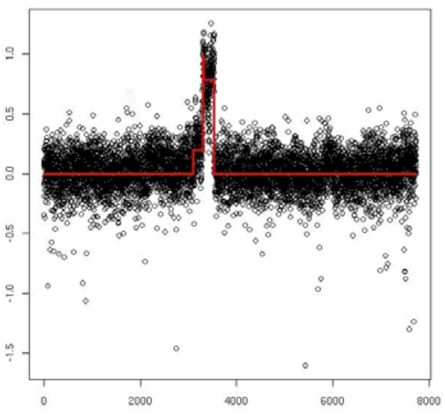


Fig. 2. The data Y (in black) and the estimated β (in red) using Iterative Selection of Blocks of Features, for Patient 1

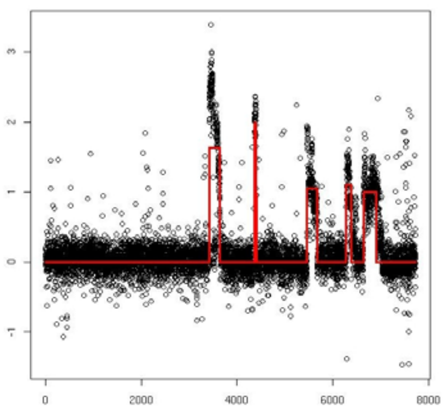


Fig. 3. The data Y (in black) and the estimated β (in red) using Iterative Selection of Blocks of Features, for Patient 2

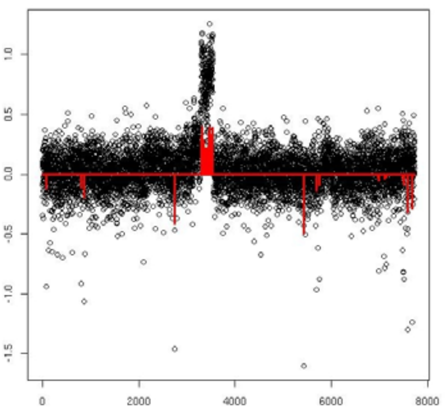


Fig. 4. The data Y (in black) and the estimated β (in red) using the S-LASSO estimator, for Patient 1

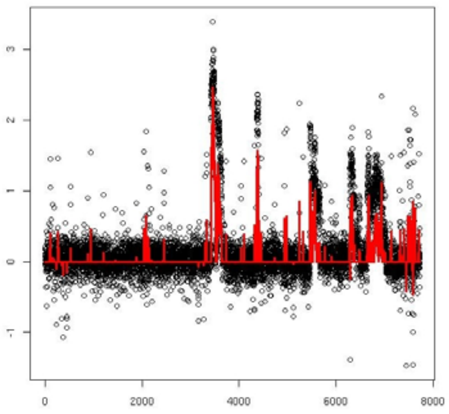


Fig. 5. The data Y (in black) and the estimated β (in red) using the S-LASSO estimator, for Patient 2

and so for any $j \in \{1, \dots, p\}$ and $k \in \{1, \dots, K_j\}$,

$$\sum_{h=j}^{j+k-1} X_h'(Y - X\beta) \sim \mathcal{N}(0, \sigma^2 \mathbf{1}_{j,k}' X' X \mathbf{1}_{j,k})$$

or

$$\frac{\sum_{h=j}^{j+k-1} X_h'(Y - X\beta)}{\mathbf{1}_{j,k}' X' X \mathbf{1}_{j,k}} \sim \mathcal{N}(0, \sigma^2).$$

This implies that, for any $j \in \{1,\ldots,p\}$, for any $k \in \{1,\ldots,K_j\}$ and for any $s > 0$,

$$P\left(\frac{|X'_h(Y - X\beta)|}{\mathbf{1}'_{j,k}X'X\mathbf{1}_{j,k}} > s\right) \leq e^{-\frac{s^2 n}{2\sigma^2}}$$

and, by a union bound argument over all $j \in \{1,\ldots,p\}$ and $k \in \{1,\ldots,K_j\}$,

$$P\left(\exists j \in \{1,\ldots,p\}, \exists k \in \{1,\ldots,K_j\} : \quad \frac{\left|\sum_{h=j}^{j+k-1} X'_h(Y - X\beta)\right|}{\mathbf{1}'_{j,k}X'X\mathbf{1}_{j,k}} > s\right)$$
$$\leq e^{-\frac{s^2 n}{2\sigma^2}} \sum_{\ell=0}^{K-1}(p-\ell) = \left[Kp - \frac{K(K-1)}{2}\right] e^{-\frac{s^2 n}{2\sigma^2}}.$$

This ends the proof. ■

Definition 3. *For any closed and convex set $\mathcal{C} \subset \mathbb{R}^p$ we define $\Pi_C(.)$ the orthogonal projection on $\mathcal{C}$ with respect to the norm induced by X:*

$$\Pi_C(b) = \arg\min_{p\in\mathcal{C}} \|X(b-p)\|_2^2.$$

We are now ready to give the proof of Theorem 1. The proof heavily uses the geometrical considerations given in [2].

Proof of Theorem 1. From now, let us fix $s > 0$ and assume that we are in the event $A_s(\beta)$ (according to Lemma 1, this is true with probability at least $1 - pK\exp(-ns^2/(2\sigma^2))$).

So, for any $j \in \{1,\ldots,p\}$, $k \in \{1,\ldots,K_j\}$, we have $\beta \in R_s(j,k)$. Moreover, note that $R_s(j,k)$ is convex and closed. Using classical convex analysis result (see [2] for example), we have, for any $b \in \mathbb{R}^p$,

$$\|X(\Pi_{R_s(j,k)}(b) - \beta)\|_2^2 \leq \|X(b-\beta)\|_2^2 - \|X(\Pi_{R_s(j,k)}(b) - b)\|_2^2, \qquad (4)$$

see Figure 6 for an illustration.

The last point of the proof will be to remark that, once that j and k are fixed, the $\hat{\beta}^{(m+1)}$ defined in the "1 coordinate move" satisfies

$$\hat{\beta}^{(m+1)} = \Pi_{R_s(j,k)}(\hat{\beta}^{(m)}).$$

To see this, remark that the projection $\Pi_{R_s(j,k)}(\hat{\beta}^{(m)})$ is defined by the program

$$\begin{cases} \min_{u\in\mathbb{R}^p} \|X(\hat{\beta}^{(m)} - u)\|_2^2 \\ s.t. \ \left|\frac{\sum_{h=j}^{j+k-1} X'_h(Y - Xu)}{\mathbf{1}'_{j,k}X'X\mathbf{1}_{j,k}}\right| \leq s, \end{cases}$$

of which the solution is given by (2). ■

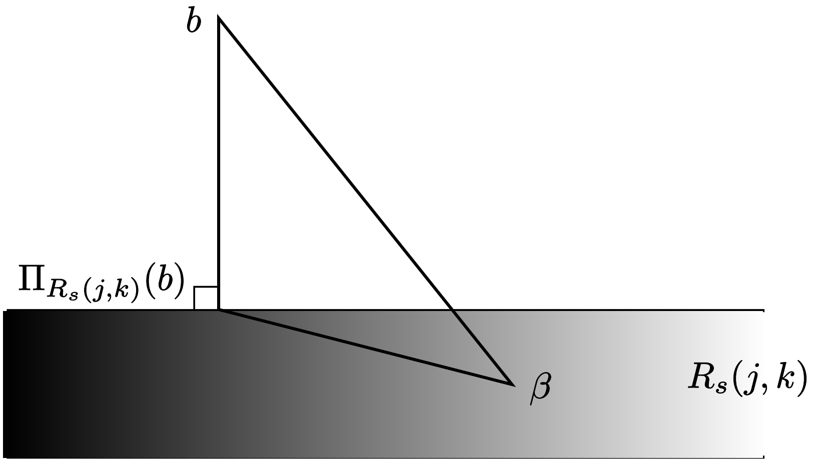


Fig. 6. Illustration of Equation 4

7 Conclusion and Perspectives

In this paper, we tried to give an estimator that takes advantage of the sparsity of β as well as of its blockwise aspect in the setting of regression in large dimension.

We proposed a very simple algorithm that performs well in practice, and give directions for the study of its theoretical performances.

In future works, we would like to give more precise theoretical results - in the spirit of [3] on the LASSO.

Eventually, we would like to point out some similarities between our estimator and other well-known estimators. In the proof of our main result, we presented our sequence of estimators $(\hat{\beta}^{(m)})_m$ as successive projections on various confidence regions of the form

$$R_s(j,k) = \left\{ b \in \mathbb{R}^p : \left| \frac{\sum_{h=j}^{j+k-1} X_h'(Y - Xb)}{\mathbf{1}_{j,k}' X' X \mathbf{1}_{j,k}} \right| \leq s \right\}.$$

We may wonder what happens if we project directly the first value $\hat{\beta}^{(0)} = 0$ into the intersections of these confidence regions. We obtain an estimator $\hat{\beta}(s,t)$ defined by the following program:

$$\hat{\beta}(s,t) = \begin{cases} \arg\min_{b \in \mathbb{R}^p} \|Xb\|_2^2 \\ s.t. \sup_{j,k} \left| \frac{\sum_{h=j}^{j+k-1} X_h'(Y-Xb)}{\mathbf{1}_{j,k}' X' X \mathbf{1}_{j,k}} \right| \leq s. \end{cases} \tag{5}$$

If we fix $K = 1$ (so we only work with constraints defined by single variables and not by groups of variables), we obtain

$$\begin{cases} \min_{b \in \mathbb{R}^p} \|Xb\|_2^2 \\ s.t. \quad \|\frac{1}{n} X'(Y - Xb)\|_\infty \leq s \end{cases}$$

that was proved to have as a solution the LASSO estimator given by

$$\min_{b\in\mathbb{R}^p}\left\{\|Y-Xb\|_2^2+2ns\sum_{j=1}^{p}|b_j|\right\},$$

see for example [13] for the proof, and [2] for a discussion of the geometric role of the constraint $\|(1/n)X'(Y-Xb)\|_\infty \leq s$. So, in some way, the estimator $\hat{\beta}(s,t)$ is really a variant of the LASSO, that tries to take into account the fact that β has blocks. The study of $\hat{\beta}(s,t)$ will be the object of future works.

Acknowledgments. We would like to thank Dr. Jean-Paul Feugeas (INSERM U944, Institut Universitaire d'Hématologie) who provided the CGH data with detailed explanations. We would also like to thank Dr. Mohamed Hebiri (ETH - Zürich) and Joseph Salmon (Université Paris 7) as well as the anonymous referees for useful comments. The project was funded by: ANR PARCIMONIE.

References

[1] Alquier, P.: Iterative feature selection in regression estimation. Annales de l'Institut Henri Poincaré, Probability and Statistics 44(1), 47–88 (2008)
[2] Alquier, P.: LASSO, iterative feature selection and the correlation selector: Oracle inequalities and numerical performances. Electron. J. Stat., 1129–1152 (2008)
[3] Bickel, P., Ritov, Y., Tsybakov, A.: Simultaneous analysis of LASSO and dantzig selector. The Annals of Statistics 37(4), 1705–1732 (2009)
[4] Breiman, L.: Better subset regression using the nonnegative garrote. Technometrics 37, 373–384 (1995)
[5] Bunea, F., Tsybakov, A., Wegkamp, M.: Sparsity oracle inequalities for the lasso. Electron. J. Stat. 1, 169–194 (2007)
[6] Candès, E., Tao, T.: The dantzig selector: statistical estimation when p is much larger than n. Ann. Statist. 35 (2007)
[7] Chesnau, C., Hebiri, M.: Some theoretical results on the grouped variables lasso. Mathematical Methods of Statistics 17(4), 317–326 (2008)
[8] Friedman, J., Hastie, T., Höfling, H., Tibshirani, R.: Pathwise coordinate optimization. Ann. Appl. Statist. 1(2), 302–332 (2007)
[9] Hebiri, M. Regularization with the smooth-LASSO procedure. Preprint LPMA, arXiv:0803.0668 (2008)
[10] Hebiri, M., Van de Geer, S.: The smooth-lasso and other $\ell_1+\ell_2$-penalized methods. arXiv:1003.4885 (2010)
[11] Hoefling, H.: A path algorithm for the fused LASSO signal approwimator. Preprint arXiv:0910.0526 (2009)
[12] Huang, J., Salim, A., Lei, K., O'Sullivan, K., Pawitan, Y.: Classification of array cgh data using smoothed logistic regression model. Statistics in Medicine 8(30), 3798–3810 (2009)
[13] Osborne, M., Presnell, B., Turlach, B.: On the LASSO and its dual. J. Comput. Graph. Statist. 9(2), 319–337 (2000)
[14] R Development Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2008) ISBN 3-900051-07-0

[15] Rapaport, F., Barillot, E., Vert, J.-P.: Classification of array-CGH data using fused SVM. Bioinformatics 24(13), 1375–1382 (2008)
[16] Rinaldo, A.: Properties and refinements of the fused LASSO. The Annals of Statistics 37(5B), 2922–2952 (2009)
[17] Slawski, M., zu Castell, W., and Tutz, G.: Feature selection guided by structural information. To appear in the Annals of Applied Statistics
[18] Tibshirani, R.: Regression shrinkage and selection via the LASSO. J. Roy. Statist. Soc. Ser. B 58(1), 267–288 (1996)
[19] Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., Knight, K.: Sparsity and smoothness via the fused lasso. JRSS-B 67(1), 91–108 (2005)
[20] Tibshirani, R.J., Taylor, J.: Regularization path for least squares problems with generalized ℓ_1 penalties (2009) (preprint)
[21] Van de Geer, S., Bühlmann, P.: On the conditions used to prove oracle results for the lasso. Electronic Journal of Statistics 3, 1360–1392 (2009)
[22] Yuan, M., Lin, Y.: Model selection and estimation in regression with grouped variables. JRSS-B 68(1), 49–67 (2006)
[23] Zhao, P., Rocha, G., Yu, B.: The composite absolute penalties for grouped and hierarchical variable selection. The Annals of Statistics 37(6A), 3468–3497 (2009)

Bayesian Active Learning Using Arbitrary Binary Valued Queries

Liu Yang[1], Steve Hanneke[2], and Jaime Carbonell[3]

[1] Machine Learning Department
Carnegie Mellon University
liuy@cs.cmu.edu
[2] Department of Statistics
Carnegie Mellon University
shanneke@stat.cmu.edu
[3] Language Technologies Institute
Carnegie Mellon University
jgc@cs.cmu.edu

Abstract. We explore a general Bayesian active learning setting, in which the learner can ask arbitrary yes/no questions. We derive upper and lower bounds on the expected number of queries required to achieve a specified expected risk.

Keywords: Active Learning, Bayesian Learning, Sample Complexity, Information Theory.

1 Introduction

In this work, we study the fundamental complexity of Bayesian active learning by examining the basic problem of learning from binary-valued queries. We are particularly interested in identifying a key quantity that characterizes the number of queries required to learn to a given accuracy, given knowledge of the prior distribution from which the target is sampled. This topic is interesting both in itself, and also as a general setting in which to derive lower bounds, which apply broadly to any active learning scenario in which binary-valued queries are employed, such as the popular setting of active learning with label requests (membership queries). The analysis of the Bayesian variant of this setting is important for at least two reasons: first, for practical reasons, as minimax analyses tend to emphasize scenarios much more difficult to learn from than what the world often offers us, so that the smoothed or average-case analysis offered by a Bayesian setting can often be an informative alternative, and second, for philosophical reasons, owing to the decision-theoretic interpretation of rational inference, which is typically formulated in a Bayesian setting.

There is much related work on active learning with binary-valued queries. However, perhaps the most relevant for us is the result of (Kulkarni et al., 1993). In this classic work, they allow a learning algorithm to ask *any* question with a yes/no answer, and derive a precise characterization of the number of these binary-valued queries necessary and sufficient for learning a target classifier to a prescribed accuracy, in a PAC-like framework. In particular, they find this quantity is essentially characterized by

© Springer-Verlag Berlin Heidelberg 2010

$\log \mathcal{M}(\epsilon)$, where $1 - \epsilon$ is the desired accuracy, and $\mathcal{M}(\epsilon)$ is the size of a maximal ϵ-packing of the concept space.

In addition to being quite interesting in their own right, these results have played a significant role in the recent developments in active learning with "label request" queries for binary classification (Hanneke, 2007b; Hanneke, 2007a; Dasgupta, 2005). Specifically, since label requests can be viewed as a type of binary-valued query, the number of label requests necessary for learning is naturally lower bounded by the number of arbitrary binary-valued queries necessary for learning. We therefore always expect to see some term relating to $\log \mathcal{M}(\epsilon)$ in our sample complexity bounds for active learning with label requests (though this factor is typically represented by its upper bound: $\propto VC(\mathbb{C}) \log(1/\epsilon)$, where $VC(\mathbb{C})$ is the VC dimension).

Also related is a certain thread of the literature on sample complexity bounds for Bayesian learning. In particular, (Haussler et al., 1994a) study the passive learning problem in a Bayesian setting, and study the effect of the information made available via access to the prior. In many cases, the learning problem is made significantly easier than the worst-case scenarios of the PAC model. In particular, (building from the work of (Haussler et al., 1994b)) they find that $VC(\mathbb{C})/\epsilon$ random labeled examples are sufficient to achieve expected error rate at most ϵ using the Bayes classifier.

Allowing somewhat more general types of queries than (Haussler et al., 1994a), a paper by (Freund et al., 1997; Seung et al., 1992) studied an algorithm known as Query by Committee (QBC). Specifically, QBC is allowed to sequentially decide which points to select, observing each response before selecting the next data point to observe. They found this additional flexibility can sometimes pay off significantly, reducing the expected number of queries needed exponentially to only $O(\log(1/\epsilon))$. However, these results only seem to apply to a very narrow family of problems, where a certain expected information gain quantity is lower bounded by a constant, a situation which seems fairly uncommon among the types of learning problems we are typically most interested in (informative priors, or clustered data). Thus, to our knowledge, the general questions, such as how much advantage we actually get from having access to the prior π, and what fundamental quantities describe the intrinsic complexity of the learning problem, remain virtually untouched in the published literature.

The "label request" query discussed in these Bayesian analyses represents a type of binary-valued query, though quite restricted compared to the powerful queries analyzed in the present work. As a first step toward a more complete understanding of the Bayesian active learning problem, we propose to return to the basic question of how many binary-valued queries are necessary and sufficient in general; but unlike the (Kulkarni et al., 1993) analysis, we adopt the Bayesian perspective of (Haussler et al., 1994a) and (Freund et al., 1997), so that the algorithms in question will directly depend on the prior π. In fact, we investigate the problem in a somewhat more general form, where reference to the underlying data distribution is replaced by direct reference to the induced pseudo-metric between elements of the concept space. As we point out below, this general problem has deep connections to many problems commonly studied in information theory (e.g., the analysis of lossy compression); for instance, one might view the well-known asymptotic results of rate distortion theory as a massively multitask variant of this problem. However, to our knowledge, the basic question of the number of binary queries necessary to approximate a single random target h^* to a given accuracy, given access to the distribution π of h^*, has not previously been addressed in generality.

Below, we are able to derive upper and lower bounds on the query complexity based on a natural analogue of the bounds of (Kulkarni et al., 1993). Specifically, we find that in this Bayesian setting, under an assumption of bounded doubling dimension, the query complexity is controlled by the entropy of a partition induced by a maximal ϵ-packing (specifically, the natural Voronoi partition); in particular, the worst-case value of this entropy is the $\log \mathcal{M}(\epsilon)$ bound of (Kulkarni et al., 1993), which represents a uniform prior over the regions of the partition. The upper bound is straightforward to derive, but nice to have; but our main contribution is the lower bound, the proof of which is somewhat more involved.

The rest of this paper is organized as follows. In Section 2, we introduce a few important quantities used in the statement of the main theorem. Following this, Section 3 contains a statement of our main result, along with some explanation. Section 4 contains the proof of our result, followed by Section 5, which states a few of the many remaining open questions about Bayesian active learning.

2 Definitions and Notation

We will formalize our discussion in somewhat more abstract terms.

Formally, throughout this discussion, we will suppose $\mathbb{C}^*$ is an arbitrary (nonempty) collection of objects, equipped with a separable pseudo-metric $\rho : \mathbb{C}^* \times \mathbb{C}^* \to [0, \infty)$.[1] We suppose $\mathbb{C}^*$ is equipped with its Borel σ-algebra induced by ρ. There is additionally a (nonempty, measurable) set $\mathbb{C} \subseteq \mathbb{C}^*$, and we denote by $\bar{\rho} = \sup_{h_1, h_2 \in \mathbb{C}} \rho(h_1, h_2)$. Finally, there is a probability measure π with $\pi(\mathbb{C}) = 1$, known as the "prior," and a $\mathbb{C}$-valued random variable h^* with distribution π, known as the "target." As the prior is essentially arbitrary, the results below will hold for *any* prior π.

As an example, in the special case of the binary classifier learning problem studied by (Haussler et al., 1994a) and (Freund et al., 1997), $\mathbb{C}^*$ is the set of all measurable classifiers $h : \mathcal{X} \to \{-1, +1\}$, $\mathbb{C}$ is the "concept space," h^* is the "target function," and $\rho(h_1, h_2) = \mathbb{P}_{X \sim \mathcal{D}}(h_1(X) \neq h_2(X))$, where $\mathcal{D}$ is the distribution of the (unlabeled) data; in particular, $\rho(h, h^*) = er(h)$ is the "error rate" of h.

To discuss the fundamental limits of learning with binary-valued queries, we define the quantity $\mathrm{QueryComplexity}(\epsilon)$, for $\epsilon > 0$, as the minimum possible expected number of binary queries for any learning algorithm guaranteed to return $\hat{h}$ with $\mathbb{E}[\rho(\hat{h}, h^*)] \leq \epsilon$, where the only random variable in the expectation is $h^* \sim \pi$ (and $\hat{h}$, which is itself determined by h^* and the sequence of queries). For simplicity, we restrict ourselves to deterministic algorithms in this paper, so that the only source of randomness is h^*.

Alternatively, there is a particularly simple interpretation of the notion of an algorithm based on arbitrary binary-valued queries, which leads to an equivalent definition

[1] The set $\mathbb{C}^*$ will not play any significant role in the analysis, except to allow for improper learning scenarios to be a special case of our setting.

of $\mathrm{QueryComplexity}(\epsilon)$: namely, a prefix-free code. That is, any deterministic algorithm that asks a sequence of yes/no questions before terminating and returning some $\hat{h} \in \mathbb{C}^*$ can be thought of as a binary decision tree (no = left, yes = right), with the return $\hat{h}$ values stored in the leaf nodes. Transforming each root-to-leaf path in the decision tree into a codeword (left = 0, right = 1), we see that the algorithm corresponds to a prefix-free binary code. Conversely, given any prefix-free binary code, we can construct an algorithm based on sequentially asking queries of the form "what is the first bit in the codeword $C(h^*)$ for h^*?", "what is the second bit in the codeword $C(h^*)$ for h^*?", etc., until we obtain a complete codeword, at which point we return the value that codeword decodes to. From this perspective, we can state an equivalent definition of $\mathrm{QueryComplexity}(\epsilon)$ in the language of lossy codes.

Formally, a *code* is a pair of (measurable) functions (C, D). The *encoder*, C, maps any element $h \in \mathbb{C}$ to a binary sequence $C(h) \in \bigcup_{q=0}^{\infty}\{0,1\}^q$ (the *codeword*). The *decoder*, D, maps any element $c \in \bigcup_{q=0}^{\infty}\{0,1\}^q$ to an element $D(c) \in \mathbb{C}^*$. For any $q \in \{0, 1, \ldots\}$ and $c \in \{0,1\}^q$, let $|c| = q$ denote the *length* of c. A *prefix-free* code is any code (C, D) such that no $h_1, h_2 \in \mathbb{C}$ have $c^{(1)} = C(h_1)$ and $c^{(2)} = C(h_2)$ with $c^{(1)} \neq c^{(2)}$ but $\forall i \leq |c^{(1)}|$, $c_i^{(2)} = c_i^{(1)}$: that is, no codeword is a prefix of another (longer) codeword.

Here, we consider a setting where the code (C, D) may be *lossy*, in the sense that for some values of $h \in \mathbb{C}$, $\rho(D(C(h)), h) > 0$. Our objective is to design the code to have small expected loss (in the ρ sense), while maintaining as small of an expected codeword length as possible, where expectations are over the target h^*, which is also the element of $\mathbb{C}$ we encode. The following defines the optimal such length.

Definition 1. *For any $\epsilon > 0$, define the* query complexity *as*

$$\mathrm{QueryComplexity}(\epsilon)$$
$$= \inf\left\{\mathbb{E}\Big[|C(h^*)|\Big] : (C, D) \textit{ is a prefix-free code with } \mathbb{E}\left[\rho\Big(D(C(h^*)), h^*\Big)\right] \leq \epsilon\right\},$$

where the random variable in both expectations is $h^* \sim \pi$.

Recalling the equivalence between prefix-free binary codes and deterministic learning algorithms making arbitrary binary-valued queries, note that this definition is equivalent to the earlier definition.

Returning to the specialized setting of binary classification for a moment, we see that this corresponds to the minimum possible expected number of binary queries for a learning algorithm guaranteed to have expected error rate at most ϵ.

Given this coding perspective, we should not be surprised to see an entropy quantity appear in the results of the next section. Specifically, define the following quantities.

Definition 2. *For any $\epsilon > 0$, define $\mathcal{Y}(\epsilon) \subseteq \mathbb{C}$ as a maximal ϵ-packing of $\mathbb{C}$. That is, $\forall h_1, h_2 \in \mathcal{Y}(\epsilon)$, $\rho(h_1, h_2) \geq \epsilon$, and $\forall h \in \mathbb{C} \setminus \mathcal{Y}(\epsilon)$, the set $\mathcal{Y}(\epsilon) \cup \{h\}$ does not satisfy this property.*

For our purposes, if multiple maximal ϵ-packings are possible, we can choose to define $\mathcal{Y}(\epsilon)$ arbitrarily from among these; the results below hold for any such choice.

Recall that any maximal ϵ-packing of $\mathbb{C}$ is also an ϵ-cover of $\mathbb{C}$, since otherwise we would be able to add to $\mathcal{Y}(\epsilon)$ the $h \in \mathbb{C}$ that escapes the cover.

Next we define a complexity measure, a type of entropy, which serves as our primary quantity of interest in the analysis of $\mathrm{QueryComplexity}(\epsilon)$. It is specified in terms of a partition induced by $\mathcal{Y}(\epsilon)$, defined as follows.

Definition 3. *For any $\epsilon > 0$, define*

$$\mathcal{P}(\epsilon) = \left\{ \left\{ h \in \mathbb{C} : f = \operatorname*{argmin}_{g \in \mathcal{Y}(\epsilon)} \rho(h, g) \right\} : f \in \mathcal{Y}(\epsilon) \right\},$$

where we break ties in the argmin *arbitrarily but consistently (e.g., based on a pre-defined preference ordering of $\mathcal{Y}(\epsilon)$). If the* argmin *is not defined (i.e., the* min *is not realized), take any $f \in \mathcal{Y}(\epsilon)$ with $\rho(f, h) \le \epsilon$ (one must exist by maximality of $\mathcal{Y}(\epsilon)$).*

Definition 4. *For any finite (or countable) partition $\mathcal{S}$ of $\mathbb{C}$ into measurable regions (subsets), define the* entropy *of $\mathcal{S}$*

$$\mathcal{H}(\mathcal{S}) = -\sum_{S \in \mathcal{S}} \pi(S) \log_2 \pi(S).$$

In particular, we will be interested in the quantity $\mathcal{H}(\mathcal{P}(\epsilon))$ in the analysis below.

Finally, we will require a notion of dimensionality for the pseudo-metric ρ. For this, we adopt the well-known *doubling dimension* (Gupta et al., 2003).

Definition 5. *Define the* doubling dimension *d as the smallest value d such that, for any $h \in \mathbb{C}$, and any $\epsilon > 0$, the size of the minimal $\epsilon/2$-cover of the ϵ-radius ball around h is at most 2^d.*

That is, for any $h \in \mathbb{C}$ and $\epsilon > 0$, there exists a set $\{h_i\}_{i=1}^{2^d}$ of 2^d elements of $\mathbb{C}$ such that

$$\{h' \in \mathbb{C} : \rho(h', h) \le \epsilon\} \subseteq \bigcup_{i=1}^{2^d} \{h' \in \mathbb{C} : \rho(h', h_i) \le \epsilon/2\}.$$

Note that, as defined here, d is a constant (i.e., has no dependence on h or ϵ). See (Bshouty et al., 2009) for a discussion of the doubling dimension of spaces $\mathbb{C}$ of binary classifiers, in the context of learning theory.

3 Main Result

Our main result can be summarized as follows. Note that, since we took the prior to be *arbitrary* in the above definitions, this result holds for *any* prior π.

Theorem 1. *If $d < \infty$ and $\bar{\rho} < \infty$, then there is a constant $c = O(d)$ such that $\forall \epsilon \in (0, \bar{\rho}/2)$,*

$$\mathcal{H}\left(\mathcal{P}\left(\epsilon \log_2(\bar{\rho}/\epsilon)\right)\right) - c \le \mathrm{QueryComplexity}(\epsilon) \le \mathcal{H}\left(\mathcal{P}\left(\epsilon\right)\right) + 1.$$

Due to the deep connections of this problem to information theory, it should not be surprising that entropy terms play a key role in this result. Indeed, this type of entropy seems to give a good characterization of the asymptotic behavior of the query complexity in this setting. We should expect the upper bound to be tight when the regions in $\mathcal{P}(\epsilon)$ are point-wise well-separated. However, it may be looser when this is not the case, for reasons discussed in the next section.

Although this result is stated for bounded psuedometrics ρ, it also has implications for unbounded ρ. In particular, the proof of the upper bound holds as-is for unbounded ρ. Furthermore, we can always use this lower bound to construct a lower bound for unbounded ρ, simply restricting to a bounded subset of $\mathbb{C}$ with constant probability and calculating the lower bound for that region. For instance, to get a lower bound for π being a Gaussian distribution on $\mathbb{R}$, we might note that $\pi([-1/2, 1/2])$ times the expected error rate under the *conditional* $\pi(\cdot|[-1/2, 1/2])$ lower bounds the total expected error rate. Thus, calculating the lower bound of Theorem 1 under the conditional $\pi(\cdot|[-1/2, 1/2])$ while replacing ϵ with $\epsilon/\pi([-1/2, 1/2])$ provides a lower bound on $\mathrm{QueryComplexity}(\epsilon)$.

4 Proof of Theorem 1

We first state a lemma that will be useful in the proof.

Lemma 1. *(Gupta et al., 2003) For any* $\gamma \in (0, \infty)$, $\delta \in [\gamma, \infty)$, *and* $h \in \mathbb{C}$, *we have*

$$|\{h' \in \mathcal{Y}(\gamma) : \rho(h', h) \leq \delta\}| \leq \left(\frac{4\delta}{\gamma}\right)^d .$$

Proof. See (Gupta et al., 2003). □

Proof (of Theorem 1). Throughout the proof, we will consider a set-valued random quantity $P_\epsilon(h^*)$ with value equal to the set in $\mathcal{P}(\epsilon)$ containing h^*, and a corresponding $\mathbb{C}$-valued random quantity $Y_\epsilon(h^*)$ with value equal the sole point in $P_\epsilon(h^*) \cap \mathcal{Y}(\epsilon)$: that is, the target's nearest representative in the ϵ-packing. Note that, by Lemma 1, $|\mathcal{Y}(\epsilon)| < \infty$ for all $\epsilon \in (0, 1)$. We will also adopt the usual notation for entropy (e.g., $\mathcal{H}(P_\epsilon(h^*))$) and conditional entropy (e.g., $\mathcal{H}(P_\epsilon(h^*)|X)$), both in base 2; see (Cover & Thomas, 2006) for definitions.

To establish the upper bound, we simply take C as the Huffman code for the random quantity $P_\epsilon(h^*)$ (Cover & Thomas, 2006). It is well-known that the expected length of a Huffman code for $P_\epsilon(h^*)$ is at most $\mathcal{H}(P_\epsilon(h^*)) + 1$ (in fact, is equal $\mathcal{H}(P_\epsilon(h^*))$ when the probabilities are powers of 2) (Cover & Thomas, 2006), and each possible value of $P_\epsilon(h^*)$ is assigned a unique codeword so that we can perfectly recover $P_\epsilon(h^*)$ (and thus also $Y_\epsilon(h^*)$) based on $C(h^*)$. In particular, define $D(C(h^*)) = Y_\epsilon(h^*)$. Finally, recall that any maximum ϵ-packing is also an ϵ-*cover*; that is, for every $h \in \mathbb{C}$, there is at least one $h' \in \mathcal{Y}(\epsilon)$ with $\rho(h, h') \leq \epsilon$ (otherwise, we could add h to the packing, contradicting its maximality). Thus, since every element of the set $P_\epsilon(h^*)$ has $Y_\epsilon(h^*)$ as its closest representative in $\mathcal{Y}(\epsilon)$, we must have $\rho(h^*, D(C(h^*))) = \rho(h^*, Y_\epsilon(h^*)) \leq \epsilon$. In fact, as this proof never relies on $d < \infty$ or $\bar{\rho} < \infty$, this establishes the upper bound even in the case $d = \infty$ or $\bar{\rho} = \infty$.

The proof of the lower bound is somewhat more involved, though the overall idea is simple enough. Essentially, the lower bound would be straightforward if the regions of $\mathcal{P}(\epsilon \log_2(\bar{\rho}/\epsilon))$ were separated by some distance, since we could make an argument based on Fano's inequality to say that since any $\hat{h}$ is "close" to at most one region, the expected distance from h^* is at least as large as half this inter-region distance times a quantity proportional to the entropy. However, it is not always so simple, as the regions can generally be quite close to each other (even adjacent), so that it is possible for $\hat{h}$ to be close to multiple regions. Thus, the proof will first "color" the regions of $\mathcal{P}(\epsilon \log_2(\bar{\rho}/\epsilon))$ in a way that guarantees no two regions of the same color are within distance $\epsilon \log_2(\bar{\rho}/\epsilon)$ of each other. Then we apply the above simple argument for each color separately (i.e., lower bounding the expected distance from h^* under the conditional given the color of $P_{\epsilon \log_2(\bar{\rho}/\epsilon)}(h^*)$ by a function of the entropy under the conditional), and average over the colors to get a global lower bound. The details follow.

Fix any $\epsilon \in (0, \bar{\rho}/2)$, and for brevity let $\alpha = \epsilon \log_2(\bar{\rho}/\epsilon)$. We suppose (C, D) is some prefix-free binary code (representing the learning algorithm's queries and return policy).

Define a function $\mathcal{K} : \mathcal{P}(\alpha) \to \mathbb{N}$ such that $\forall P_1, P_2 \in \mathcal{P}(\alpha)$,

$$\mathcal{K}(P_1) = \mathcal{K}(P_2) \implies \inf_{h_1 \in P_1, h_2 \in P_2} \rho(h_1, h_2) \geq \alpha, \tag{1}$$

and suppose $\mathcal{K}$ has minimum $\mathcal{H}(\mathcal{K}(P_\alpha(h^*)))$ subject to (1). We will refer to $\mathcal{K}(P)$ as the *color* of P.

Now we are ready to bound the expected distance from h^*. Let $\hat{h} = D(C(h^*))$ denote the element returned by the algorithm (decoder), and let $P_\alpha(\hat{h}; \mathcal{K})$ denote the set $P \in \mathcal{P}(\alpha)$ having $\mathcal{K}(P) = \mathcal{K}$ with smallest $\inf_{h \in P} \rho(h, \hat{h})$ (breaking ties arbitrarily). We know

$$\mathbb{E}[\rho(\hat{h}, h^*)] = \mathbb{E}\left[\mathbb{E}[\rho(\hat{h}, h^*)|\mathcal{K}(P_\alpha(h^*))]\right]. \tag{2}$$

Furthermore, by (1) and a triangle inequality, we know no $\hat{h}$ can be $\alpha/3$-close to more than one $P \in \mathcal{P}(\alpha)$ of a given color. Therefore,

$$\mathbb{E}[\rho(\hat{h}, h^*)|\mathcal{K}(P_\alpha(h^*))] \geq \frac{\alpha}{3}\mathbb{P}(P_\alpha(\hat{h}; \mathcal{K}(P_\alpha(h^*))) \neq P_\alpha(h^*)|\mathcal{K}(P_\alpha(h^*))). \tag{3}$$

By Fano's inequality, we have

$$\mathbb{E}\left[\mathbb{P}(P_\alpha(\hat{h};\mathcal{K}(P_\alpha(h^*))) \neq P_\alpha(h^*)|\mathcal{K}(P_\alpha(h^*)))\right] \geq \frac{\mathcal{H}(P_\alpha(h^*)|C(h^*), \mathcal{K}(P_\alpha(h^*))) - 1}{\log_2 |\mathcal{Y}(\alpha)|}. \tag{4}$$

It is generally true that, for a prefix-free binary code $C(h^*)$, $C(h^*)$ is a lossless prefix-free binary code for itself (i.e., with the identity decoder), so that the classic entropy lower bound on average code length (Cover & Thomas, 2006) implies $\mathcal{H}(C(h^*)) \leq \mathbb{E}[|C(h^*)|]$. Also, recalling that $\mathcal{Y}(\alpha)$ is maximal, and therefore also an α-cover, we have that any $P_1, P_2 \in \mathcal{P}(\alpha)$ with $\inf_{h_1 \in P_1, h_2 \in P_2} \rho(h_1, h_2) \leq \alpha$ have $\rho(Y_\alpha(h_1), Y_\alpha(h_2)) \leq 3\alpha$ (by a triangle inequality). Therefore, Lemma 1 implies that, for any given $P_1 \in \mathcal{P}(\alpha)$, there are at most 12^d sets $P_2 \in \mathcal{P}(\alpha)$ with $\inf_{h_1 \in P_1, h_2 \in P_2} \rho(h_1, h_2) \leq \alpha$. We therefore know there exists a function $\mathcal{K}' : \mathcal{P}(\alpha) \to \mathbb{N}$ satisfying (1) such that $\max_{P \in \mathcal{P}(\alpha)} \mathcal{K}'(P)$

$\leq 12^d$ (i.e., we need at most 12^d colors to satisfy (1)). That is, if we consider coloring the sets $P \in \mathcal{P}(\alpha)$ sequentially, for any given P_1 not yet colored, there are $< 12^d$ sets $P_2 \in \mathcal{P}(\alpha) \setminus \{P_1\}$ within α of it, so there must exist a color among $\{1, \ldots, 12^d\}$ not used by any of them, and we can choose that for $\mathcal{K}'(P_1)$. In particular, by our choice of $\mathcal{K}$ to minimize $\mathcal{H}(\mathcal{K}(P_\alpha(h^*)))$ subject to (1), this implies

$$\mathcal{H}(\mathcal{K}(P_\alpha(h^*))) \leq \mathcal{H}(\mathcal{K}'(P_\alpha(h^*))) \leq \log_2(12^d) \leq 4d.$$

Thus,

$$\begin{aligned}
&\mathcal{H}(P_\alpha(h^*)|C(h^*), \mathcal{K}(P_\alpha(h^*))) && (5)\\
&= \mathcal{H}(P_\alpha(h^*), C(h^*), \mathcal{K}(P_\alpha(h^*))) - \mathcal{H}(C(h^*)) - \mathcal{H}(\mathcal{K}(P_\alpha(h^*))|C(h^*)) && (6)\\
&\geq \mathcal{H}(P_\alpha(h^*)) - \mathcal{H}(C(h^*)) - \mathcal{H}(\mathcal{K}(P_\alpha(h^*))) \geq \mathcal{H}(P_\alpha(h^*)) - \mathbb{E}\left[|C(h^*)|\right] - 4d \\
&= \mathcal{H}(\mathcal{P}(\alpha)) - \mathbb{E}\left[|C(h^*)|\right] - 4d. && (7)
\end{aligned}$$

Thus, combining (2), (3), (4), and (7), we have

$$\begin{aligned}
\mathbb{E}[\rho(\hat{h}, h^*)] &\geq \frac{\alpha}{3}\frac{\mathcal{H}(\mathcal{P}(\alpha)) - \mathbb{E}\left[|C(h^*)|\right] - 4d - 1}{\log_2 |\mathcal{Y}(\alpha)|}\\
&\geq \frac{\alpha}{3}\frac{\mathcal{H}(\mathcal{P}(\alpha)) - \mathbb{E}\left[|C(h^*)|\right] - 4d - 1}{d\log_2(4\bar{\rho}/\alpha)},
\end{aligned}$$

where the last inequality follows from Lemma 1.

Thus, for any code with

$$\mathbb{E}\left[|C(h^*)|\right] < \mathcal{H}(\mathcal{P}(\alpha)) - 4d - 1 - 3d\frac{\log_2(4\bar{\rho}/\epsilon)}{\log_2(\bar{\rho}/\epsilon)},$$

we have $\mathbb{E}[\rho(\hat{h}, h^*)] > \epsilon$, which implies

$$\text{QueryComplexity}(\epsilon) \geq \mathcal{H}(\mathcal{P}(\alpha)) - 4d - 1 - 3d\frac{\log_2(4\bar{\rho}/\epsilon)}{\log_2(\bar{\rho}/\epsilon)}.$$

Since $\log_2(4\bar{\rho}/\epsilon)/\log_2(\bar{\rho}/\epsilon) \leq 3$, we have

$$\text{QueryComplexity}(\epsilon) = \mathcal{H}(\mathcal{P}(\alpha)) - O(d).$$

□

5 Open Problems

Generally, we feel this topic of Bayesian active learning is relatively unexplored, and as such there is an abundance of ripe open problems ready for solvers.

In our present context, there are several interesting questions, such as whether the $\log(\bar{\rho}/\epsilon)$ factor in the entropy argument of the lower bound can be removed, whether the additive constant in the lower bound might be improved, and in particular whether a similar result might be obtained without assuming $d < \infty$ (e.g., by making a VC class assumption instead).

Additionally, one can ask for necessary and sufficient conditions for this entropy lower bound to be achievable via a restricted type of query, such as label requests (membership queries).

Overall, the challenge here is to understand, to as large an extent as possible, how much benefit we get from having access to the prior, and what the general form of improvements we can expect in the query complexity given this information are.

Acknowledgments

Liu Yang would like to extend her sincere gratitude to Avrim Blum and Venkatesan Guruswami for several enlightening and highly stimulating discussions.

References

Bshouty, N.H., Li, Y., Long, P.M.: Using the doubling dimension to analyze the generalization of learning algorithms. Journal of Computer and System Sciences 75, 323–335 (2009)

Cover, T.M., Thomas, J.A.: Elements of information theory. John Wiley & Sons, Inc., Chichester (2006)

Dasgupta, S.: Coarse sample complexity bounds for active learning. In: Advances in Neural Information Processing Systems, vol. 18 (2005)

Freund, Y., Seung, H.S., Shamir, E., Tishby, N.: Selective sampling using the query by committee algorithm. Machine Learning 28, 133–168 (1997)

Gupta, A., Krauthgamer, R., Lee, J.R.: Bounded geometries, fractals, and lowdistortion mbeddings. In: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (2003)

Hanneke, S.: A bound on the label complexity of agnostic active learning. In: Proceedings of the 24th International Conference on Machine Learning (2007a)

Hanneke, S.: Teaching dimension and the complexity of active learning. In: Proceedings of the 20th Annual Conference on Learning Theory (2007b)

Haussler, D., Kearns, M., Schapire, R.: Bounds on the sample complexity of Bayesian learning using information theory and the VC dimension. Machine Learning 14, 83–113 (1994a)

Haussler, D., Littlestone, N., Warmuth, M.: Predicting $\{0, 1\}$-functions on randomly drawn points. Information and Computation 115, 248–292 (1994b)

Kulkarni, S.R., Mitter, S.K., Tsitsiklis, J.N.: Active learning using arbitrary binary valued queries. Machine Learning 11, 23–35 (1993)

Seung, H.S., Opper, M., Sompolinsky, H.: Query by committee. In: Proceedings of the 5th Workshop on Computational Learning Theory (1992)

Approximation Stability and Boosting

Wei Gao and Zhi-Hua Zhou

National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing 210093, China
{gaow,zhouzh}@lamda.nju.edu.cn

Abstract. Stability has been explored to study the performance of learning algorithms in recent years and it has been shown that stability is sufficient for generalization and is sufficient and necessary for consistency of ERM in the general learning setting. Previous studies showed that AdaBoost has almost-everywhere uniform stability if the base learner has L_1 stability. The L_1 stability, however, is too restrictive and we show that AdaBoost becomes constant learner if the base learner is not real-valued learner. Considering that AdaBoost is mostly successful as a classification algorithm, stability analysis for AdaBoost when the base learner is not real-valued learner is an important yet unsolved problem. In this paper, we introduce the *approximation stability* and prove that approximation stability is sufficient for generalization, and sufficient and necessary for learnability of AERM in the general learning setting. We prove that AdaBoost has approximation stability and thus has good generalization, and an exponential bound for AdaBoost is provided.

1 Introduction

Stability has been considered as an important tool for studying the performance of learning algorithms in recent years. Intuitively, the stability of a learning algorithm can be referred as perturbation sensitivity in the training sample. It was first introduced in [5] for estimating leave-one-out error and further used to bound empirical risk of regression [10], which discovered a connection between finite VC dimension and stability. Bousquet and Elisseeff [3] obtained an exponential bound for uniform stability and proved that the Tikhonov regularized algorithms hold uniform stability property. Kutin and Niyogi [12] generalized the uniform stability to almost-everywhere algorithmic stability and derived generalization error bounds with extensions of McDiarmid's inequality. Stability has also been employed to bound the bias and variance of estimators for ERM (empirical risk minimization) or general algorithm [17]. An influential work of Mukherjee et al. [16] showed that stability is sufficient for generalization and sufficient and necessary for consistency of ERM in supervised regression and classification. Later, this result was extended to general learning setting by Shalev-Shwartz et al. [22].

AdaBoost [6, 7] is one of the most influential learning algorithms during the past decades. Many theoretical efforts have been devoted to studying the mysteries behind the great success of AdaBoost. There are different interpretations

© Springer-Verlag Berlin Heidelberg 2010

from different aspects, and they have shed important insights for understanding the behaviors of AdaBoost. However, debates are still lasting to date, for example, on margin-based interpretation [4, 21, 19, 25] and statistical-view-based interpretation [2, 9, 15].

Considering the recent advances in stability, it is interesting to study the stability issues of AdaBoost. Kutin and Niyogi [11] proved that AdaBoost has almost-everywhere uniform stability if the base learner is L_1 stable. To the best of our knowledge, this is the only stability result for AdaBoost. The requirement of L_1 stability, however, is too restrictive, and as we will show in Section 4, AdaBoost becomes constant learner when the base learner is not real-valued learner. Note that as Freund and Schapire [8] indicated, AdaBoost is a *classification* algorithm, and so it is important to study the situation when the base learner is not real-valued learner.

In this paper, we introduce the notion of *approximation stability*, and prove that the approximation stability is sufficient for generalization, and is sufficient and necessary for learnability of AERM (asymptotical empirical risk minimization) in the general learning setting. Then, we prove that AdaBoost has approximation stability and thus has good generalization, and an exponential bound for AdaBoost is provided. All bounds obtained in this paper do not rely on any space complexity measure, but rather on the way the algorithm searches the space, and thus can be used even when the VC dimension is infinite.

In the rest of this paper we begin by introducing some notations and background knowledge in Section 2. Then, we give our results in Sections 3 and 4, and finally present the detail proofs in Section 5.

2 Preliminaries

2.1 Notations

Let $\mathcal{Z}$ denote an instance space and $\mathcal{D}$ denote an unknown probability distribution over $\mathcal{Z}$. We use $\Pr_{\mathcal{D}}[\cdot]$ to refer to the probability with respect to $\mathcal{D}$ and $\Pr_S[\cdot]$ to denote the probability with respect to a uniform distribution over the training sample S. Similarly, we use $E_D[\cdot]$ and $E_S[\cdot]$ to denote the expected values, respectively. For a positive number n, we denote by $[n]$ the set $\{1, 2, \cdots, n\}$. Given two distributions p and q with finite support, $||p - q||$ is defined to be the L_1-norm of $p - q$, i.e., $||p - q|| = \sum_{z\in\mathcal{Z}} |p(z) - q(z)|$. For a given sample $S = \{z_1, z_2, \cdots, z_n\}$ drawn i.i.d. according to distribution $\mathcal{D}$, let $S^i = S \setminus z_i$ be the sample with the i-th example z_i removed from S. For any $u \in \mathcal{Z}$, we denote by $S^{i,u} = S^i \cup \{u\}$ the sample with the i-th example z_i replaced by u in S.

A learning algorithm is a function $\mathbb{A}$ which maps a distribution p over $\mathcal{Z}$ onto a function $\mathbb{A}_p \in \mathcal{H}$, where $\mathcal{H}$ is a specific hypothesis class. Note that $\mathbb{A}_S$ means $\mathbb{A}_p$ where p is the uniform distribution on the training sample S. Throughout this paper, we consider symmetric algorithms, i.e., algorithms depend upon the given sample but not on the order of examples in the sample.

To measure the performance, we introduce a cost function $c\colon \mathcal{H} \times \mathcal{Z} \to \mathbb{R}$. We assume such cost function is bounded by some constant B, i.e., $|c(h, z)| \leq B$

for every $h \in \mathcal{H}$ and $z \in \mathcal{Z}$. Given a sample S with size n and function $h \in \mathcal{H}$, define the empirical risk and expect risk, respectively, as

$$R_S(h) = E_{z \sim S}[c(h,z)] = \frac{1}{n} \sum_{z \in S} c(h,z) \text{ and } R(h) = E_{z \sim \mathcal{D}}[c(h,z)].$$

The general learning setting [24] is to minimize the expect risk, i.e., $\min_{h \in \mathcal{H}} R(h)$. Such setting comprises density estimation, stochastic optimization and supervised classification and regression. For instance, in supervised learning, $z = (x, y)$ is an instance-label pair and $c(h, z) = c(h(x), y)$ is the prediction loss for $h \in \mathcal{H}$.

Classical learning theory focuses on ERM, that is,

$$R_S(\mathbb{A}_S) = R_S(\hat{h}_S) = \min_{h \in \mathcal{H}} R_S(h),$$

where we denote by $\hat{h}_S = \arg\min_{h \in \mathcal{H}} R_S(h)$. A learning algorithm $\mathbb{A}$ is said to be AERM with rate $\epsilon_{\text{erm}}(n)$ under distribution $\mathcal{D}$ if

$$E_{S \sim \mathcal{D}^n}[R_S(\mathbb{A}_S) - R_S(\hat{h}_S)] \le \epsilon_{\text{erm}}(n).$$

We focus on AERM learning problems in this paper and ERM can be resolved in a similar way.

We say a learning algorithm $\mathbb{A}$ is consistent with rate $\epsilon_{\text{con}}(n)$ under distribution $\mathcal{D}$ if

$$E_{S \sim \mathcal{D}^n}[R(\mathbb{A}_S) - R(h^*)] \le \epsilon_{\text{con}}(n) \text{ for all } n,$$

where $h^* = \arg\min_{h \in \mathcal{H}} R(h)$. An algorithm $\mathbb{A}$ is universally consistent with rate $\epsilon_{\text{con}}(n)$ if it is consistent with rate $\epsilon_{\text{con}}(n)$ under all distributions $\mathcal{D}$ over $\mathcal{Z}$. A problem is learnable if there exists a universal consistent algorithm. The most influential result in classical learning theory for supervised classification and regression is that a problem is learnable if and only if the empirical risk $R_S(h)$ converges to the expect risk $R(h)$ [24]. This equivalence, however, does not always hold in the general learning setting [1, 23].

We say a learning algorithm $\mathbb{A}$ generalizes with rate $\epsilon_{\text{gen}}(n)$ under distribution $\mathcal{D}$ if

$$E_{S \sim \mathcal{D}^n}[|R(\mathbb{A}_S) - R_S(\mathbb{A}_S)|] \le \epsilon_{\text{gen}}(n) \text{ for all } n.$$

An algorithm $\mathbb{A}$ universally generalizes with $\epsilon_{\text{gen}}(n)$ if it generalizes with rate $\epsilon_{\text{gen}}(n)$ under all distributions $\mathcal{D}$ over $\mathcal{Z}$. In this paper we require $\epsilon_{\text{erm}}(n)$, $\epsilon_{\text{con}}(n)$, $\epsilon_{\text{gen}}(n) \to 0$ as $n \to \infty$.

2.2 Stability

Stability has been explored as an alternative for learnability. Definitions 1 and 2 show the CV stability [12, 17] and uniform stability [3], respectively.

Definition 1. *A learning algorithm $\mathbb{A}$ has CV stability $\eta(n)$ under distribution $\mathcal{D}$ if*

$$\forall\ i \in [n], \quad E_{S,u \sim \mathcal{D}^{n+1}}[|c(\mathbb{A}_S, u) - c(\mathbb{A}_{S^{i,u}}, u)|] \le \eta(n).$$

Algorithm 1. AdaBoost

```
**Input**: Sample $S = \{z_1 = (x_1, y_1), z_2 = (x_2, y_2), \cdots, z_n = (x_n, y_n)\} \in \mathcal{Z}^n$, base learner $\mathbb{A}$ and iteration rounds $T$.
**Initialization**: $P_S^1(z_i) = 1/n$ for each $z_i \in S$.
  **for** $t = 1$ to $T$ **do**
    1. Call $\mathbb{A}$ with respect to distribution $P_S^t$ to obtain a hypothesis $\mathbb{A}_{P_S^t}$.
    2. Choose $\alpha_S^t = \frac{1}{2} \ln \frac{1-\text{err}_S^t}{\text{err}_S^t}$ with $\text{err}_S^t = E_{z \sim P_S^t}[c(\mathbb{A}_{P_S^t}, z)]$, where $c(\mathbb{A}_{P_S^t}, z) = I[\mathbb{A}_{P_S^t}(x) \neq y]$.
    3. Update $P_S^{t+1}(z_i) = \frac{1}{Z_t} P_S^t(z_i) \exp(-\alpha_S^t y_i \mathbb{A}_{P_S^t}(x_i))$, where $Z_t$ is a normalization factor (such that $P_S^{t+1}$ is a distribution).
  **end for**
**Output**: The learner $\text{sgn}(\mathbb{H}_S(x))$ where $\mathbb{H}_S(x) = \sum_{t=1}^T \alpha_S^t \mathbb{A}_{P_S^t}(x)$.
```

Definition 2. *A learning algorithm* $\mathbb{A}$ *has uniform stability* $\beta(n)$ *if*

$$\forall\ S \in \mathcal{Z}^n,\ \forall\ i \in [n] \text{ and } \forall\ z, u \in \mathcal{Z},\quad |c(\mathbb{A}_S, z) - c(\mathbb{A}_{S^{i,u}}, z)| \leq \beta(n).$$

A relevant concept, on-average-LOO stability [22], is defined as follows:

Definition 3. *A learning algorithm* $\mathbb{A}$ *has on-average-LOO stability* $\beta(n)$ *if*

$$\left|\frac{1}{n}\sum\nolimits_{i=1}^n E_{S\sim D^n}[c(\mathbb{A}_{S^i}, z_i) - c(\mathbb{A}_S, z_i)]\right| \leq \beta(n).$$

Here and whenever talking about "stability" $\beta(n)$ and $\eta(n)$, we require $\beta(n), \eta(n) \to 0$ as $n \to \infty$.

In this paper we will introduce *approximation stability* which is a kind of replacement version stability. As Shalev-Shwartz et al. [22] indicated, previously many researchers defined stability with respect to the deletion rather than replacement of an example. For instance, the deletion version uniform stability [5], the hypothesis stability [3], the cross-validation-(deletion) stability [17], the CV_{loo} stability [16], etc. It is worth noting, however, that the deletion version stability implies the replacement version stability but not vice versa[1]; this is the reason why we focus on replacement version stability in this paper.

2.3 AdaBoost

Algorithm 1 shows a commonly used description of AdaBoost [6]. Kutin and Niyogi [11] studied the stability of AdaBoost and proved that when the base learner has L_1 stability and is real-valued with loss function $c(h, z) = |h(x) - y|$, AdaBoost has almost-everywhere uniform stability. The L_1 stability is given in Definition 4, which is equivalent to uniform stability, and the main result of [11] is shown in Theorem 1 using our notations.

[1] An example can be found in [16]: Let $\mathcal{Z} = \mathcal{X} \times \{+1, -1\}$ with $\mathcal{X}$ being uniform on $[0, 1]$. Suppose the target function is $t(x) = 1$ with 0/1 loss function. Given a sample S_n of size n, a non-AERM algorithm $\mathbb{A}_{S_n}(x) = (-1)^n$. Note that $\mathbb{A}_S$ does not have any deletion version stability but has replacement version uniform stability.

Definition 4. *A learning algorithm $\mathbb{A}$ has L_1 stability λ (constant) if $|c(\mathbb{A}_p, z) - c(\mathbb{A}_q, z)| \leq \lambda ||p - q||$ for any $z \in \mathcal{Z}$ and any given distributions p and q on $\mathcal{Z}$ with finite support.*

Theorem 1. *Suppose the base learner $\mathbb{A}$ has L_1 stability λ, and let*

$$\epsilon_\star = \frac{1}{2} \lim_{n\to\infty} \inf E_{S\sim\mathcal{D}^n}\Big[\inf_{\hat{S}\in\mathcal{Z}^m, m\leq n} R_S(\mathbb{A}_{\hat{S}})\Big] > 0.$$

Then, for all sufficiently large n and for all T, it holds for AdaBoost that

$$\Pr_{S\sim\mathcal{D}^n}[\forall\ i \in [n], \forall\ u, z \in \mathcal{Z}, |c(\mathbb{H}_S, z) - c(\mathbb{H}_{S^{i,u}}, z)| \leq \beta(n)] \geq 1 - \delta(n),$$

where $\beta(n) = \frac{2}{n}\sum_{t=1}^{T} 2^{t^2+1}(\lambda+1)^t/\epsilon_\star^{2t-1}$ and $\delta(n) = \exp(-n\epsilon_\star^2/2)$.

To the best of our knowledge, this is the only stability result for AdaBoost. It is worth noting, however, that it was obtained based on real-valued learner with the loss function $c(h, z) = |h(x) - y|$. We will show in Section 4 that for this result, AdaBoost becomes constant learner when the base learner is not real-valued learner. As Freund and Schapire [8] indicated, AdaBoost is a *classification* algorithm and therefore, it is important to study the stability of AdaBoost when the base learner is not real-valued learner, with the loss function $c(h, z) = I[h(x) \neq y]$ that is popularly used by classification algorithms; this remains an open problem and we will try to tackle it in the following sections.

3 Approximation Stability

We first introduce the empirical stability, expected empirical stability, validation stability and expected validation stability:

Definition 5. *A learning algorithm $\mathbb{A}$ has empirical stability $\beta(n)$ if*

$$\forall\ S \in \mathcal{Z}^n,\ \forall\ i \in [n] \text{ and } \forall\ u \in \mathcal{Z},\quad |R_S(\mathbb{A}_S) - R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})| \leq \beta(n).$$

A learning algorithm $\mathbb{A}$ has expected empirical stability $\beta(n)$ under distribution $\mathcal{D}$ if

$$\forall\ i \in [n],\quad E_{S,u\sim\mathcal{D}^{n+1}}[|R_S(\mathbb{A}_S) - R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})|] \leq \beta(n).$$

Definition 6. *A learning algorithm $\mathbb{A}$ has validation stability $\beta(n)$ under distribution $\mathcal{D}$ if*

$$\forall\ S \in \mathcal{Z}^n \text{ and } \forall\ i \in [n],\quad |R(\mathbb{A}_S) - E_{u\sim\mathcal{D}}[c(\mathbb{A}_{S^{i,u}}, u)]| \leq \beta(n).$$

A learning algorithm $\mathbb{A}$ has expected validation stability $\beta(n)$ under distribution $\mathcal{D}$ if

$$\forall\ i \in [n],\quad E_{S\sim\mathcal{D}^n}[|R(\mathbb{A}_S) - E_{u\sim\mathcal{D}}[c(\mathbb{A}_{S^{i,u}}, u)]|] \leq \beta(n).$$

An algorithm $\mathbb{A}$ has universally expected validation stability $\beta(n)$ if the stability holds with $\beta(n)$ for all distributions $\mathcal{D}$ over $\mathcal{Z}$. Combining the expected empirical stability and expected validation stability gives approximation stability:

Definition 7. *A learning algorithm $\mathbb{A}$ has approximation stability $(\beta_1(n), \beta_2(n))$ under distribution $\mathcal{D}$ if it exhibits both expected empirical stability $\beta_1(n)$ and expected validation stability $\beta_2(n)$.*

We prove that approximation stability is sufficient for generalization in the following theorem:

Theorem 2. *If an algorithm $\mathbb{A}$ has approximation stability $(\beta_1(n), \beta_2(n))$, then $\mathbb{A}$ generalizes with rate $\epsilon_{gen}(n) = B/\sqrt{n} + \sqrt{3\beta_1(n)B/2 + 4\beta_2(n)B + 3B^2/\sqrt{n}}$, that is,*

$$E_{S\sim\mathcal{D}^n}[|R(\mathbb{A}_S) - R_S(\mathbb{A}_S)|] \le B/\sqrt{n} + \sqrt{3\beta_1(n)B/2 + 4\beta_2(n)B + 3B^2/\sqrt{n}}.$$

Note that the CLT (central limit theorem) guarantees that the average of i.i.d. random variables converges to expectation. However, $\mathbb{A}_S$ is dependent on S and thus the CLT is not applicable. The proof in Section 5 shows that the combination of expected validation stability and expected empirical stability implies generalization, though neither the expected validation stability nor the expected empirical stability is sufficient.

Next, we study the relationship between the approximation stability and the learnability of AERM in the general learning setting. Lemma 1 shows that AERM implies expected empirical stability. Hence we only need to study the relationship between the expected validation stability and the learnability of AERM. Theorem 3 establishes the equivalence between them.

Lemma 1 (AERM $\Rightarrow$ Expected empirical stability). *If a learning algorithm $\mathbb{A}$ is AERM with rate $\epsilon_{erm}(n)$ under distribution $\mathcal{D}$, then $\mathbb{A}$ has expected empirical stability $\beta(n) = 2\epsilon_{erm}(n) + 2B/n$.*

Proof. For any $i \in [n]$ and any $u \in \mathcal{Z}$, we have

$$\begin{aligned} E_{S\sim\mathcal{D}^n}[|R_S(\mathbb{A}_S) - R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})|] &\le E_{S\sim\mathcal{D}^n}[|R_S(\mathbb{A}_S) - R_S(\hat{h}_S)|] \\ + E_{S\sim\mathcal{D}^n}[|R_S(\hat{h}_S) - R_{S^{i,u}}(\hat{h}_{S^{i,u}})|] &+ E_{S\sim\mathcal{D}^n}[|R_{S^{i,u}}(\hat{h}_{S^{i,u}}) - R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})|] \\ &\le 2\epsilon_{\text{erm}}(n) + 2B/n, \end{aligned}$$

since $|R_S(\hat{h}_S) - R_{S^{i,u}}(\hat{h}_{S^{i,u}})| \le 2B/n$ from the definition of ERM. □

Theorem 3. *The following are equivalent for an AERM:*

- *Universal expected validation stability;*
- *Universal consistency;*
- *Universal generalization.*

The equivalence between the on-average-LOO stability and learnability has been established in [22]. We thus work by establishing the equivalence between the expected validation stability and on-average-LOO stability under AERM, though this equivalence does not always hold in general and we use other techniques in such case in the proof.

It is worth noting that the uniform stability [3], which could be strictly stronger than any other stability, is not necessary for learnability [22]. Mukherjee et al. [16] suggested that LOO stability implies generalization, and is necessary and sufficient for consistency of ERM via uniform convergence of $R_S(h)$ to $R(h)$. It is well-known that uniform convergence is not equivalent to ERM consistency [1, 23] and thus their work is specific to supervised learning. In the general learning setting, the equivalence between on-average-LOO stability and learnability has been established for AERM [22]. However, out of the AERM framework there are also many useful learning algorithms, on which the on-average-LOO stability could not be applied. For instance, we could not guarantee that AdaBoost is AERM, and thus our approximation stability is meaningful for its analysis. Overall, comparing to previous stabilities, our approximation stability does not only promise generalization for general algorithm, but also guarantee sufficiency and necessity of learnability of AERM in the general setting.

Finally, we derive a bound for learning algorithm $\mathbb{A}$ which has both empirical stability and validation stability. The following theorem shows that the empirical risk converges to expect risk with high probability when $\beta_1(n) = o(n^{-\frac{1}{2}})$ and $\beta_2(n) = o(n^{-\frac{1}{2}})$ where $o(n)$ represents $\frac{o(n)}{n} \to 0$ as $n \to \infty$.

Theorem 4. *If a learning algorithm $\mathbb{A}$ has both empirical stability $\beta_1(n)$ and validation stability $\beta_2(n)$ under distribution $\mathcal{D}$, then, for all $n \geq 1$ and $\epsilon > 0$,*

$$\Pr_{S\sim\mathcal{D}^n}[|R(\mathbb{A}_S) - R_S(\mathbb{A}_S)| \geq \epsilon + \beta_2(n)] \leq 2\exp\left(\frac{-2\epsilon^2}{n(\beta_1(n) + 2\beta_2(n))^2}\right).$$

Uniform stability is sufficient for exponential generalization bound [3], however, it can only be used for regression or classification with real-valued learners. Note that the uniform stability implies empirical stability and validation stability, but not vice versa. Thus we get an exponential bound though our assumption is weaker than that used by [3] for the uniform stability bound.

Proof. Let $F(S) = R(\mathbb{A}_S) - R_S(\mathbb{A}_S)$. For any $i \in [n]$, we have

$$\begin{aligned}&|E_{S\sim\mathcal{D}^n}[F(S)]| \leq |E_{S,u\sim\mathcal{D}^{n+1}}[R(\mathbb{A}_S) - R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})]| \\ &+ |E_{S,u\sim\mathcal{D}^{n+1}}[R_{S^{i,u}}(\mathbb{A}_{S^{i,u}}) - R_S(\mathbb{A}_S)]| = |E_{S,u\sim\mathcal{D}^{n+1}}[R(\mathbb{A}_S) - R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})]|,\end{aligned}$$

by using $E_{S,u\sim\mathcal{D}^{n+1}}[R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})] = E_{S\sim\mathcal{D}^n}[R_S(\mathbb{A}_S)]$. From symmetry and i.i.d assumption, $E_{S,u\sim\mathcal{D}^{n+1}}[R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})] = E_{S,u\sim\mathcal{D}^{n+1}}[c(\mathbb{A}_{S^{i,u}}, u)]$, which leads to

$$\begin{aligned}&|E_{S,u\sim\mathcal{D}^{n+1}}[R(\mathbb{A}_S) - R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})]| \\ &\qquad = |E_{S\sim\mathcal{D}^n}[R(\mathbb{A}_S) - E_{u\sim\mathcal{D}}[c(\mathbb{A}_{S^{i,u}}, u)]]| \leq \beta_2(n).\end{aligned}$$

Thus we bound $|E_{S\sim\mathcal{D}^n}[F(S)]| \leq \beta_2(n)$. Meanwhile, it holds

$$\begin{aligned}|F(S) - F(S^{i,u})| &\leq |R(\mathbb{A}_S) - R(\mathbb{A}_{S^{i,u}})| + |R_S(\mathbb{A}_S) - R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})| \\ &\leq |R(\mathbb{A}_S) - E_{z\sim\mathcal{D}}[c(\mathbb{A}_{S^{i,z}}, z)] + E_{z\sim\mathcal{D}}[c(\mathbb{A}_{S^{i,z}}, z)] - R(\mathbb{A}_{S^{i,u}})| \\ &\qquad + \beta_1(n) \leq \beta_1(n) + 2\beta_2(n).\end{aligned}$$

This theorem follows by applying McDiarmid formula [14] to $F(S)$. □

4 Approximation Stability for AdaBoost

The following lemma shows that the L_1 stability is too restrictive for non real-valued learners with cost function $c(h,z) = I[h(x) \neq y]$.

Lemma 2. *If a learning algorithm $\mathbb{A}$ has L_1 stability λ, then $\mathbb{A}_{S_n}$ is a constant algorithm for $n > 2\lambda$.*

Proof. From the definition of L_1 stability, we have

$$\forall\ S \in \mathcal{Z}^n, \forall\ i \in [n] \text{ and } \forall\ z, u \in \mathcal{Z}, \quad |c(\mathbb{A}_S, z) - c(\mathbb{A}_{S^{i,u}}, z)| \leq 2\lambda/n.$$

It follows $\mathbb{A}_S(z) = \mathbb{A}_{S^{i,u}}(z)$ since $c(h,z) \in \{0,1\}$ and $|c(\mathbb{A}_S, z) - c(\mathbb{A}_{S^{i,u}}, z)| < 1$ for $n > 2\lambda$. We can also prove $\mathbb{A}_S(z) = \mathbb{A}_{S^i}(z)$ for all $z \in \mathcal{Z}$ in a similar way. $\square$

If the base learner in AdaBoost is not real-valued learner for large-size sample, then the base learner is a constant learner according to the above lemma. This follows that AdaBoost becomes a constant learner. Thus, Theorem 1, the only stability result for AdaBoost, does not completely explain the stability of AdaBoost for general base learner.

Since AdaBoost is mostly successful as a classification algorithm, in contrast to considering real-valued base learner with loss function $c(h,z) = |h(x) - y|$, it may be more interesting to consider non real-valued base learner with loss function $c(h,z) = I[h(x) \neq y]$ which is adopted by classifiers such as decision trees and decision stumps that are popularly used with AdaBoost in practice.

Below we will discuss the stability of AdaBoost. Observing that

$$\Pr_S[y \neq \text{sgn}(\mathbb{H}_S(x))] = E_S[I[y\mathbb{H}_S(x) \leq 0]] \leq E_S[\exp(-y\mathbb{H}_S(x))],$$

we choose the cost function for $\mathbb{H}_S(x)$ as $c(\mathbb{H}_S, z) = \exp(-y\mathbb{H}_S(x))$. This is also in accordance with the theory that AdaBoost can be regarded as a coordinate descent algorithm [4, 9, 13, 18] for minimizing $R_S(\mathbb{H}_S)$. For base learner, we set the cost function $c(\mathbb{A}_S^t, z) = I[\mathbb{A}_S^t(x) \neq y]$ described in Algorithm 1.

We assume the iteration number T for AdaBoost is given in advance, and thus T is a constant since stability could not be used to analyze AdaBoost for unfixed or infinite T. We also assume $\gamma \leq \text{err}_S^t \leq 1 - \gamma$ for some small $\gamma > 0$, because $c(\mathbb{H}_S, z)$ may approach to infinity if $\text{err}_S^t \to 0$ or $\text{err}_S^t \to 1$, which goes beyond our discussion (bounded cost function). Such assumption can be viewed as a variation of "bounded edges" in [20]. A bound for $c(\mathbb{H}_S, z)$ is given as follows.

Lemma 3. *For constant $T \geq 1$ and any $S \in \mathcal{Z}^n$, if the base learner in each iteration satisfies $\gamma \leq err_S^t \leq 1 - \gamma$ with $\gamma > 0$, then $c(\mathbb{H}_S, z) \leq \big((1-\gamma)/\gamma\big)^{T/2}$.*

Proof. Since $\alpha_S^t = \frac{1}{2}\ln((1-\text{err}_S^t)/\text{err}_S^t)$ and $\gamma \leq \text{err}_S^t \leq 1-\gamma$, we have $\frac{1}{2}\ln(\gamma/(1-\gamma)) \leq \alpha_S^t \leq \frac{1}{2}\ln((1-\gamma)/\gamma)$. It follows $c(\mathbb{H}_S, z) = \exp\big(-y\sum_{t=1}^T \alpha_S^t \mathbb{A}_{P_S^t}(x)\big) \leq \exp\big(\sum_{t=1}^T |\alpha_S^t|\big) \leq \big((1-\gamma)/\gamma\big)^{T/2}$ as desired. $\square$

Denote by B the bound of $c(\mathbb{H}_S, z)$ for notational simplicity. We have the following theorem on the approximation stability of AdaBoost:

Theorem 5. *AdaBoost has approximation stability $(\beta_1(n), \beta_2(n))$ for constant $T \geq 1$, if the base learner in each round has CV stability $\eta(n)$, and for any $u \in \mathcal{Z}$, $i \in [n]$, $t \in [T]$ and small $\gamma > 0$, the following holds:*

$$E_{S,u\sim\mathcal{D}^{n+1}}[|err_S^t - err_{S^{i,u}}^t|] \leq \zeta(n) \quad \text{and} \quad \gamma \leq err_S^t,\, err_{S^{i,u}}^t \leq 1-\gamma.$$

Here

$$\beta_1(n) = \frac{\zeta(n)T}{\sqrt{\gamma(1-\gamma)}} \text{ and } \beta_2(n) = \frac{BT}{2}\left(\eta(n)\ln\frac{1-\gamma}{\gamma} + \frac{\zeta(n)}{\gamma(1-\gamma)}\right).$$

Also, we have

$$E_{S\sim\mathcal{D}^n}[|R(\mathbb{H}_S) - R_S(\mathbb{H}_S)|] \leq B/\sqrt{n} + \sqrt{3\beta_1(n)B/2 + 4\beta_2(n)B + 3B^2/\sqrt{n}}.$$

We can also have a tighter bound for AdaBoost by considering Theorem 4:

Theorem 6. *AdaBoost has empirical stability $\beta_1(n)$ and validation stability $\beta_2(n)$ for constant $T \geq 1$, if for any $u, S \in \mathcal{Z}^{n+1}$, $i \in [n]$, $t \in [T]$ and small $\gamma > 0$, the following holds:*

$$E_{u\sim\mathcal{D}}[|c(\mathbb{A}_S, u) - c(\mathbb{A}_{S^{i,u}}, u)|] \leq \eta(n), \quad |err_S^t - err_{S^{i,u}}^t| < \zeta(n),$$

and $\gamma \leq err_S^t, err_{S^{i,u}}^t \leq 1-\gamma$. Here

$$\beta_1(n) = \frac{\zeta(n)T}{\sqrt{\gamma(1-\gamma)}} \quad \text{and} \quad \beta_2(n) = \frac{BT}{2}\left(\eta(n)\ln\frac{1-\gamma}{\gamma} + \frac{\zeta(n)}{\gamma(1-\gamma)}\right).$$

For $\epsilon > 0$, we have

$$\Pr_{S\sim\mathcal{D}^n}[|R(\mathbb{H}_S) - R_S(\mathbb{H}_S)]| \geq \epsilon + \beta_2(n)] \leq 2\exp\left(\frac{-2\epsilon^2}{n(\beta_1(n) + 2\beta_2(n))^2}\right).$$

5 Proofs

This section presents detail proofs of our main theorems. Before proceeding our proofs, we introduce some tools which will be used:

Proposition 1. *[22] Let $|X_i| \leq B$ and $X = \sum_{i=1}^n X_i/n$ for i.i.d. X_i. Then we have $E[|X - E[X]|] \leq B/\sqrt{n}$.*

Proposition 2. *[22] If X, Y are random variables s.t. $X \leq Y$ almost surely, then $E[|X|] \leq |E[X]| + 2E[|Y|]$.*

Proposition 3. *If a learning algorithm $\mathbb{A}$ has expected validation stability $\beta(n)$ under distribution $\mathcal{D}$, then $E_{S,u\sim\mathcal{D}^{n+1}}[|R(\mathbb{A}_S) - R(\mathbb{A}_{S^{i,u}})|] \leq 2\beta(n)$ for all $i \in [n]$.*

The last proposition follows from the fact $E_{S,u\sim\mathcal{D}^{n+1}}[|R(\mathbb{A}_S) - R(\mathbb{A}_{S^{i,u}})|] = E_{S,u\sim\mathcal{D}^{n+1}}[|E_{z\sim\mathcal{D}}[c(\mathbb{A}_S, z) - c(\mathbb{A}_{S^{i,z}}, z) + c(\mathbb{A}_{S^{i,z}}, z) - c(\mathbb{A}_{S^{i,u}}, z)]|]$.

5.1 Proof of Theorem 2

We start by introducing a ghost sample $\hat{S} = \{\hat{z}_1, \hat{z}_2, \cdots, \hat{z}_n\}$ drawn i.i.d according to distribution $\mathcal{D}$ and denote $R_{\hat{S}}(\mathbb{A}_S) = \sum_{i=1}^n c(\mathbb{A}_S, \hat{z}_i)/n$. It follows

$$E_{S\sim\mathcal{D}^n}[|R_S(\mathbb{A}_S) - R(\mathbb{A}_S)|] \le E_{S,\hat{S}\sim\mathcal{D}^{2n}}[|R(\mathbb{A}_S) - R_{\hat{S}}(\mathbb{A}_S)|] + E_{S,\hat{S}\sim\mathcal{D}^{2n}}[|R_S(\mathbb{A}_S) - R_{\hat{S}}(\mathbb{A}_S)|].$$

We bound the first term by $E_{S,\hat{S}\sim\mathcal{D}^{2n}}[|R(\mathbb{A}_S) - R_{\hat{S}}(\mathbb{A}_S)|] \le B/\sqrt{n}$ from Proposition 1 since $\mathbb{A}_S$ is independent of $\hat{S}$. For the second term, using the Jensen's inequality,

$$E_{S,\hat{S}\sim\mathcal{D}^{2n}}[|R_S(\mathbb{A}_S) - R_{\hat{S}}(\mathbb{A}_S)|] \le \sqrt{E_{S,\hat{S}\sim\mathcal{D}^{2n}}[(R_S(\mathbb{A}_S) - R_{\hat{S}}(\mathbb{A}_S))^2]}.$$

To bound this expression, we introduce a random permutation which swaps elements between S and $\hat{S}$, i.e., a permutation σ on $\{1, 2, \cdots, n, \hat{1}, \hat{2}, \cdots, \hat{n}\}$ s.t. $\{\sigma(i), \sigma(\hat{i})\} = \{i, \hat{i}\}$. Denote by S^σ and $\hat{S}^\sigma$ the permuted samples of S and $\hat{S}$, respectively, and define z_i^σ and $\hat{z}_i^\sigma$ in an obvious way. Since S and $\hat{S}$ are chosen i.i.d according to distribution $\mathcal{D}$, $E_{S,\hat{S}\sim\mathcal{D}^{2n}}[(R_S(\mathbb{A}_S) - R_{\hat{S}}(\mathbb{A}_S))^2]$ equals to

$$\begin{aligned}&E_{S,\hat{S}\sim\mathcal{D}^{2n}}\Big[\sum\nolimits_\sigma \big(R_{S^\sigma}(\mathbb{A}_{S^\sigma}) - R_{\hat{S}^\sigma}(\mathbb{A}_{S^\sigma})\big)^2/2^n\Big]\\ &= \frac{1}{n^2 2^n} E_{S,\hat{S}\sim\mathcal{D}^{2n}}\Big[\sum_{i,j,\sigma} \big(c(\mathbb{A}_{S^\sigma}, z_i^\sigma) - c(\mathbb{A}_{S^\sigma}, \hat{z}_i^\sigma)\big)\big(c(\mathbb{A}_{S^\sigma}, z_j^\sigma) - c(\mathbb{A}_{S^\sigma}, \hat{z}_j^\sigma)\big)\Big].\end{aligned}$$

Given σ and i, we define two permutations σ_1 and σ_2 as follows: $\sigma_1(i) = i$, $\sigma_1(\hat{i}) = \hat{i}$, $\sigma_2(i) = \hat{i}$, $\sigma_2(\hat{i}) = i$ and $\sigma_1(k) = \sigma_2(k) = \sigma(k)$ for $k \ne i, \hat{i}$. It holds

$$\begin{aligned}&\sum\nolimits_\sigma \big(c(\mathbb{A}_{S^\sigma}, z_i^\sigma) - c(\mathbb{A}_{S^\sigma}, \hat{z}_i^\sigma)\big)\big(c(\mathbb{A}_{S^\sigma}, z_j^\sigma) - c(\mathbb{A}_{S^\sigma}, \hat{z}_j^\sigma)\big)\\ &= \sum\nolimits_{\sigma_1} \big(c(\mathbb{A}_{S^{\sigma_1}}, z_i^{\sigma_1}) - c(\mathbb{A}_{S^{\sigma_1}}, \hat{z}_i^{\sigma_1})\big)\big(c(\mathbb{A}_{S^{\sigma_1}}, z_j^{\sigma_1}) - c(\mathbb{A}_{S^{\sigma_1}}, \hat{z}_j^{\sigma_1})\big)/2\\ &\quad + \sum\nolimits_{\sigma_2} \big(c(\mathbb{A}_{S^{\sigma_2}}, z_i^{\sigma_2}) - c(\mathbb{A}_{S^{\sigma_2}}, \hat{z}_i^{\sigma_2})\big)\big(c(\mathbb{A}_{S^{\sigma_2}}, z_j^{\sigma_2}) - c(\mathbb{A}_{S^{\sigma_2}}, \hat{z}_j^{\sigma_2})\big)/2\\ &= \sum (\Theta_{ij} + \Delta_{ij})/2\end{aligned}$$

where $\Theta_{ij} = \chi_3(\chi_4 - \chi_2)$ and $\Delta_{ij} = \chi_2(\chi_1 + \chi_3)$ with $\chi_1 = c(\mathbb{A}_{S^{\sigma_1}}, z_i^{\sigma_1}) - c(\mathbb{A}_{S^{\sigma_1}}, \hat{z}_i^{\sigma_1})$, $\chi_2 = c(\mathbb{A}_{S^{\sigma_1}}, z_j^{\sigma_1}) - c(\mathbb{A}_{S^{\sigma_1}}, \hat{z}_j^{\sigma_1})$, $\chi_3 = c(\mathbb{A}_{S^{\sigma_2}}, z_i^{\sigma_2}) - c(\mathbb{A}_{S^{\sigma_2}}, \hat{z}_i^{\sigma_2})$ and $\chi_4 = c(\mathbb{A}_{S^{\sigma_2}}, z_j^{\sigma_2}) - c(\mathbb{A}_{S^{\sigma_2}}, \hat{z}_j^{\sigma_2})$. Noting σ_1 and σ_2 are independent of j,

$$\begin{aligned}E_{S,\hat{S}}\Big[\sum\nolimits_{i=1}^n \sum\nolimits_{j=1}^n \Theta_{ij}/n^2\Big] &= E_{S,\hat{S}}\Big[\sum\nolimits_{i=1}^n \big(c(\mathbb{A}_{S^{\sigma_2}}, z_i^{\sigma_2}) - c(\mathbb{A}_{S^{\sigma_2}}, \hat{z}_i^{\sigma_2})\big)\\ &\quad \times \big(R_{S^{\sigma_2}}(\mathbb{A}_{S^{\sigma_2}}) - R_{S^{\sigma_1}}(\mathbb{A}_{S^{\sigma_1}}) + R_{\hat{S}^{\sigma_1}}(\mathbb{A}_{S^{\sigma_1}}) - R_{\hat{S}^{\sigma_2}}(\mathbb{A}_{S^{\sigma_2}})\big)/n\Big].\end{aligned}$$

Since $|c(\mathbb{A}_S, z)| \le B$ and $\mathbb{A}$ has approximation stability $(\beta_1(n), \beta_2(n))$, we obtain $E_{S,\hat{S}\sim\mathcal{D}^{2n}}[|R_{S^{\sigma_1}}(\mathbb{A}_{S^{\sigma_1}}) - R_{S^{\sigma_2}}(\mathbb{A}_{S^{\sigma_2}})|] \le \beta_1(n)$ and

$$\begin{aligned} E_{S,\hat{S}\sim\mathcal{D}^{2n}}[|R_{\hat{S}^{\sigma_2}}(\mathbb{A}_{S^{\sigma_2}}) - R_{\hat{S}^{\sigma_1}}(\mathbb{A}_{S^{\sigma_1}})|] &\le E_{S,\hat{S}\sim\mathcal{D}^{2n}}[|R_{\hat{S}^{\sigma_2}}(\mathbb{A}_{S^{\sigma_2}}) - R(\mathbb{A}_{S^{\sigma_2}})|] \\ + E_{S,\hat{S}\sim\mathcal{D}^{2n}}[|R_{\hat{S}^{\sigma_1}}(\mathbb{A}_{S^{\sigma_1}}) - R(\mathbb{A}_{S^{\sigma_1}})|] &+ E_{S,\hat{S}\sim\mathcal{D}^{2n}}[|R(\mathbb{A}_{S^{\sigma_1}}) - R(\mathbb{A}_{S^{\sigma_2}})|] \\ &\le 2\beta_2(n) + 2B/\sqrt{n}, \quad (1) \end{aligned}$$

from Proposition 1 and Proposition 3. Thus we show

$$\left| E_{S,\hat{S}}\Big[\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\Theta_{ij}\Big]\right| \le 2B\beta_1(n) + 4B\beta_2(n) + 4B^2/\sqrt{n}. \quad (2)$$

For $E_{S,\hat{S}}[\sum_{i=1}^{n}\sum_{j=1}^{n}\Delta_{ij}/n^2]$, we also have

$$\begin{aligned} E_{S,\hat{S}}\Big[\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\Delta_{ij}\Big] &= E_{S,\hat{S}}\Big[\frac{1}{n}\sum_{i=1}^{n}\big(R_{S^{\sigma_1}}(\mathbb{A}_{S^{\sigma_1}}) - R_{\hat{S}^{\sigma_1}}(\mathbb{A}_{S^{\sigma_1}})\big) \\ &\times \big(c(\mathbb{A}_{S^{\sigma_1}}, z_i^{\sigma_1}) - c(\mathbb{A}_{S^{\sigma_1}}, \hat{z}_i^{\sigma_1}) + c(\mathbb{A}_{S^{\sigma_2}}, z_i^{\sigma_2}) - c(\mathbb{A}_{S^{\sigma_2}}, \hat{z}_i^{\sigma_2})\big)\Big]. \end{aligned}$$

This expression could not be summed directly since σ_1 and σ_2 are dependent on i. But from symmetry and i.i.d assumption, we have

$$\begin{aligned} E_{S,\hat{S}}\Big[\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\Delta_{ij}\Big] &= E_{S,\hat{S}}[\big(R_{S^{\sigma_1^*}}(\mathbb{A}_{S^{\sigma_1^*}}) - R_{\hat{S}^{\sigma_1^*}}(\mathbb{A}_{S^{\sigma_1^*}})\big) \\ &\times \big(c(\mathbb{A}_{S^{\sigma_1^*}}, z_1) - c(\mathbb{A}_{S^{\sigma_1^*}}, \hat{z}_1) + c(\mathbb{A}_{S^{\sigma_2^*}}, \hat{z}_1) - c(\mathbb{A}_{S^{\sigma_2^*}}, z_1)\big)], \end{aligned}$$

where $\sigma_1^*(1) = 1$, $\sigma_1^*(\hat{1}) = \hat{1}$, $\sigma_2^*(1) = \hat{1}$, $\sigma_2^*(\hat{1}) = 1$ and $\sigma_1^*(k) = \sigma_2^*(k) = \sigma(k)$ for $k \ne 1, \hat{1}$. Let $z, \hat{z}$ be two new examples and set $S_1 = S^{1,z}$, $\hat{S}_1 = \hat{S}^{1,\hat{z}}$. In a similar way to prove Eq.(2), we have

$$\begin{aligned} E_{S,\hat{S},z,\hat{z}}[|R_{S^{\sigma_1^*}}(\mathbb{A}_{S^{\sigma_1^*}}) - R_{\hat{S}^{\sigma_1^*}}(\mathbb{A}_{S^{\sigma_1^*}}) - R_{S_1^{\sigma_1^*}}(\mathbb{A}_{S_1^{\sigma_1^*}}) + R_{\hat{S}_1{}^{\sigma_1^*}}(\mathbb{A}_{S_1^{\sigma_1^*}})|] \\ \le \beta_1(n) + 2\beta_2(n) + 2B/\sqrt{n}. \quad (3) \end{aligned}$$

Since z_1 and $\hat{z}_1$ are independent to S_1 and $\hat{S}_1$, it holds

$$\begin{aligned} &\Big|E_{S,\hat{S},z,\hat{z}}\Big[\big(c(\mathbb{A}_{S^{\sigma_1^*}}, z_1) - c(\mathbb{A}_{S^{\sigma_1^*}}, \hat{z}_1) + c(\mathbb{A}_{S^{\sigma_2^*}}, \hat{z}_1) - c(\mathbb{A}_{S^{\sigma_2^*}}, z_1)\big)\times \\ &\big(R_{S_1^{\sigma_1^*}}(\mathbb{A}_{S_1^{\sigma_1^*}}) - R_{\hat{S}_1{}^{\sigma_1^*}}(\mathbb{A}_{S_1^{\sigma_1^*}})\big)\Big]\Big| = \Big|E_{S^1,\hat{S}^1,z,\hat{z}}\Big[\big(R_{S_1^{\sigma_1^*}}(\mathbb{A}_{S_1^{\sigma_1^*}}) - R_{\hat{S}_1{}^{\sigma_1^*}}(\mathbb{A}_{S_1^{\sigma_1^*}})\big) \\ &\times \big(E_{z_1\sim\mathcal{D}}[c(\mathbb{A}_{S^{\sigma_1^*}}, z_1) - c(\mathbb{A}_{S^{\sigma_2^*}}, z_1)] + E_{\hat{z}_1\sim\mathcal{D}}[c(\mathbb{A}_{S^{\sigma_2^*}}, \hat{z}_1) - c(\mathbb{A}_{S^{\sigma_1^*}}, \hat{z}_1)]\big)\Big]\Big| \\ &\le 4B\beta_2(n). \end{aligned}$$

Thus we derive $\left|E_{S,\hat{S}}\left[\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\Delta_{ij}\right]\right| \le 4B(\beta_1(n) + 3\beta_2(n) + 2B/\sqrt{n})$ from Eq.(3), which yields $E_{S,\hat{S}\sim\mathcal{D}^{2n}}[(R_S(\mathbb{A}_S) - R_{\hat{S}}(\mathbb{A}_S))^2] \le 3B\beta_1(n)/2 + 4B\beta_2(n) + 3B^2/\sqrt{n}$ from Eq.(2). This theorem follows. □

5.2 Proof of Theorem 3

The equivalence between the universal consistency and universal generalization has been established in [22]. Combining Lemma 1 and Theorem 2 proves that $\mathbb{A}$ generalizes with rate $\epsilon_{\text{gen}}(n) = \sqrt{3\epsilon_{\text{erm}}(n)B + 4\beta_2(n)B + 3B^2/\sqrt{n} + 3B^2/n} + B/\sqrt{n}$ if $\mathbb{A}$ is AERM with rate $\epsilon_{\text{erm}}(n)$ and has expected validation stability $\beta_2(n)$. Thus Theorem 3 follows from the following lemma.

Lemma 4 (AERM + generalization + consistency $\Rightarrow$ Expected validation stability). *Suppose $\mathbb{A}$ is consistent with rate $\epsilon_{cons}(n)$, generalized with rate $\epsilon_{gen}(n)$, and AERM with rate $\epsilon_{erm}(n)$ such that $n\epsilon_{erm}(n) \to 0$ as $n \to \infty$. Then $\mathbb{A}$ has expected validation stability $\beta(n) = \epsilon_{gen} + 4\epsilon_{cons} + 2n\epsilon_{erm}(n)$.*

Proof. For $i \in [n]$, since $|E_{u\sim\mathcal{D}}[c(\mathbb{A}_S, u) - c(\mathbb{A}_{S^{i,u}}, u)]| \leq |E_{u\sim\mathcal{D}}[c(\mathbb{A}_S, u) - E_{z\sim\mathcal{D}}[c(\mathbb{A}_{S^{i,z}}, u)]]| + |E_{u\sim\mathcal{D}}[E_{z\sim\mathcal{D}}[c(\mathbb{A}_{S^{i,z}}, u)] - c(\mathbb{A}_{S^{i,u}}, u)]|$, we have

$$\begin{aligned}
&E_{S\sim\mathcal{D}^n}[|E_{u\sim\mathcal{D}}[c(\mathbb{A}_S, u) - c(\mathbb{A}_{S^{i,u}}, u)]|] \\
&\qquad \leq E_{S,z\sim\mathcal{D}^{n+1}}[|E_{u\sim\mathcal{D}}[c(\mathbb{A}_S, u) - c(\mathbb{A}_{S^{i,z}}, u)]|]+ \\
&\qquad E_{S\sim\mathcal{D}^n}[|E_{u,z\sim\mathcal{D}^2}[c(\mathbb{A}_{S^{i,z}}, u) - c(\mathbb{A}_{S^{i,u}}, u) + c(\mathbb{A}_{S^{i,u}}, z) - c(\mathbb{A}_{S^{i,z}}, z)]|]/2.
\end{aligned}$$

For the first term, we can easily upper bound

$$\begin{aligned}
E_{S,z\sim\mathcal{D}^{n+1}}[|E_{u\sim\mathcal{D}}[c(\mathbb{A}_S, u) - c(\mathbb{A}_{S^{i,z}}, u)]|] &\leq E_{S,z\sim\mathcal{D}^{n+1}}[|R(\mathbb{A}_S) - R(h^*)|] \\
&+ E_{S,z\sim\mathcal{D}^{n+1}}[|R(h^*) - R(\mathbb{A}_{S^{i,z}})|] \leq 2\epsilon_{\text{cons}}(n) \quad (4)
\end{aligned}$$

from the consistency of $\mathbb{A}_S$. For the second term, it holds

$$\begin{aligned}
&E_{S\sim\mathcal{D}^n}[|E_{u,z\sim\mathcal{D}^2}[c(\mathbb{A}_{S^{i,z}}, u) - c(\mathbb{A}_{S^{i,u}}, u) + c(\mathbb{A}_{S^{i,u}}, z) - c(\mathbb{A}_{S^{i,z}}, z)]|] = \\
&nE_{S\sim\mathcal{D}^n}[|E_{u,z\sim\mathcal{D}^2}[R_{S^{i,u}}(\mathbb{A}_{S^{i,z}}) - R_{S^{i,u}}(\mathbb{A}_{S^{i,u}}) + R_{S^{i,z}}(\mathbb{A}_{S^{i,u}}) - R_{S^{i,z}}(\mathbb{A}_{S^{i,z}})]|] \\
&\qquad = 2nE_{S\sim\mathcal{D}^n}[|E_{u,z\sim\mathcal{D}^2}[R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})] - R_{S^{i,u}}(\mathbb{A}_{S^{i,z}})|].
\end{aligned}$$

We will use Proposition 2 to bound the above expression. It holds

$$\begin{aligned}
&|E_{S\sim\mathcal{D}^n}[E_{u,z\sim\mathcal{D}^2}[R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})] - R_{S^{i,u}}(\mathbb{A}_{S^{i,z}})]| \\
&\qquad = |E_{S,u,z\sim\mathcal{D}^{n+2}}[c(\mathbb{A}_{S^{i,u}}, u) - c(\mathbb{A}_{S^{i,z}}, u)]|/n.
\end{aligned}$$

Meanwhile, it is easy to obtain $E_{S,u\sim\mathcal{D}^{n+1}}[c(\mathbb{A}_{S^{i,u}}, u)] = E_{S,u\sim\mathcal{D}^{n+1}}[R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})]$ and $E_{S,u,z\sim\mathcal{D}^{n+2}}[c(\mathbb{A}_{S^{i,z}}, u)] = E_{S,z\sim\mathcal{D}^{n+1}}[R(\mathbb{A}_{S^{i,z}})]$ from symmetry and i.i.d assumption. This leads to

$$|E_{S\sim\mathcal{D}^n}[E_{u,z\sim\mathcal{D}^2}[R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})] - R_{S^{i,u}}(\mathbb{A}_{S^{i,z}})]| \leq \epsilon_{\text{gen}}/n + 2\epsilon_{\text{cons}}/n.$$

For ERM, $R_{S^{i,u}}(\mathbb{A}_{S^{i,u}}) - R_{S^{i,u}}(\mathbb{A}_{S^{i,z}}) \leq R_{S^{i,u}}(\mathbb{A}_{S^{i,u}}) - R_{S^{i,u}}(\hat{h}_{S^{i,u}})$ and

$$E_{S,u\sim\mathcal{D}^{n+1}}[|R_{S^{i,u}}(\mathbb{A}_{S^{i,u}}) - R_{S^{i,u}}(\hat{h}_{S^{i,u}})|] \leq \epsilon_{\text{erm}}(n).$$

By Proposition 2, we have

$$nE_{S\sim\mathcal{D}^n}[|E_{u,z\sim\mathcal{D}^2}[R_{S^{i,u}}(\mathbb{A}_{S^{i,u}})] - R_{S^{i,u}}(\mathbb{A}_{S^{i,z}})|] \leq \epsilon_{\text{gen}} + 2\epsilon_{\text{cons}} + 2n\epsilon_{\text{erm}}(n),$$

which concludes this lemma by combining with Eq.(4). □

5.3 Proofs of Theorems 5 and 6

The two proofs are relatively similar, and thus we only give the detail proof of Theorem 6. Set $\mathrm{err}(t) = \mathrm{err}_S^t$, $\mathrm{err}'(t) = \mathrm{err}_{S^{i,u}}^t$, $\alpha(t) = \alpha_S^t$, $\alpha'(t) = \alpha_{S^{i,u}}^t$, $h_t(x) = \mathbb{A}_{P_S^t}(x)$ and $h'_t(x) = \mathbb{A}_{P_{S^{i,u}}^t}(x)$ for short in this subsection. The following lemma establishes AdaBoost's empirical stability.

Lemma 5. *For any $u, S \in \mathcal{Z}^{n+1}$, any $i \in [n]$, any $t \in [T]$ and small $\gamma > 0$, if base learner satisfies $\gamma \le err_S^t$, $err_{S^{i,u}}^t \le 1-\gamma$ and $|err_S^t - err_{S^{i,u}}^t| \le \zeta(n)$, then the combined learner $\mathbb{H}_S(x)$ has empirical stability $\beta_1(n) = \zeta(n)T/\sqrt{\gamma(1-\gamma)}$.*

Proof. From [21] we derive

$$R_S(\mathbb{H}_S) = 2^T \prod\nolimits_{t=1}^T \sqrt{\mathrm{err}(t)(1-\mathrm{err}(t))}$$
$$R_{S^{i,u}}(\mathbb{H}_{S^{i,u}}) = 2^T \prod\nolimits_{t=1}^T \sqrt{\mathrm{err}'(t)(1-\mathrm{err}'(t))}.$$

Since $\gamma < \mathrm{err}(t) < 1-\gamma$, it is easy to get $\sqrt{\gamma(1-\gamma)} \le \sqrt{\mathrm{err}(t)(1-\mathrm{err}(t))} \le 1/2$, which leads to

$$\begin{aligned}&|\sqrt{\mathrm{err}(t)(1-\mathrm{err}(t))} - \sqrt{\mathrm{err}'(t)(1-\mathrm{err}'(t))}|\\ &\qquad = \frac{|\mathrm{err}(t)-\mathrm{err}'(t)| \times |1-\mathrm{err}(t)-\mathrm{err}'(t)|}{\sqrt{\mathrm{err}(t)(1-\mathrm{err}(t))} + \sqrt{\mathrm{err}'(t)(1-\mathrm{err}'(t))}} \le \frac{\zeta(n)/2}{\sqrt{\gamma(1-\gamma)}},\end{aligned}$$

and $R_S(\mathbb{H}_S) < 1$. Thus $|R_S(\mathbb{H}_S) - R_{S^{i,u}}(\mathbb{H}_{S^{i,u}})|$ is bounded by

$$\begin{aligned}&2^{T-1}\Big|\prod_{t=1}^{T-1} \sqrt{\mathrm{err}(t)(1-\mathrm{err}(t))} - \prod_{t=1}^{T-1} \sqrt{\mathrm{err}'(t)(1-\mathrm{err}'(t))}\Big|\\ &+ 2^T\Big|\sqrt{\mathrm{err}(T)(1-\mathrm{err}(T))} - \sqrt{\mathrm{err}'(T)(1-\mathrm{err}'(T))}\Big| \prod_{t=1}^{T-1} \sqrt{\mathrm{err}(t)(1-\mathrm{err}(t))}\\ &\le 2^{T-1}\Big|\prod_{t=1}^{T-1} \sqrt{\mathrm{err}(t)(1-\mathrm{err}(t))} - \prod_{t=1}^{T-1} \sqrt{\mathrm{err}'(t)(1-\mathrm{err}'(t))}\Big| + \frac{\zeta(n)}{\sqrt{\gamma(1-\gamma)}}\end{aligned}$$

which leads to $|R_S(\mathbb{H}_S) - R_{S^{i,u}}(\mathbb{H}_{S^{i,u}})| \le \zeta(n)T/\sqrt{\gamma(1-\gamma)}$ as desired. □

Lemma 6. *If $h_t(x), h'_t(x)$ are two binary learners with cost function $c(h,z) = I[h(x) \ne y]$, then we have $E_{z\sim\mathcal{D}}[|h_t(x) - h'_t(x)|] = E_{z\sim\mathcal{D}}[|c(h_t,z) - c(h'_t,z)|]$.*

This lemma holds from the fact $|I[h_t(x) \ne y] - I[h'_t(x) \ne y]| = |h_t(x) - h'_t(x)|$. The following lemma establishes the validation stability of AdaBoost.

Lemma 7. *For any $u, S \in \mathcal{Z}^{n+1}$, any $i \in [n]$, any $t \in [T]$ and $0 < \gamma < 1/2$, if the base learner satisfies $\gamma \le err_S^t$, $err_{S^{i,u}}^t \le 1-\gamma$, $|err_S^t - err_{S^{i,u}}^t| \le \zeta(n)$ and $E_{u\sim\mathcal{D}}[|c(\mathbb{A}_S, u) - c(\mathbb{A}_{S^{i,u}}, u)|] \le \eta(n)$, then $\mathbb{H}_S(x)$ has validation stability $\beta_2(n) = TB\big(\frac{\eta(n)}{2}\ln\frac{1-\gamma}{\gamma} + \frac{\zeta(n)}{2\gamma(1-\gamma)}\big)$.*

Proof. We first set $u = (x, y)$. From mean value theorem and $\gamma \leq \mathrm{err}(t), \mathrm{err}'(t) \leq 1 - \gamma$, we have $|\alpha(t) - \alpha'(t)| \leq \zeta(n)/(2\gamma(1-\gamma))$. It follows from Lemma 6 that

$$\begin{aligned} E_{u\sim\mathcal{D}}[|\alpha(t)h_t(x) - \alpha'(t)h'_t(x)|] &\leq E_{u\sim\mathcal{D}}[|\alpha(t)||h_t(x) - h'_t(x)|] \\ &+ E_{u\sim\mathcal{D}}[|h'_t(x)||\alpha(t) - \alpha'(t)|] \leq \frac{\eta(n)}{2}\ln\frac{1-\gamma}{\gamma} + \frac{\zeta(n)}{2\gamma(1-\gamma)}, \end{aligned} \quad (5)$$

Using mean value theorem again, we obtain

$$\begin{aligned} &|\exp(-y\alpha(t)h_t(x)) - \exp(-y\alpha'(t)h'_t(x))| \\ &\qquad \leq \sqrt{(1-\gamma)/\gamma}\,|\alpha(t)h_t(x) - \alpha'(t)h'_t(x)|\,. \end{aligned}$$

Combining with Eq.(5) gives

$$\begin{aligned} &|R(\mathbb{H}_S) - E_{u\sim\mathcal{D}}[R(\mathbb{H}_{S^{i,u}})]| \leq E_{u\sim\mathcal{D}}\left[|\exp(-y\alpha'(T)h'_T(x)) \times \varGamma|\right] + \\ &E_{u\sim\mathcal{D}}\Big[\Big|\exp(-y\sum_{t=1}^{T-1}\alpha(t)h_t(x))\big(\exp(-y\alpha(T)h_T(x)) - \exp(-y\alpha'(T)h'_T(x))\big)\Big|\Big] \\ &\quad \leq (B\eta(n)/2)\ln((1-\gamma)/\gamma) + B\zeta(n)/(2\gamma(1-\gamma)) + E_{u\sim\mathcal{D}}[|\varGamma|]\sqrt{(1-\gamma)/\gamma}, \end{aligned}$$

where $\varGamma = \exp\left(-y\sum_{t=1}^{T-1}\alpha(t)h_t(x)\right) - \exp\left(-y\sum_{t=1}^{T-1}\alpha'(t)h'_t(x)\right)$. This completes the proof by straight evaluation. $\square$

By Lemmas 5 and 7 we get that AdaBoost has empirical stability and validation stability, respectively. Thus, by using Theorem 4, we get Theorem 6.

Acknowledgements

We thank the anonymous reviewers for helpful comments. The work was supported by the National Science Foundation of China (60635020, 60721002), the National Fundamental Research Program of China (2010CB327903) and the Jiangsu Science Foundation (BK2008018).

References

[1] Alon, N., Ben-David, S., Cesa-Bianchi, N., Haussler, D.: Scale-sensitive dimensions, uniform convergence, and learnablity. J. ACM 44(4), 615–631 (1997)
[2] Bickel, J.P., Ritov, Y., Zakai, A.: Some theory for generalized boosting algorithms. J. Mach. Learn. Res. 7, 705–732 (2006)
[3] Bousquet, O., Elisseeff, A.: Stability and generalization. J. Mach. Learn. Res. 2, 499–526 (2002)
[4] Breiman, L.: Prediction games and arcing classifiers. Neural Comput. 11(7), 1493–1517 (1999)
[5] Devroye, L.P., Wagner, T.J.: Distribution-free performance bounds for potential function rules. IEEE Trans. Inform. Theory 25, 601–604 (1979)

[6] Freund, Y., Schapire, R.E.: Game theory, on-line prediction and boosting. In: Proc. of 9th COLT, Desenzano sul Garda, Italy, pp. 325–332 (1996)
[7] Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. 55(1), 119–139 (1997)
[8] Freund, Y., Schapire, R.E.: Response to "Evidence contrary to the statistical view of Boosting". J. Mach. Learn. Res. 9, 171–174 (2008)
[9] Friedman, J., Hastie, T., Tibshirani, R.: Addtive logistic regression: A statistical view of boosting (with discussion). Ann. Statist. 28, 337–407 (2000)
[10] Kearns, M., Ron, D.: Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. Neural Comput. 11, 1427–1453 (1999)
[11] Kutin, S., Niyogi, P.: The interaction of stability and weakness in Adaboost. Technical Report 30, Department of Computer Science, University of Chicago, Chicago, IL (2001)
[12] Kutin, S., Niyogi, P.: Almost-everywhere algorithmic stability and generalization error. In: Proc. of 18th UAI, Edmonton, Canada, pp. 275–282 (2002)
[13] Mason, L., Baxter, J., Bartlett, P.L., Freman, M.R.: Boosting algorithms as gradient descent. In: Solla, S.A., Leen, T.K., Müller, K.-R. (eds.) Advances in NIPS, vol. 12, pp. 512–518. MIT Press, Cambridge (1999)
[14] McDiarmid, C.: On the method of bounded differences. In: Surveys in Combinatorics, pp. 148–188. Cambridge University Press, Cambridge (1989)
[15] Mease, D., Wyner, A.: Evidence contrary to the statistical view of boosting with discussion. J. Mach. Learn. Res. 9, 131–201 (2008)
[16] Mukherjee, S., Niyogi, P., Poggio, T., Rifkin, R.: Learning theory: Stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimzation. Adv. Comput. Math. 25, 161–193 (2006)
[17] Rakhlin, A., Mukherjee, S., Poggio, T.: Stability results in learning theory. Anal. Appl. 4, 397–417 (2005)
[18] Rätsch, G., Onoda, T., Müller, K.R.: Soft margins for Adaboost. Mach. Learn. 42, 287–320 (2001)
[19] Reyzin, L., Schapire, R.E.: How boosting the margin can also boost classifier complexity. In: Proc. of 23rd ICML, Pittsburgh, PA, pp. 753–760 (2006)
[20] Rudin, C., Schapire, R.E., Daubechies, I.: Precise statements of convergence for adaboost and arc-gv. In: Proc. of AMS-IMS-SIAM Joint Summer Research Conference: Machine learning, Statistics and Discovery, Snowbird, Utah, pp. 131–145 (2007)
[21] Schapire, R., Freund, Y., Bartlett, P.L., Lee, W.: Boosting the margin: A new explanation for the effectives of voting methods. Ann. Statist. 26, 1651–1686 (1998)
[22] Shalev-Shwartz, S., Shamir, O., Srebro, N., Sridharan, K.: Learnability and stability in the general learning setting. In: Proc. of 22nd COLT, Montreal, Canada (2009)
[23] Shalev-Shwartz, S., Shamir, O., Srebro, N., Sridharan, K.: Stochastic convex optimation. In: Proc. of 22nd COLT, Montreal, Canada (2009)
[24] Vapnik, V.N.: Statistical Learning Theory. John Wiley & Sons, New York (1998)
[25] Wang, L., Sugiyama, M., Yang, C., Zhou, Z.-H., Feng, J.: On the margin explanation of boosting algorithm. In: Proc. of 21st COLT, Helsinki, Finland, pp. 479–490 (2008)

A Spectral Approach for Probabilistic Grammatical Inference on Trees*

Raphaël Bailly, Amaury Habrard, and François Denis

Laboratoire d'Informatique Fondamentale de Marseille,
UMR CNRS 6166, Aix-Marseille Université
CMI, 39 rue F. Joliot Curie, 13453 Marseille cedex 13, France
{raphael.bailly,amaury.habrard,francois.denis}@lif.univ-mrs.fr

Abstract. We focus on the estimation of a probability distribution over a set of trees. We consider here the class of distributions computed by weighted automata - a strict generalization of probabilistic tree automata. This class of distributions (called rational distributions, or rational stochastic tree languages - RSTL) has an algebraic characterization: All the residuals (conditional) of such distributions lie in a finite-dimensional vector subspace. We propose a methodology based on Principal Components Analysis to identify this vector subspace. We provide an algorithm that computes an estimate of the target residuals vector subspace and builds a model which computes an estimate of the target distribution.

1 Introduction

In this article, we focus on the problem of learning probability distributions over trees. This problem is motivated by the high need in XML applications or natural language processing to represent large tree sets by probabilistic models. From a machine learning standpoint, this problem can be formulated as follows. Given a sample of trees independently drawn according to an unknown distribution p, a classical problem is to infer an estimate of p in some class of probabilistic models [1]. This is a classical problem in grammatical inference and the objective here is to find a good estimate of the model's parameters. A usual class of models is the class of probabilistic tree automata (PTA) where the parameters lie in $[0, 1]$.

Recent approaches propose using a larger class of representation: the class of rational distributions (also called rational stochastic tree languages, or RSTL) that can be computed by weighted tree automata - with parameters in $\mathbb{R}$, hence with weights that can be negative and without any per state normalisation condition. This class has two interesting properties: It has a high level of expressiveness since it strictly includes the class of PTA and it admits a canonical form with a minimal number of parameters (see [2] for an illustration in the string case). It has notably the characterization that the residuals of a rational distribution (a special kind of conditional distributions) lie in a finite-dimensional subspace. This set of residuals spans a vector subspace W of the vector space of real values functions over trees. W is finite dimensional and its dimension

* This work was partially supported by the ANR LAMPADA ANR-09-EMER-007 project and by the IST Programme of the European Community, under the PASCAL2 Network of Excellence, IST-2007-216886. This publication only reflects the authors' views.

© Springer-Verlag Berlin Heidelberg 2010

corresponds to the minimal number of states needed by a weighted tree automaton to compute p. Thus, a goal of an inference algorithm might be to identify this subspace W. This was illustrated by the algorithm DEES [3, 4] which builds iteratively a weighted automaton computing an estimate of p. However, the iterative approach presented before suffers from the drawback to rely on statistical tests that are done on fewer and fewer examples when the structure grows.

In order to overcome this drawback, in this paper we investigate the possibility of using Principal Component Analysis (PCA) to identify the target vector subspace spanned by the residuals of a rational distribution, and then to build a representation from this subspace. PCA has already been used in grammatical inference for learning rational string distributions in [5], and in another framework in [6]. Another spectral approach was proposed in [7, 8] for learning a class of Hidden Markov Models (HMM) over sequences. In this paper, we show that considering the class of rational distributions offers a natural framework for applying PCA to identify the target residuals subspace. Moreover, we obtain a high gain of expressiveness since we are able to infer classes of distributions that can not be computed by PTA. This gain in expressiveness has unfortunately two main drawbacks: the class of rational distributions is not recursively enumerable and it is not decidable if a rational series defines a distribution [9]. In spite of these strong constraints, we give some asymptotic error bounds and provide pointwise convergence result.

The paper is organized as follows. Section 2 gives the preliminaries on trees and rational tree series. Section 3 is devoted to our algorithm, while the convergence properties are presented in Section 4. Some experiments are provided in the last section.

2 Preliminaries

In this section, we introduce the objects that will be used all along in the paper. We mainly follow notations and definitions from [10] about trees. Formal power tree series have been introduced in [11] where the main results appear. Some notations about norms and matrices terminate this section.

2.1 Trees and Contexts

Unranked Trees. Let F be an unranked alphabet. The set of *unranked trees* over F is the smallest set T_F satisfying $F \subseteq T_F$, and for any $f \in F$, and $t_1, \ldots, t_m \in T_F$, $f(t_1, \ldots, t_m) \in T_F$.

Ranked Trees. Let $F = F_0 \cup \cdots \cup F_n$ be a ranked alphabet where the elements in F_0 are also called constant symbols. The set of *trees* over F is the smallest set T_F satisfying $F_0 \subseteq T_F$, and for any $f \in F_k$, and any $t_1, \ldots, t_k \in T_F$, $f(t_1, \ldots, t_k) \in T_F$.

Any tree defined over an unranked alphabet F can be represented over a ranked alphabet $F^@ = F_2^@ \cup F_0^@$ with only one binary symbol @, i.e. $F_2^@ = \{@(\cdot, \cdot)\}$ where $@ \notin F$ and constants that comprise all symbols in F: $F_0^@ = F$. Figure 1(d) shows such a representation (called curryfication) of the tree of Figure 1(c). Curryfication can be formally defined by induction:

- $curry(f(t_1,\ldots,t_n)) = @(curry(f(t_1,\ldots,t_{n-1})), curry(t_n))$
- $curry(f(t)) = @(f, curry(t))$
- $curry(a) = a$ for $a \in F$

This particular class of ranked alphabet is in bijection with the set of *unranked trees* [10], i.e. labeled trees in which any node may have an unbounded number of children. Weighted automata on unranked trees are defined in [12], where it is proved that weighted unranked tree automata on F are equivalent to weighted tree automata on $F^{@}$. As ranked trees are a particular case of unranked trees and weighted ranked tree automata can be seen as a particular case of weighted unranked tree automata, the results still hold for any ranked alphabet.

Hence, without loss of generality, and in all the rest of the paper, we will only consider a ranked alphabet equipped with constant symbols and with only one binary symbol in the following of the paper. For convenience, we will use f for denoting the binary symbol instead of @.

Contexts. Contexts are element c of $C_F \subset T_{F\cup\{\$\}}$ where \$ is a variable that appears exactly once as a leaf in c (\$ is a constant and $\$ \notin F$). Given a context $c \in C_F$ and a tree $t \in T_F$, one can build a tree $c[t] \in T_F$ by replacing the (unique) occurrence of \$ in c by the tree t.

Example 1. Let $F_0 = \{a, b\}$, $F_1 = \{g(\cdot)\}$ and $F_2 = \{f(\cdot,\cdot)\}$. Then $t = f(a, g(b)) \in T_F$ (Figure 1(a)), $c = f(a, \$) \in C_F$ (Figure 1(b)) and $c[t] = f(f(a, g(b)), a)$ (Figure 1(c)).

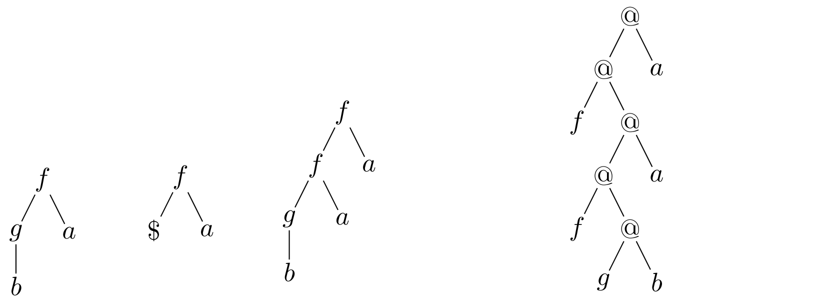


Fig. 1. An example of tree $t = f(a, g(b))$, context $c = f(\$, a)$ and their composition $c[t] = f(f(a, g(b)), a)$, as defined in Example 1. On the right a representation of t over an alphabet with only one binary symbol @ and with the elements of F seen as constant symbols.

Definition 1. *The* length *of a tree or a context is the number of functional symbols used to define it, including the special symbol* \$.

T_F^k (resp. $T_F^{\geq k}$) will denote the set of trees of length k (resp. length greater or equal than k).

2.2 Tree Series

A *(formal power) tree series* on T_F is a mapping $r : T_F \to \mathbb{R}$. The vector space of all tree series on T_F is denoted by $\mathbb{R}[T_F]$. We denote by $\ell_2(T_F)$ the vector subspace of $\mathbb{R}[T_F]$ of tree series r such that $\sum_{t \in T_F} r(t)^2 < \infty$. This vector subspace is equipped with a dot product $(r, s) = \sum_{t \in T_F} r(t)s(t)$.

Given $r \in \mathbb{R}[T_F]$, a residual of r is a series $s \in \mathbb{R}[T_F]$ such that $s(t) = r(c[t])$ for some $c \in C_F$. This series is denoted $\dot{c}r : t \mapsto r(c[t])$. One defines the set of residuals of r by $\{\dot{c}r | c \in C_F\}$. Let c_i be an enumeration of C_F, and t_j an enumeration of T_F. Given $r \in \mathbb{R}[T_F]$, one defines the (infinite) observation matrix X of r by: $(X)_{i,j} = r(c_i[t_j])$.

$$\begin{pmatrix} r(c_1[t_1]) \dots & r(c_1[t_j]) \dots \\ \vdots & \vdots \\ r(c_i[t_1]) \dots & r(c_i[t_j]) \dots \\ \vdots & \vdots \end{pmatrix}$$

Rational series (series computed by weighted automata) are the series with a finite rank observation matrix. The rank of the observation matrix is the rank of the rational series (i.e. the state number of a minimal automaton computing the series). From this observation, one can define a canonical linear representation of a rational tree series as introduced in [3]. We give here a simpler definition:

Definition 2. *The linear representation of a rational tree series over T_F is given by:*

- *the rank d of the series, and $\{q_1, \dots q_d\}$ a basis of $\mathbb{R}^d$.*
- *$\tau \in \mathbb{R}^d$.*
- *for each $a \in F_0$, a vector $\underline{a} \in \mathbb{R}^d$.*
- *for $f \in F_2$, a bilinear mapping $\underline{f} \in \mathcal{L}(\mathbb{R}^d, \mathbb{R}^d; \mathbb{R}^d)$.*

The linear representation is denoted by $(F, d, \underline{\ \ }, \tau)$.

The mapping $\underline{\ \ }$ can be inductively extended to a mapping $\underline{\ \ } : T_F \to \mathbb{R}^d$ that satisfies $\underline{f(t_1, t_2)} = \underline{f}(\underline{t_1}, \underline{t_2})$ for any $t_1, t_2 \in T_F$.

Finally, the value of $r(t)$ is given by: $r(t) = \underline{t}^\top \tau$ where $^\top$ denotes the transpose operator.

Example 2. Let $F = \{a, f(\cdot, \cdot)\}$ a ranked alphabet, consider the linear representation $(F, 2, \underline{\ \ }, \tau)$ of the series r such that $\{e_1, e_2\}$ is a basis of $\mathbb{R}^2$, $\tau = (1, 0)$ and defined by the following expressions:

$$\underline{a} = \tfrac{2e_1}{3} + \tfrac{e_2}{3}, \quad \underline{f}(e_1, e_2) = \tfrac{e_1}{3} + \tfrac{2e_2}{3}, \quad \underline{f}(e_i, e_j) = 0 \; for \; (i, j) \neq (1, 2).$$

One has:

$$r(f(a, a)) = \underline{f(a, a)}^\top \tau = \underline{f}(\underline{a}, \underline{a})^\top \tau = \underline{f}(\frac{2e_1}{3} + \frac{e_2}{3}, \frac{2e_1}{3} + \frac{e_2}{3})^\top \tau$$
$$= (\frac{2}{3}\frac{2}{3}\underline{f}(e_1, e_1) + \frac{2}{3}\frac{1}{3}\underline{f}(e_1, e_2) + \frac{1}{3}\frac{2}{3}\underline{f}(e_2, e_1) + \frac{1}{3}\frac{1}{3}\underline{f}(e_2, e_2))^\top \tau$$
$$= (\frac{2}{3}\frac{2}{3}(0, 0) + \frac{2}{3}\frac{1}{3}(\frac{1}{3}, \frac{2}{3}) + \frac{1}{3}\frac{2}{3}(0, 0) + \frac{1}{3}\frac{1}{3}(0, 0)) \cdot \tau = \frac{2}{3^3}.$$

Rational tree series can be equivalently represented by weighted tree automata where the number of states of the automata corresponds to the dimension of the linear representations. Indeed, a tree automaton is a tuple (Q, F, τ, δ) where Q, τ and δ are respectively the set of states, the terminal vector and the transition function. Let $(F, 2, \_, \tau)$ be a linear representation and let $(e_1, \ldots, e_d)$ be a basis of $\mathbb{R}^d$. Let $Q = \{e_1, \ldots, e_d\}$. Any linear relation of the form $\underline{f}(e_i, e_j) = \sum_k \alpha_{i,j}^k e_k$ yields to d transition rules of the form $f(e_i, e_j) \xrightarrow{\alpha_{i,j}^k} e_k$ and $\tau(e_i)$ is set to $\tau^\top e_i$. See [4, 13] for more details.

Example 3. A weighted automaton computing the series in example 2 would be $Q = \{q_1, q_2\}$, $F = \{a, f\}$, δ defined by:
$a \xrightarrow{2/3} q_1$, $a \xrightarrow{1/3} q_2$, $f(q_1, q_2) \xrightarrow{1/3} q_1$, $f(q_1, q_2) \xrightarrow{2/3} q_2$ and $\tau(q_1) = 1, \tau(q_2) = 0$.

Let $\mathbb{R}[C_F]$ be the set of mappings $s : C_F \to \mathbb{R}$. For $r \in \mathbb{R}[T_F]$ and $t \in T_F$, one can define $\bar{t}r \in \mathbb{R}[C_F]$ by:

$$\bar{t}r(c) = \dot{c}r(t) = r(c[t]).$$

The $\dot{c}r$ correspond to the rows of the observation matrix X and the $\bar{t}r$ to its columns, and one has the following equivalent properties:

1. $r \in \mathbb{R}[T_F]$ has an observation matrix X with finite rank d.
2. The vector subspace of $\mathbb{R}[T_F]$ spanned by $\{\dot{c}r | c \in C_F\}$ has dimension d.
3. The vector subspace of $\mathbb{R}[C_F]$ spanned by $\{\bar{t}r | t \in T_F\}$ has dimension d.

Let us denote by C_n the set of contexts of length lower than n, and let $\sim_{C_n}$ be the equivalence relation over $\mathbb{R}[C_F]$ defined by $f \sim_{C_n} g$ iff $\forall c \in C_n, f(c) = g(c)$. One defines $\mathbb{R}[C_n]$ as the quotient vector space $\mathbb{R}[C_F]/\sim_{C_n}$, equipped with the regular dot product $(f, g) = \sum_{c \in C_n} f(c)g(c)$.

2.3 Rational Distribution and Strong Consistency

Definition 3. *A* rational distribution *(or* rational stochastic tree language*, RSTL) over T_F is a rational series computing a probability distribution.*

In other words, a RSTL is a probability distribution that can be computed by a weighted automaton (or that admits a linear representation). It can be shown that there exists some rational distributions that cannot be computed by any *probabilistic tree automaton*. It is undecidable to know whether a rational series given by a linear representation defines a probability distribution (see [2] for an illustration in the string case).

Definition 4. *A* strongly consistent stochastic tree languages *(or* strongly consistent distribution*) over T_F is a probability distribution over T_F having a bounded average tree size i.e. $\sum_{t \in T_F} p(t)|t| < \infty$.*

It can be shown (see [4]) that, if p is a rational distribution having a bounded average tree size, there exists some constants $0 < C$ and $0 < \rho < 1$ such that:

$$\sum_{t \in T_F^{\geq k}} p(t) \leq C\rho^k.$$

3 Principle of the Algorithm

Let p be rational distribution on T_F (strongly consistent or not). We give first a general algorithm that takes a sample i.i.d. according to p as input. For this purpose, let C_n be the set of contexts of length lower that n and let us make the assumption that $\{\dot{c}p|c \in C_n\}$ and $\{\dot{c}p|c \in C_F\}$ span the same vector subspace of $\mathbb{R}[T_F]$. In other words, we suppose that considering the set C_n is sufficient to get the whole space of residuals.

Let V be the finite dimensional subspace of $\ell^2(T_F)$ spanned by the set $\{\dot{c}p|c \in C_n\}$. V^* will denote the set $\{\bar{t}p|_{C_n}, t \in T_F\} \subset \mathbb{R}[C_n]$ - for convenience we still denote by $\bar{t}p$ the mapping $\bar{t}p|_{C_n}$. Π_V denotes the orthogonal projection over V relatively to the dot product inherited from $\ell_2(T_F)$, and Π_{V^*} denotes the orthogonal projection over V^* relatively to the dot product inherited from $\mathbb{R}[C_n]$. Let S be a sample of N trees independently and identically drawn according to p and let p_S be the empirical distribution on T_F defined from S. V_S denotes the vector subspace of $\ell^2(T_F)$ spanned by $\{\dot{c}p_S|c \in C_n\}$, and V_S^* the subspace of $\mathbb{R}[C_n]$ spanned by $\{\bar{t}p_S|t \in T_F\}$.

We first build from S an estimate $V_{S,d}^*$ of V^* and then we show that $V_{S,d}^*$ can be used to build a linear representation such that its associated rational series approximate the target p. In this section, we implicitly suppose that the dimension d of V^* is known. We will show in the next section how it can be estimated from the data.

3.1 Estimating the Target Space

Let $d > 0$ be an integer. The first step consists in finding the d-dimensional vector subspace $V_{S,d}^*$ of V_S^* that minimizes the distance to $\{\bar{t}p_S|t \in T_F\}$:

$$V_{S,d}^* = \underset{dim(W^*)=d, W^* \subseteq V_S^*}{\arg\min} \sum_{t \in T_F} \|\bar{t}p_S - \Pi_{W^*}(\bar{t}p_S)\|^2.$$

$V_{S,d}^*$ can be computed using principal component analysis.

Let $\{t_j\}$ be an enumeration of T_F, and $\{c_i\}$ be an enumeration of C_n. Let X_S the empirical mean matrix defined by: $(X_S)_{i,j} = p_S(c_i[t_j])$, and let X be the expectation matrix defined by: $(X)_{i,j} = p(c_i[t_j])$.

$V_{S,d}^*$ corresponds to the vector subspace spanned by the d first (normalized) eigenvectors (corresponding to d largest eigenvalues) of the matrix $M_S = X_S X_S^\top$. N_S will denote the matrix $X_S^\top X_S$ which corresponds to the dual problem of the PCA. We will denote by $W^* = \{w_1^*, \ldots, w_d^*\}$ the set of eigenvectors (ordered by decreasing eigenvalues) of M_S, and by $W = \{w_1, \ldots, w_d\}$ the corresponding eigenvectors of N_S. W^* is the matrix with the vectors $\{w_1^*, \ldots, w_d^*\}$ as columns - this matrix corresponds to the projection operator Π_{V^*}, while W is the matrix with vectors $\{w_1, \ldots, w_d\}$ as columns, because both W and W^* are orthonormal.

Let $\lambda_1, \ldots, \lambda_d$ be the associated singular values; they also are the square roots of the eigenvalues of M_S.

We recall here the relationships between the w_i and w_i^* eigenvectors: $X_S w_i = \lambda_i w_i^*$ and $X_S^\top w_i^* = \lambda_i w_i$. In particular,

$$M_S w_i^* = X_S X_S^\top w_i^* = \lambda_i X_S w_i = \lambda_i^2 w_i^*.$$

3.2 Building the Linear Representation from the Dual Space

The eigenvectors found in the previous section form the basis of the residual space. In order to complete the linear representation, we now need to define, in the basis $\{w_1^*, \ldots, w_d^*\}$, the terminal vector τ, the mapping $\underline{\ }$ for the constant symbols $\underline{a}$ and the bi-linear operator $\underline{f}$.

The idea is to identify, for any tree t, the mapping $\bar{t}p_S$ to its projection on the space spanned by W^*, that is $W^* {W^*}^\top \bar{t}p_S$. We shall see in next section that this identification leads to a bounded error, decreasing as the size of the sample grows.

- The vector space is the space spanned by W^*.
- For each $a \in F_0$, $\underline{a} = {W^*}^\top \bar{a}p_S$.

In order to define $\underline{f}$, we use a known relation between eigenvectors of the standard and dual PCA: $w_i^* = \sum_k \frac{(w_i)_k}{\lambda_i} \bar{t_k} p_S$. We use the bilinearity of $\underline{f}$ to obtain:

- $\underline{f}(w_i^*, w_j^*) = \sum_{1 \leq k,l \leq d} \frac{(w_i)_k (w_j)_l}{\lambda_i \lambda_j} {W^*}^\top \overline{f(t_k, t_l)} p_S$.
- $\tau_i = w_i^*(\$)$, corresponding to the terminal weight of a tree in a bottom-up process.
- Finally, $r(t) = \underline{t}^\top . \tau$.

The different steps of the algorithm are described in Algorithm 1.

```
Data: A sample $S$ of trees in $T_F$ i.i.d. according to a distribution $p$, a dimension $d$ and a
      set of contexts $C_n$.
Result: A linear representation $A$ of a tree series $(F, d, \underline{\ }, \tau)$.
Let $X$ the matrix defined by $X[i,j] = p_S(c_i[t_j])$;
$M = XX^\top$ /* variance-covariance matrix */;
$(\lambda_i, w_i^*, w_i) \leftarrow$ square roots of eigenvalues of $M$ in decreasing order and corresponding
eigenvectors, and eigenvectors in the dual;
Let $w_1^*, \ldots, w_d^*$ be the eigenvectors corresponding to the $d$ largest eigenvalues and let
$W^* = [w_1^*, \ldots, w_d^*]$ be the matrix having the vectors $w_i^*$ as columns
Let $\underline{\ }$ be the operator defined by:
foreach $f \in \mathcal{F}$ do
    if $a \in \mathcal{F}_0$ then $\underline{a} = {W^*}^\top \bar{a}p_S$;
    if $f \in \mathcal{F}_2$ then $\underline{f}(w_i^*, w_j^*) = \sum_{1 \leq k,l \leq d} \frac{(w_i)_k (w_j)_l}{\lambda_i \lambda_j} {W^*}^\top \overline{f(t_k, t_l)} p_S$;
end
$(\tau)_i = w_i^*(\$)$ ;
return $A = (F, d, \underline{\ }, \tau)$;
```

Algorithm 1. Building a linear representation corresponding to a sample S and a dimension d

4 Consistency

Let us consider the observation matrix X defined by: $X_{ij} = p(c_i[t_j])$, where c_i and t_j are respectively contexts and trees. Let S be a sample of size N i.i.d. from p. X_S is defined as the empirical observation matrix built from the empirical distribution p_S. In this section, we will bound the difference between those two matrices, and show how it induces a bound for the convergence of the singular values and on the distance between the estimate and the target distribution for a tree t.

First, here is a simple result straightforward from the properties of empirical mean:

Lemma 1. *Let p a probability distribution over T_F, and p_S its empirical estimate from a sample of size N drawn i.i.d. from p, one has*

$$\mathbf{E}(\|p_S - p\|_2^2) = \sum_{t \in T_F} \mathbf{E}((p_S(t) - p(t))^2) = \sum_{t \in T_F} \frac{p(t)(p(t)-1)}{N} \leq \frac{1}{N}.$$

Let C_n be the set of contexts of length lower or equal than n, it can easily be shown that:

Lemma 2. *Let t be a tree. There is at most n contexts in C_n such that $t = c[t']$ for some tree t'.*

The previous lemma helps us to bound the occurrence number of a tree in the matrix X, which will allow us to use some concentration inequality to bound the error over X. One denotes $\|\|_F$ as the Frobenius norm on matrices, and $\Delta_X = \|X - X_S\|_F$.

Lemma 3. *Let X be a probability observation matrix restricted to the contexts belonging to C_n. Let X_S the empirical estimator of X from a sample S of size N. Then, one has with probability at least $1 - \delta$ ($\delta > 0$):*

$$\Delta_X = \|X - X_S\|_F \leq \sqrt{\frac{n}{N}} \left(1 + \sqrt{\log(\frac{1}{\delta})}\right).$$

Proof. This proof uses a construction similar to the proof of Proposition 19 in [8]. Let z be a discrete random variable that takes values in T_F. Let X be a probability observation matrix built from a set C_n of contexts as lines and trees from T_F as columns. One estimates X from N i.i.d. copies of z_i of z ($i = 1, \ldots, N$).

One associates to each variable z_i a matrix X_i indexed by contexts of C_n and trees of T_F such that

$$X_i[j,k] = 1 \text{ if } z_i = c_j[t_k] \text{ and 0 otherwise.}$$

From Lemma 2, X_i has at most n non null entries.

The empirical estimate of X is $X_S = \frac{1}{N}\sum_{i=1}^{N} X_i$. Our objective is to bound $\|X_S - X\|_F$.

Let S' be a sample that differs from S on at most one example z'_k.

Then,

$$|\|X_S - X\|_F - \|X_{S'} - X\|_F| \leq \|X_S - X_{S'}\|_F \leq \sqrt{\frac{2n}{N}}.$$

From McDiarmid inequality [14], one obtains:

$$Pr(\|X_S - X\|_F \geq \mathbf{E}(\|X_S - X\|_F) + \epsilon) \leq e^{-\frac{N}{n}\epsilon^2}.$$

By Lemma 1 and Lemma 2 and by using Jensen's inequality, it can be proved that $\mathbf{E}(\|X_S - X\|_F) \leq \sqrt{\frac{n}{N}}$. By fixing $\delta = e^{-\frac{N}{n}\epsilon^2}$, one gets the result. □

4.1 Singular Values Convergence

We use the previous result to show how one can assess the correct dimension of the target space. We will first recall some known result. Given an observation matrix X of rank d in the target space, and given its empirical estimate X_S, we can rewrite X_S as a sum $X + E$ where E models the sampling error. We have the following result from [15].

Lemma 4. *(Theorem 4.11 in [15]). Let $X \in \mathbb{R}^{m\times n}$ with $m \geq n$, and let $X_S = X + E$. If the singular values of X and X_S are ($\lambda_1 > \ldots > \lambda_n$) and ($\lambda_{S,1} > \ldots > \lambda_{S,n}$) respectively, then*

$$|\lambda_{S,i} - \lambda_i| \leq \|E\|_2, i = 1, \ldots, n.$$

Applied to our situation, this provides a valid way to assess the target dimension: let d be the rank of the target rational series, X_S be the observation matrix deduced from a sample S, $|S| = N$.

Theorem 1. *Let Λ be the set of singular values of X_S. Let Λ_s be the subset of singular values of X_S greater than s. For a given confidence parameter δ, let $d' = |\Lambda_s|$ for $s = \sqrt{\frac{n}{N}}(1 + \sqrt{\log(\frac{1}{\delta})})$. With probability greater than $1 - \delta$, one has $d \geq d'$.*

Proof. Straightforward from Lemma 3 and Lemma 4: with probability greater than $1 - \delta$, the singular values in Λ_s match non-zeros singular values from the target observation matrix X. □

Theorem 2. *Let λ_d the smallest non-zero eigenvalue of X. Let Λ be the set of singular values of X_S. Let Λ_s be the subset of singular values of X_S greater than s. For a given confidence parameter δ, let $d' = |\Lambda_s|$ for $s = \sqrt{\frac{n}{N}}(1 + \sqrt{\log(\frac{1}{\delta})})$. Suppose that*

$$N > \frac{4n}{\lambda_d^2}\left(1 + \sqrt{\log(\frac{1}{\delta})}\right)^2$$

Then, with probability greater than $1 - \delta$, one has $d = d'$.

Proof. The condition $N > \frac{4n}{\lambda_d^2}(1 + \sqrt{\log(\frac{1}{\delta})})^2$ implies that $s < \frac{\lambda_d}{2}$, thus the corresponding singular value $\lambda_{S,d}$ from X_S satisfies $\lambda_{S,d} > 2s - \|X - X_S\|_2$. This quantity is greater than s with probability at least $1 - \delta$. □

4.2 Bounds for the Estimation Error

We suppose here that the correct dimension has been found. We will not provide exact bounds, but only asymptotic bounds, and we will often use the equivalence between norms of vectors and matrices - since the vector spaces considered are finite-dimensional. Let us first introduce some notations corresponding to errors over the objects handled by our algorithm:

- $\Delta_x = \max \|x - x_S\|_2$ with x (resp. x_S) a row or a column of the observation matrix X (resp. X_S).
- $\Delta_v = \max \|w - w_S\|_2$ with w (resp. w_S) a left singular vector of the observation matrix X (resp. X_S).
- $\Delta_\lambda = \max \|\lambda - \lambda_S\|_2$ with λ (resp. λ_S) a singular value of the observation matrix X (resp. X_S).
- $\Delta_\Pi = \|WW^T - W_S W_S^T\|_F$ with W (resp. W_S) the d first singular vectors of the observation matrix X (resp. X_S).

Lemma 5. *$\Delta_\lambda < \Delta_X$ and $\Delta_x < \Delta_X$.*

Proof. Straightforward from Lemma 4 and the norm relation $\|\|_2 \leq \|\|_F$ for the first inequality, and the definition of Δ_x and Δ_X for the second. □

The following corollary gives an asymptotic bound on the error of the covariance matrix used to compute the eigenvectors.

Corollary 1. *$\Delta_M = \|M - M_S\|_F$. One has $\|M - M_S\|_F \leq O(\Delta_X)$*

Proof. One has $\|M - M_S\|_F \leq \|XX^\top - XX_S^\top + XX_S^\top - X_S X_S^\top\|_F \leq (\|X\|_F + \|X_S^\top\|_F)\Delta_X$. Thus:

$$\Delta_M \leq \Delta_X (2\|X\|_F + \Delta_X).$$ □

In order to provide asymptotic bounds on the other errors, we need to introduce some known results about eigenvectors and PCA from [16]. Let A be a symmetric positive Hilbert-Schmidt operator with positive eigenvalues[1] $\lambda_1^2 > \cdots > \lambda_d^2 > 0$. $\delta_r = \frac{1}{2}(\lambda_r^2 - \lambda_{r+1}^2)$, and let $\tilde{\delta_r} = \inf(\delta_r, \delta_{r-1})$. Let B be a symmetric positive Hilbert-Schmidt operator such that $\|B\|_F < \tilde{\delta_r}/2$ and that $\|B\|_F < \delta_d/2$. The results from [16] provide error bounds on projection operators and eigenvectors. In our framework, A corresponds to the covariance matrix M and $A + B$ to the empirical one M_S. Let W (resp. W_S) be the matrix of the d first eigenvectors of A (resp. $A + B$), and w_r (resp. $w_{S,r}$) the corresponding r-th eigenvector, we have the following results.

Lemma 6. *(Theorem 2 - remark of [16])* $\|w_r - w_{S,r}\|_2 \leq \frac{2\|B\|_F}{\tilde{\delta_r}}$.

Theorem 3. *(Theorem 3 of [16])* $\|WW^\top - W_S W_S^\top\|_F \leq \frac{\|B\|_F}{\delta_d}$.

We are now able to provide asymptotic bounds for the two remaining errors.

[1] Recall that according to our notation the λ_i denote singular values.

Lemma 7. *One has $\Delta_v = O(\Delta_X)$ and $\Delta_\Pi = O(\Delta_X)$.*

Proof. By using respectively Lemma 6 and Theorem 3, and from Corollary 1, one has

$$\Delta_v \leq \frac{4\Delta_X(2\|X\|_F + \Delta_X)}{\tilde{\delta_d}} = O(\Delta_X)$$

and

$$\Delta_\Pi \leq \frac{\Delta_X(2\|X\|_F + \Delta_X)}{\tilde{\delta_d}} = O(\Delta_X).$$

□

Let us denote $\lambda = \inf_{1\dots d} \lambda_i$. We will now study some errors on the parameters of the linear representation built by our algorithm. Let $p = (F, d, \underline{\ }, \tau)$ be the target linear representation equipped with the basis $\{w_1^*, \dots, w_d^*\}$ and let $r_S = (F, d, \underline{\ }_S, \tau_S)$ the linear representation equipped with the basis $\{w_{S,1}^*, \dots, w_{S,d}^*\}$ found by our algorithm from a sample S. Let us define the following error bounds on the coefficients:

- $\Delta_\tau = \sup_i(\tau - \tau_S)_i$,
- $\Delta_{\underline{a}} = \sup_i(\underline{a} - \underline{a}_S)_i$ for $a \in F_0$,
- $\Delta_{\underline{f}} = \sup_{i,j,k}(\underline{f}(w_j^*, w_k^*) - \underline{f}_S(w_{S,j}^*, w_{S,k}^*))_i$.

One can check the following lemma.

Lemma 8

$$\begin{aligned}
&\Delta_\tau \leq dO(\Delta_v) \leq 2d\|X\|_F O(\Delta_X),\\
&\Delta_{\underline{a}} \leq 2O(\Delta_X),\\
&\Delta_{\underline{f}} \leq \frac{O(\Delta_X)}{\lambda}\left[\|w_i w_j^\top\|_F(2 + 2\frac{\|X\|_F}{\delta} + 8\frac{\|X\|_F}{\delta}(\|w_i\|_1 + \|w_j\|_1))\right].
\end{aligned}$$

All the mappings considered through the algorithm are continuous, thus the mapping deduced from the algorithm converges pointwisely towards the target distribution as Δ_X tends to zero. We provide then the following result which gives a bound for the estimation error.

Theorem 4. *Let p be a rational distribution with rank d. Let n be the maximum length of a context used in the algorithm. If S is a sample i.i.d. from p, let r_S be the mapping deduced from the algorithm. There exists C such that for any $0 < \delta < 1$, for any tree t of length k, if S is an i.i.d. sample of size N then, with confidence at least $1 - \delta$, one has:*

$$|r_s(t) - p(t)| < Ckd^{2k}\sqrt{\frac{n}{N}\log(\frac{1}{\delta})}.$$

Proof. Let us prove the statement by induction on k. Let us denote Δ_k a bound for the error made for the estimate the coefficient of $\underline{t}$ for a tree t of height k. One has:

$$\Delta_1 = \Delta_{\underline{a}} = O(\Delta_X)$$

Using $\underline{f(u,v)} = \sum_{1\le i\le d}\sum_{1\le i\le d}(\underline{u})_i(\underline{v})_j \underline{f(w_i^*, w_j^*)}$, with $|u|+|v| = k-1$, one has:

$$\Delta_k = O(d^2(|u|d^{2|u|} + |v|d^{2|v|} + 1)\Delta_X) \le O(kd^{2k}\Delta_X) = O(kd^{2k}\sqrt{\frac{n}{N}\log(\frac{1}{\delta})}).$$

Then, since $p(t) = \underline{t}^T.\tau$ and $r_s(t) = \underline{t}_S^T.\tau_S$, one has the conclusion. $\square$

4.3 Strongly Convergent Case

In the case of strongly consistent distribution, one does not need to consider a finite set of contexts to perform the algorithm: one has, with confidence greater than $1-\delta$, $|t| < \frac{\log(2C/\delta)}{\log(1/\rho)}$. One can provide a bound result for this special case.

Theorem 5. *Let* p *be a* strongly consistent *rational distribution with rank* d*. Let* S *be a sample i.i.d. from* p*, let* r_S *be the mapping deduced from the algorithm. There exists* C *such that for any* $0 < \delta < 1$*, for any tree* t *of length* k*, if* S *is an i.i.d. sample of size* N *then, with confidence at least* $1-\delta$*, one has:*

$$|r_s(t) - p(t)| < Ckd^{2k}\sqrt{\frac{\log(N)}{N}log^2(\frac{1}{\delta})}.$$

Proof. One bounds the length of a tree drawn by p: with a confidence greater than $1-\delta/2N$,

$$|t| < \frac{\log(2NC/\delta)}{\log(1/\rho)}.$$

Thus, with confidence $1-\delta/2$, S has only trees of length lower than $\frac{\log(2NC/\delta)}{\log(1/\rho)}$. By replacing n in the previous result, one obtains the conclusion. $\square$

5 Illustration

In order to illustrate the algorithm, we consider the distribution p defined by tree series of Example 2.

To study the behavior our algorithm, we consider an observation matrix 5×5, built on the set of trees $T = \{t_1, t_2, t_3, t_4, t_5\}$, where $t_1 = a$, $t_2 = f(a,a)$, $t_3 = f(a, f(a,a))$, $t_4 = f(f(a,a),a)$, $t_5 = f(f(a,a), f(a,a))$ and the set of contexts

$$C = \{\$, f(a,\$), f(\$,a), f(f(a,a),\$), f(\$, f(a,a))\}.$$

We generate i.i.d. samples from p of different sizes containing respectively 10^3, 10^4 , 10^5 and 10^6 trees. On Figure 2, we show the different eigenvalues (square of singular values) in decreasing order obtained from the different samples.

We can observe that the convergence of the computed series towards the target value, and the convergence of singular values, is closely $O(\frac{1}{\sqrt{|S|}})$ (in average values).

We also compared the average standard deviation of the probabilities of the trees in T obtained with our model, with rank 2 learned from the different learning samples, with the theoretical standard deviation of the classical probability (binomial) estimator. We have that $p(t_1) \simeq 0.6666$, $p(t_2) \simeq 0.0741$, $p(t_3) \simeq 0.0329$, $p(t_4) \simeq 0.0082$ and $p(t_5) \simeq 0.0037$. The values obtained are shown on Figure 3. The estimated standard deviation is, in majority (21/25), lower than the theoretical standard deviation of a the binomial estimator: The algorithm seems to work better than the simple frequency estimator for the task of density estimation.

Now, we consider the problem of the dimension estimate. The second singular value $\lambda_2 \sim 7.74 \cdot 10^{-3}$. Using the bound Δ_X to estimate the correct dimension (Theorem 2), we can estimate that:

- For $N = 10^6$, the rank 2 is found with a parameter $\delta \sim 0.59$ (confidence 0.41).
- For $N = 2 \cdot 10^6$, the rank 2 is found with a parameter $\delta \sim 0.12$ (confidence 0.88).
- For $N = 3 \cdot 10^6$, the rank 2 is found with a parameter $\delta \sim 0.02$ (confidence 0.98).

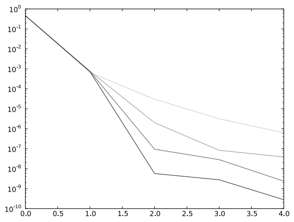


Fig. 2. Eigenvalues Curves - square of singular values - (in logarithmic scale) for sample size of 10^3, 10^4 , 10^5 and 10^6 trees (the lightest to the darkest)

t	σ_{10^2}	σ_{10^3}	σ_{10^4}	σ_{10^5}	σ_{10^6}
t_1	$3.76.10^{-2}$	$1.23.10^{-2}$	$3.50.10^{-3}$	$1.16.10^{-3}$	$4.12.10^{-4}$
	$4.71.10^{-2}$	*$1.49.10^{-2}$*	*$4.71.10^{-3}$*	*$1.49.10^{-3}$*	*$4.71.10^{-4}$*
t_2	$2.04.10^{-2}$	$6.77.10^{-3}$	$1.94.10^{-3}$	$7.30.10^{-4}$	$2.36.10^{-4}$
	$2.62.10^{-2}$	*$8.28.10^{-3}$*	*$2.62.10^{-3}$*	*$8.28.10^{-4}$*	*$2.62.10^{-4}$*
t_3	$1.63.10^{-2}$	$6.70.10^{-3}$	$2.13.10^{-3}$	$6.95.10^{-4}$	$1.95.10^{-4}$
	$1.78.10^{-2}$	*$5.64.10^{-3}$*	*$1.78.10^{-3}$*	*$5.64.10^{-4}$*	*$1.78.10^{-4}$*
t_4	$9.01.10^{-3}$	$2.23.10^{-3}$	$6.93.10^{-4}$	$2.20.10^{-4}$	$5.73.10^{-5}$
	$9.03.10^{-3}$	*$2.86.10^{-3}$*	*$9.03.10^{-4}$*	*$2.86.10^{-4}$*	*$9.03.10^{-5}$*
t_5	$4.90.10^{-3}$	$1.51.10^{-3}$	$4.52.10^{-4}$	$1.46.10^{-4}$	$3.90.10^{-5}$
	$6.04.10^{-3}$	*$1.91.10^{-3}$*	*$6.04.10^{-4}$*	*$1.91.10^{-4}$*	*$6.04.10^{-5}$*

Fig. 3. Average standard deviation of trees in T measured from the 2-dimensional model learned on samples of size 10^2, 10^3, 10^4, 10^5 and 10^6. The standard deviation of the theoretical binomial estimator is indicated in *italics.*

6 Conclusion and Discussion

We have studied the problem of learning an unknown distribution p from finite independently and identically drawn samples. We have proposed a new approach for identifying rational distributions on trees, or rational stochastic tree languages. Most classical inference algorithms in probabilistic grammatical inference build an automaton or a grammar iteratively from a sample S. Starting from an automaton composed of only one state, then they have to decide whether a new state must be added to the structure. This iterative decision relies on a statistical test with a known drawback: as the structure grows, the test relies on fewer and fewer examples. Instead of this iterative approach, we tackle the problem globally and our algorithm computes in one step the space needed to build the output automaton. That is, we have reduced the problem set in the classical probabilistic grammatical inference framework to a classical optimization problem. This point offers the interesting opportunity to apply classical results in statistical machine learning theory to probabilistic grammatical inference.

We have provided three types of results. First, we have given a result for convergence of eigenvalues which can be used for the estimation of the dimension of the target vector space, which is a crucial point in probabilistic grammatical inference and may allow to avoid costly cross-validation procedures. Second, we have provided error bounds for the convergence of the parameters of a linear representation. We have finally obtained pointwise convergence results for the probability estimate of a tree.

One perspective would then to obtain an ℓ_1-convergence, probably restricted to the case of strongly consistent stochastic tree languages, and to obtain tighter bounds. We finally need to experimentally study and compare our approach to existing ones on real data, this is a work in progress. Another perspective would consist in introducing non linearity via the kernel PCA technique developed in [17] and by the Hilbert space embedding of distributions proposed in [18, 19].

Acknowledgement

The authors wish to thank the anonymous reviewers for their comments and suggestions.

References

[1] Carrasco, R., Oncina, J., Calera-Rubio, J.: Stochastic inference of regular tree languages. Machine Learning 44, 185–197 (2001)

[2] Denis, F., Esposito, Y.: On rational stochastic languages. Fundamenta Informaticae 86, 41–77 (2008)

[3] Denis, F., Habrard, A.: Learning rational stochastic tree languages. In: Hutter, M., Servedio, R.A., Takimoto, E. (eds.) ALT 2007. LNCS (LNAI), vol. 4754, pp. 242–256. Springer, Heidelberg (2007)

[4] Denis, F., Gilbert, E., Habrard, A., Ouardi, F., Tommasi, M.: Relevant representations for the inference of rational stochastic tree languages. In: Grammatical Inference: Algorithms and Applications, 9th International Colloquium, pp. 57–70. Springer, Heidelberg (2008)

[5] Bailly, R., Denis, F., Ralaivola, L.: Grammatical inference as a principal component analysis problem. In: Proceedings of the 26th International Conference on Machine Learning, Montréal, Canada, pp. 33–40. Omnipress (2009)

[6] Clark, A., Costa Florêncio, C., Watkins, C.: Languages as hyperplanes: grammatical inference with string kernels. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 90–101. Springer, Heidelberg (2006)

[7] Hsu, D., Kakade, S., Zhang, T.: A spectral algorithm for learning hidden markov models. In: Proceedings of COLT 2009, Springer, Heidelberg (2009)

[8] Hsu, D., Kakade, S., Zhang, T.: A spectral algorithm for learning hidden markov models. Technical report, Arxiv archive (2009), http://arxiv.org/abs/0811.4413

[9] Denis, F., Esposito, Y.: Learning classes of probabilistic automata. In: Shawe-Taylor, J., Singer, Y. (eds.) COLT 2004. LNCS (LNAI), vol. 3120, pp. 124–139. Springer, Heidelberg (2004)

[10] Comon, H., Dauchet, M., Gilleron, R., Jacquemard, F., Lugiez, D., Löding, C., Tison, S., Tommasi, M.: Tree automata techniques and applications (2007), http://tata.gforge.inria.fr/ (release October 12, 2007)

[11] Berstel, J., Reutenauer, C.: Recognizable formal power series on trees. Theorical computer science 18, 115–148 (1982)

[12] Högberg, J., Maletti, A., Vogler, H.: Bisimulation minimisation of weighted automata on unranked trees. Fundam. Inform. 92(1-2), 103–130 (2009)
[13] Borchardt, B.: The Theory of Recognizable Tree Series. PhD thesis, TU Dresden (2004)
[14] McDiarmid, C.: On the method of bounded differences. In: Surveys in Combinatorics, pp. 148–188. Cambridge University Press, Cambridge (1989)
[15] Stewart, G., Sun, J.G.: Matrix Perturbation Theory. Academic Press, London (1990)
[16] Zwald, L., Blanchard, G.: On the convergence of eigenspaces in kernel principal component analysis. In: Proceedings of NIPS 2005 (2006)
[17] Shawe-Taylor, J., Cristianini, N., Kandola, J.: On the concentration of spectral properties. In: Proc. of NIPS, vol. 14, pp. 511–517. MIT Press, Cambridge (2001)
[18] Smola, A., Gretton, A., Song, L., Schölkopf, B.: A hilbert space embedding for distributions. In: 18th International Conference on Algorithmic Learning Theory, pp. 13–31. Springer, Heidelberg (2007)
[19] Song, L., Boots, B., Saddiqi, S., Gordon, G., Smola, A.: Hilbert space embeddings of Hidden Markov Models. In: Proceedings of ICML 2010 (2010)

PageRank Optimization in Polynomial Time by Stochastic Shortest Path Reformulation

Balázs Csanád Csáji[1,2], Raphaël M. Jungers[3,4], and Vincent D. Blondel[4]

[1] Department of Electrical and Electronic Engineering, School of Engineering, The University of Melbourne, Australia
bcsaji@unimelb.edu.au
[2] Computer and Automation Research Institute, Hungarian Academy of Sciences
[3] Lab. for Information and Decision Systems, Massachusetts Institute of Technology
[4] Department of Mathematical Engineering, Université catholique de Louvain, Belgium
vincent.blondel@uclouvain.be, raphael.jungers@uclouvain.be

Abstract. The importance of a node in a directed graph can be measured by its PageRank. The PageRank of a node is used in a number of application contexts – including ranking websites – and can be interpreted as the average portion of time spent at the node by an infinite random walk. We consider the problem of maximizing the PageRank of a node by selecting some of the edges from a set of edges that are under our control. By applying results from Markov decision theory, we show that an optimal solution to this problem can be found in polynomial time. It also indicates that the provided reformulation is well-suited for reinforcement learning algorithms. Finally, we show that, under the slight modification for which we are given mutually exclusive pairs of edges, the problem of PageRank optimization becomes NP-hard.

Keywords: PageRank, graphs, complexity, Markov decision processes.

1 Introduction

The *importance* of a node in a directed graph can be measured by its *PageRank*. The PageRank of a node [4] can be interpreted as the average portion of time spent at the node by an infinite *random walk* [10]. It is traditionally applied for ordering web-search results, but it also has many other applications [2], for example, in bibliometrics, ecosystems, spam detection, web-crawling, semantic networks, relational databases and natural language processing.

It is of natural interest to search for the maximum or minimum PageRank that a node (e.g., a website) can have depending on the presence or absence of some of the edges (e.g., hyperlinks) in the graph. For example, since PageRank is used for ordering web-search results, a web-master could be interested in increasing the PageRank of some of his websites by suitably placing hyperlinks on his own site or by buying advertisements or making alliances with other sites [1, 5]. Another motivation is that of estimating the PageRank of a node in the presence of *missing information* on the graph structure. If some of the links on the internet

© Springer-Verlag Berlin Heidelberg 2010

are broken, for example, because the server is down or there are network traffic problems, we may have only partial information on the link structure of the web-graph. However, we may still want to estimate the PageRank of a website by computing the maximum and minimum PageRank that the node may possibly have depending on the presence or absence of the hidden hyperlinks [9]. These hidden edges are often referred to as *fragile links*.

It is known that if we place a new edge in a directed graph, the PageRank of the terminal node of the edge can only increase. Optimal linkage strategies are known for the case in which we want to optimize the PageRank of a node and we only have access to the edges starting from this node [1]. This first result has later been generalized to the case for which we are allowed to configure all of the edges starting from a given set of nodes [5]. The general problem of optimizing the PageRank of a node in the case where we are allowed to decide the absence or presence of the edges in a given arbitrary subset of edges is proposed by Ishii and Tempo [9]. They are motivated by the problem of "fragile links" and mention the lack of efficient, polynomial time algorithms to this problem. Then, using interval matrices, they propose an approximate solution to the problem.

We show that the PageRank optimization problem can be efficiently formulated as a *Markov decision process* (MDP), more precisely, as a *stochastic shortest path* (SSP) problem, and that it can therefore be solved in *polynomial time*. Our proof provides a *linear programming* formulation that can then be solved by standard techniques. Our result and the given reformulation indicate that this problem is well-suited for *reinforcement learning* methods, as well. We also prove that under the slight modification for which we are given mutually exclusive constraints between pairs of edges, the problem becomes *NP-hard*.

2 Definitions and Preliminaries

In this section we define the concept of *PageRank* and the PageRank optimization problem as well as give an introduction to stochastic shortest path problems.

2.1 PageRank

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a directed graph, where $\mathcal{V} = \{1, \dots, n\}$ is the set of vertices and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. First, for simplicity, we assume that $\mathcal{G}$ is *strongly connected*. The adjacency matrix of $\mathcal{G}$ is denoted by A. Since $\mathcal{G}$ is strongly connected, A is *irreducible*. We are going to consider a random walk on the graph defined as follows. If we are in node i, in the next step we will go to node j with probability $1/deg(i)$ if j is an out-neighbor of i, where $deg(\cdot)$ denotes out-degree. This defines a *Markov chain* with transition-matrix

$$P \triangleq \left(D_A^{-1} A\right)^{\mathrm{T}} \qquad \text{with} \qquad D_A \triangleq diag(A\mathbb{1}) \tag{1}$$

where $\mathbb{1} = \langle 1, \dots, 1 \rangle^{\mathrm{T}}$ is the all-one vector and $diag(\cdot)$ is an operator that creates a diagonal matrix from a vector, more precisely, $(D_A)_{ii} \triangleq (A\mathbb{1})_i = deg(i)$. Note that P is a column (left) stochastic matrix and the chain can be interpreted as an infinite uniform random walk on the graph (e.g., a random surfing).

The PageRank vector, $\boldsymbol{\pi}$, of the graph is defined as the *stationary distribution* of the above described homogeneous Markov chain, more precisely, as $P\,\boldsymbol{\pi} = \boldsymbol{\pi}$, where $\boldsymbol{\pi}$ is non-negative and $\boldsymbol{\pi}^{\mathrm{T}}\mathbb{1} = 1$. Since P is an irreducible stochastic matrix, we know (Perron-Frobenius theorem) that $\boldsymbol{\pi}$ exists and it is unique.

Now, we turn to the general case, when we do not assume that $\mathcal{G}$ is strongly connected, it can be an arbitrary directed graph. In this case, there may be nodes which do not have any outgoing edges. They are usually referred to as *dangling nodes*. There are many ways to handle them [2], e.g., we can delete them, we can add a self-loop to them, each dangling node can be linked to an ideal node (sink) or we can connect each dangling node to every other node. This last solution can be interpreted as restarting the walk from a random starting state if we reach a dangling node. Henceforth, we will assume that we have already dealt with the dangling nodes and, hence, every node has at least one outgoing edge.

We can define a Markov chain similarly to (1), but this chain may not have a unique stationary distribution. The solve this problem, the PageRank vector, $\boldsymbol{\pi}$, of $\mathcal{G}$ is defined as the stationary distribution of the "Google matrix" [10]

$$G \triangleq (1-c)\,P + c\,\boldsymbol{z}\mathbb{1}^{\mathrm{T}}, \tag{2}$$

where $\boldsymbol{z}$ is a positive *personalization vector* satisfying $\boldsymbol{z}^{\mathrm{T}}\mathbb{1} = 1$, and $c \in (0,1)$ is a *damping constant*. In practice, values between 0.1 and 0.15 are usually applied for c and $\boldsymbol{z} = (1/n)\,\mathbb{1}$ [2]. The Markov chain defined by G is *ergodic* that is *irreducible* and *aperiodic*, hence, its stationary distribution uniquely exists and the Markov chain converges to it from any initial distribution [11].

The idea of PageRank is that $\boldsymbol{\pi}(i)$ can be interpreted as the "importance" of i. Thus, $\boldsymbol{\pi}$ defines a *linear order* on the nodes by treating $i \leq j$ if $\boldsymbol{\pi}(i) \leq \boldsymbol{\pi}(j)$.

The PageRank vector can be approximated by the iteration $x_{n+1} \triangleq G x_n$, where x_0 is an arbitrary stochastic vector. It can also be directly computed [1]

$$\boldsymbol{\pi} = c\,(I-(1-c)P)^{-1}\boldsymbol{z}, \tag{3}$$

where I denotes an $n \times n$ identity matrix. Since $c \in (0,1)$ and P is stochastic, it follows that matrix $I-(1-c)P$ is strictly diagonally dominant, thus invertible.

2.2 PageRank Optimization

We will investigate a problem in which a subset of links are "fragile", we do not know their exact values or we have control over them, and we want to compute the maximum (or minimum) PageRank that a specific node can have [9]. More precisely, we are given a digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a node $v \in \mathcal{V}$ and a set $\mathcal{F} \subseteq \mathcal{E}$ corresponding to those edges which are under our control. It means that we can choose which edges in $\mathcal{F}$ are present and which are absent, but the edges in $\mathcal{E} \setminus \mathcal{F}$ are fixed, they must be present in the graph. We will call any $\mathcal{F}_+ \subseteq \mathcal{F}$ a *configuration* of fragile links: $\mathcal{F}_+$ determines those edges that we add to the graph, while $\mathcal{F}_- = \mathcal{F} \setminus \mathcal{F}_+$ denotes those edges which we remove. The PageRank of node v under the $\mathcal{F}_+$ configuration is defined as the PageRank of v with

The Max-PageRank Problem	
Instance:	A digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a node $v \in \mathcal{V}$ and a set of controllable edges $\mathcal{F} \subseteq \mathcal{E}$.
Optional:	A damping constant $c \in (0, 1)$ and a stochastic personalization vector z.
Task:	Compute the maximum possible PageRank of v by changing the edges in $\mathcal{F}$ and provide a configuration of edges in $\mathcal{F}$ for which the maximum is taken.

respect to the graph $\mathcal{G}_0 = (\mathcal{V}, \mathcal{E} \setminus \mathcal{F}_-)$. The problem is how should we configure the fragile links to maximize (or minimize) the PageRank of a given node v?

There are finitely many configurations, thus, we can try to compute them one-by-one. If we have d fragile links, then there are 2^d possible graphs. The PageRank vector of a graph can be computed in $O(n^3)$ via a matrix inversion[1]. The resulting "exhaustive search" algorithm has $O(n^3 2^d)$ time complexity.

We note that if the graph was *undirected*, the Max-PageRank problem would be easy. We know [13] that a random walk on an undirected graph (viz., a time-reversible Markov chain) has the stationary distribution $\boldsymbol{\pi}(i) = deg(i)/2m$ for all nodes i, where m denotes the number of edges and $deg(i)$ is the degree of node i. Therefore, it is easy to see that, in order to maximize the PageRank of a given node v, we should keep edge $(i, j) \in \mathcal{F}$ if and only if $i = v$ or $j = v$.

2.3 Stochastic Shortest Path Problems

In this section we give an overview on stochastic shortest path problems, since our solutions to PageRank optimization are built upon their theory.

Stochastic shortest path (SSP) problems are generalizations of (deterministic) shortest path problems [3]. In an SSP problem the transitions between the nodes are uncertain, but we have some control over their probability distributions. We aim at finding a policy (a function from nodes to controls) such that minimizes the expected cost of reaching a given target state. SSP problems are undiscounted *Markov decision processes* (MDPs) with an absorbing, cost-free terminal state.

An SSP problem can be stated as follows. We have given a finite set of *states*, $\mathbb{S}$, and a finite set of control *actions*, $\mathbb{U}$. For simplicity, we assume that $\mathbb{S} = \{1, \dots, n, n+1\}$, where $\tau = n + 1$ is a special state, the *target* or *termination* state. In each state i we can choose an action $u \in \mathcal{U}(i)$, where $\mathcal{U}(i) \subseteq \mathbb{U}$ is the set of allowed actions in state i. After the action was chosen, the system moves to state j with probability $p(j \,|\, i, u)$ and we incur cost $g(i, u, j)$. The cost function is real valued and the transition-probabilities are, of course, nonnegative as well as they sum to one for each state i and action u. The target state is *absorbing* and *cost-free* that is, if we reach state τ, we remain there forever without incurring any more costs. More precisely, for all $u \in \mathcal{U}(\tau)$, $p(\tau \,|\, \tau, u) = 1$ and $g(\tau, u, \tau) = 0$.

The problem is to find a control *policy* such that it reaches state τ with probability one and minimizes the expected costs, as well. A (stationary, Markov) *deterministic* policy is a function from states to actions, $\mu : \mathbb{S} \to \mathbb{U}$. A *randomized* policy can be formulated as $\mu : \mathbb{S} \to \Delta(\mathbb{U})$, where $\Delta(\mathbb{U})$ denotes the set of all probability distributions over set $\mathbb{U}$. It can be shown that every such policy

[1] It can be done a little faster, in $O(n^{2.376})$, using the Coppersmith-Winograd method.

induces a *Markov chain* on the state space [6]. A policy is called *proper* if, using this policy, the termination state will be reached with probability one, and it is *improper* otherwise. The *value* or *cost-to-go* function of policy μ gives us the expected total costs of starting from a state and following μ thereafter; that is,

$$J^{\mu}(i) \triangleq \lim_{k\to\infty} \mathbb{E}_{\mu}\left[\sum_{t=0}^{k-1} g(i_t, u_t, i_{t+1}) \,\middle|\, i_0 = i\right], \tag{4}$$

for all states i, where i_t and u_t are random variables representing the state and the action taken at time t, respectively. Naturally, i_{t+1} is of distribution $p(\cdot \,|\, i_t, u_t)$ and u_t is of distribution $\mu(i_t)$; or $u_t = \mu(i_t)$ for deterministic policies.

Applying a proper policy, we arrive at a finite horizon problem, however, the length of the horizon may be random and may depend on the policy, as well.

We say that $\mu_1 \leq \mu_2$ if and only if for all states i, $J^{\mu_1}(i) \leq J^{\mu_2}(i)$. A policy is (uniformly) *optimal* if it is better than or equal to all other policies. There may be many optimal policies, but assuming that (A1) there exists at least one proper policy and (A2) every improper policy yields infinite cost for at least one initial state, they all share the same unique optimal value function, J^*. Then, function J^* is the unique solution of the *Bellman optimality equation*, $TJ^* = J^*$, where T is the *Bellman optimality operator* [3]; defined for all states i as

$$(TJ)(i) \triangleq \min_{u\in\mathcal{U}(i)} \sum_{j=1}^{n+1} p(j \,|\, i, u)\Big[g(i, u, j) + J(j)\Big]. \tag{5}$$

Operator T is *monotone* and, assuming that (APP) all allowed policies are proper, T is a *contraction* with respect to a weighted maximum norm [3].

From a given value function J, it is straightforward to get a policy, e.g., by applying a *greedy* policy with respect to J [3]. There are several solution methods for solving MDPs, e.g., in the fields of *reinforcement learning* and [neuro-] *dynamic programming*. Many of these algorithms aim at finding (or approximating) the optimal value function, since good approximations to J^* directly lead to good policies [3]. General solution methods include value iteration, policy iteration, Q-learning, SARSA and TD(λ): temporal difference learning [3, 6, 15].

It is known that finite MDPs are P-complete [14] and SSP problems can be reformulated as *linear programming* (LP) problems [3]. More precisely, the optimal cost-to-go, $J^*(1), \dots, J^*(n)$, solves the following LP in variables $x_1, \dots, x_n$:

$$\text{maximize} \qquad \sum_{i=1}^{n} x_i \tag{6a}$$

$$\text{subject to} \qquad x_i \leq \sum_{j=1}^{n+1} p(j \,|\, i, u)\Big[g(i, u, j) + x_j\Big] \tag{6b}$$

for all states i and actions $u \in \mathcal{U}(i)$. Note that x_{n+1} is fixed at zero. This LP has n variables and $O(nm)$ constraints, where m is the maximum number of allowed actions per state. Knowing that an LP can be solved in polynomial time [8], this reformulation provides a *polynomial time* solution to SSP problems.

3 PageRank Optimization as a Markov Decision Process

Before we prove that efficient algorithms to the Max-PageRank problem do exist, first, we recall a basic fact about stationary distributions of Markov chains.

Let $(X_0, X_1, \dots)$ denote a time-homogeneous Markov chain defined on a finite set Ω. The *expected first return time* of a state $i \in \Omega$ is defined as follows

$$\boldsymbol{\varphi}(i) \triangleq \mathbb{E}\left[\inf\{\, t \geq 1 : X_t = i \,\} \mid X_0 = i\,\right]. \tag{7}$$

If state i is *recurrent*, then $\varphi(i)$ is finite. Moreover, if the chain is irreducible,

$$\boldsymbol{\pi}(i) = \frac{1}{\boldsymbol{\varphi}(i)}, \tag{8}$$

for all states i, where $\boldsymbol{\pi}$ is the stationary distribution of the chain [11]. This naturally generalizes to *unichain* processes, viz., when we have a single *communicating class* of states and possibly some *transient* states. In this case we need the convention that $1/\infty = 0$, since the expected first return time to transient states is ∞. Hence, the stationary distribution of state i can be interpreted as the *average portion of time* spent in i during an infinite random walk. It follows from (8) that maximizing (minimizing) the PageRank of a node is equivalent to minimizing (maximizing) the expected first return time to this node.

We will show that the Max-PageRank problem can be efficiently formulated as a *stochastic shortest path* (SSP) problem [3], where "efficiently" means that the construction (reduction) takes polynomial time. First, we will consider the PageRank optimization *without damping*, namely $c = 0$, but later, we will extend the model to the case of damping and personalization, as well. We will start with a simple, but intuitive reformulation of the problem. Though, this reformulation will not ensure that Max-PageRank can be solved in polynomial time, it is good to demonstrate the main ideas and to motivate the refined solution.

3.1 Assumptions

First, we will make two assumptions, in order to simplify the presentation of the construction, but later, in the main theorem, they will be relaxed.

(AD) *Dangling Nodes Assumption*: We assume that there is a fixed (not fragile) outgoing edge from each node. This assumption guarantees that there are no dangling nodes and there are no nodes with only fragile links.

(AR) *Reachability Assumption*: We also assume that for at least one configuration of fragile links we have a unichain process and node v is recurrent, namely, we can reach node v with positive probability from all nodes. This assumption is required to have a well-defined PageRank for at least one configuration. In our SSP formulation this will be equivalent to assuming that there is at least one *proper* policy. In case of damping this assumption is automatically satisfied, since then the Markov chain is irreducible, and hence unichain, no matter how we configure the fragile links, thus all policies are proper.

3.2 Simple SSP Formulation

First, let us consider an instance of Max-PageRank. We are going to build an associated SSP problem that solves the original PageRank optimization. The *states* of the MDP are the nodes of the graph, except for v which we "split" into two parts and replace by two new states: v_s and v_t. Intuitively, state v_s will be our "starting" state: it has all the outgoing edges of v (both fixed and fragile), but it does not have any incoming edges. The "target" state will be v_t: it has all the incoming edges of node v and, additionally, it has only one outgoing edge: a self-loop. Note that $\tau = v_t$, namely, v_t is the *absorbing termination state.*

An *action* in state i is to select a subset of fragile links (starting from i) which we "turn on" (activate). All other fragile links from i will be "turned off" (deactivated). Thus, for all states i, the allowed set of actions is $\mathcal{U}(i) \triangleq \mathcal{P}(\mathcal{F}_i)$, where $\mathcal{P}$ denotes the power set and $\mathcal{F}_i$ the set of outgoing fragile edges from i.

Let us assume that we are in state i, where there are $a_i \geq 1$ fixed outgoing edges and we have activated $b_i(u) \geq 0$ fragile links, determined by action $u \in \mathcal{U}(i)$. Then, the *transition-probability* to all states j that can be reached from state i using a fixed or an activated fragile link is $p(j \,|\, i, u) \triangleq 1/(a_i + b_i(u))$.

We define the *immediate-cost* of all control actions as one, except for the actions taken at the cost-free target state. Thus, the immediate-cost function is

$$g(i,u,j) \triangleq \begin{cases} 0 & \text{if } i = v_t, \\ 1 & \text{otherwise}, \end{cases} \tag{9}$$

for all states i, j and actions u. Note that taking an action can be interpreted as performing a step in the random walk. Therefore, the expected cumulative cost of starting from state v_s until we reach the target state v_t is equal to the expected number of steps until we first return to node v according to our original random walk. It follows, that the above defined SSP formalizes the problem of *minimizing* (via a configuration) the expected first return time to state v. Consequently, its solution is equivalent to *maximizing* the PageRank of node v.

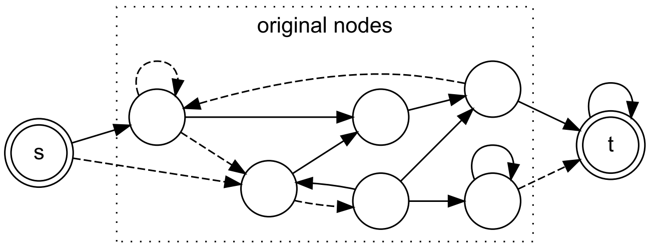


Fig. 1. SSP reformulation: the starting state is $s = v_s$, the target state is $t = v_t$ and the dashed edges denote fragile links. The original nodes in the rectangle exclude v.

Each allowed deterministic *policy* μ defines a potential way to configure the fragile links. Moreover, the v_s component of the cost-to-go function, $J^{\mu}(v_s)$, is the expected first return time to v using the fragile link configuration of μ. Therefore, we can compute the PageRank of node v by

$$\boldsymbol{\pi}(v) = \frac{1}{J^{\mu}(v_s)}, \tag{10}$$

where we applied the convention of $1/\infty = 0$, which is needed when v is not recurrent under μ. Thus, the maximal PageRank of v is $1/J^{*}(v_s)$.

It is known that MDPs can be solved in polynomial time in the number of states, N, and the maximum number of actions per state, M (and the maximum number of bits required to represent the components, L), e.g., by linear programming [12, 14]. The size of the state space of the current formulation is $N = n+1$, where n is the number of vertices of the original graph, but, unfortunately, its action space does not have a polynomial size. For example, if we have maximum m fragile links leaving a node, we had 2^m possible actions to take, namely, we could switch each fragile link independently on or off, consequently, $M = 2^m$. Since $m = O(n)$, from the current reformulation of problem, we have that there is a solution which is polynomial but in 2^n, which is obviously not good enough. However, we can notice that if we restrict the maximum number of fragile links per node to a constant, k, then we could have a solution which is polynomial in n (since the maximum number of actions per state becomes constant: 2^k). This motivates our refined solution, in which we reduce the maximum number of actions per state to two while only slightly increasing the number of states.

3.3 Refined SSP Formulation

We are going to modify our previous SSP formulation. The key idea will be to introduce an *auxiliary state* for each fragile link. Therefore, if we have a fragile link from node i to node j in the original graph, we place an artificial state, f_{ij}, "between" them in the refined reformulation. The refined transition-probabilities are as follows. Let us assume that in node i there were $a_i \geq 1$ fixed outgoing edges and $b_i \geq 0$ fragile links. Now, in the refined formulation, in state i we have only one available action which brings us uniformly, with $1/(a_i + b_i)$ probability, to state j or to state f_{ij} depending respectively on whether there was a fixed or a fragile link between i and j. Notice that this probability is *independent* of how many fragile links are turned on, it is always the same. In each auxiliary state f_{ij} we have two possible actions: we could either turn the fragile link on or off. If our action is "on" (activation), we go with *probability one* to state j, however, if our action is "off" (deactivation), we go back with *probability one* to state i.

We should check whether the transition-probabilities between the original nodes of graph are not affected by this reformulation. Suppose, we are in node i, where there are a fixed and b fragile links[2], and we have turned k of the fragile links on. Then, the transition-probability to each node j, which can be reached via a fixed or an activated fragile link, should be $1/(a + k)$. In our refined reformulation, the immediate transition-probability from state i to state

[2] For simplicity, now we do not denote their dependence on node i.

j is $1/(a+b)$, however, we should not forget about those $b-k$ auxiliary nodes in which the fragile links are deactivated and which lead back to state i with probability one, since, after we returned to state i we have again $1/(a+b)$ probability to go to state j an so on. Now, we will compute the probability of eventually arriving at j if we start in i and only visit auxiliary states meantime.

To simplify the calculations, let us temporarily replace each edge leading to an auxiliary sate corresponding to a *deactivated* fragile link with a self-loop. We can safely do so, since these states lead back to state i with probability one, therefore, the probability of eventually arriving at node j does not change by this modification. Then, the probability of arriving at j can be written as

$$\mathbb{P}\,(\,\exists\, t : X_t = j \,|\, \forall\, s < t : X_s = i\,) \;= \tag{11a}$$

$$= \sum_{t=1}^{\infty} \mathbb{P}\,(\,X_t = j \,|\, X_{t-1} = i\,) \prod_{s=1}^{t-1} \mathbb{P}\,(\,X_s = i \,|\, X_{s-1} = i\,) \;= \tag{11b}$$

$$= \sum_{t=1}^{\infty} \frac{1}{a+b} \left(\frac{b-k}{a+b}\right)^{t-1} = \frac{1}{a+b} \sum_{t=0}^{\infty} \left(\frac{b-k}{a+b}\right)^{t} = \frac{1}{a+k}\,. \tag{11c}$$

With this, we have proved that the probability of eventually arriving at state j if we start in state i, before arriving at any (non-auxiliary) state l that was reachable via a fixed or a fragile link from state i in the original graph, is the same as the one-step transition-probability was from state i to state j according to the original random walk. This partially justifies the construction.

However, we should be careful, since we might have performed several steps in the auxiliary nodes before we finally arrived at state j. Fortunately, this phenomenon does not ruin our ability to optimize the expected first return time to state v in the original graph, since we count the steps with the help of the cost function, which can be refined according to our needs. All we have to do is to allocate zero cost to those actions which lead us to auxiliary states:

$$g(i,u,j) \triangleq \begin{cases} 0 & \text{if } \; i = v_t \; \text{ or } \; j = f_{il} \; \textit{ or } \; u = \text{“off”}, \\ 1 & \text{otherwise}, \end{cases} \tag{12}$$

for all states i, j, l and action u. Consequently, we only incur cost if we directly go from state i to state j, without visiting an auxiliary node (viz., it was a fixed link), or if we go to state j via an activated fragile link, since we have $g(f_{ij}, u, j) = 1$ if $u =$ “on”. It is easy to see that in this way we only count the steps of the original random walk and, for example, it does not matter how many times we visit auxiliary nodes, since these visits do not have any cost.

This reformulation also has the nice property that $J^{\mu}(v_s)$ is the expected first return time to node v in the original random walk, in case we have configured the fragile links according to policy μ. The minimum expected first return time that can be achieved with suitably setting the fragile links is $J^*(v_s)$, where J^* is the optimal cost-to-go function of the above constructed SSP problem. Consequently, the *maximum* PageRank node v can have is $1/J^*(v_s)$.

It is also easy to see that if we want to compute the *minimum* possible PageRank of node v, we should simply define a new immediate-cost function as $\hat{g} = -\,g$, where g is defined by equation (12). If the optimal cost-to-go function of this modified SSP problem is $\hat{J}^*$, then the *minimum* PageRank v can have is $1/|\hat{J}^*(v_s)|$. Thus, Min-PageRank can be handled with the same construction.

The number of states of this formulation is $N = n + d + 1$, where n is the number of nodes of the original graph and d is the number of fragile links. Moreover, the maximum number of allowed actions per state is $M = 2$, therefore, this SSP formulation provides a proof that, assuming (AD) and (AR), Max-PageRank can be solved in *polynomial time* in the size of the problem.

The resulted SSP problem can be reformulated as a linear program, namely, the optimal cost-to-go function solves the following LP in variables x_i and x_{ij},

$$\text{maximize} \qquad \sum_{i \in \mathcal{V}} x_i + \sum_{(i,j) \in \mathcal{F}} x_{ij} \tag{13a}$$

$$\text{subject to} \qquad x_{ij} \leq x_i\,, \quad \text{and} \quad x_{ij} \leq x_j + 1\,, \quad \text{and} \tag{13b}$$

$$x_i \leq \frac{1}{deg(i)} \left[\sum_{(i,j) \in \mathcal{E} \setminus \mathcal{F}} (x_j + 1) + \sum_{(i,j) \in \mathcal{F}} x_{ij} \right], \tag{13c}$$

for all $i \in \mathcal{V} \setminus \{v_t\}$ and $(i,j) \in \mathcal{F}$, where x_i is the cost-to-go of state i, x_{ij} relates to the auxiliary states of the fragile edges, and $deg(\cdot)$ denotes out-degree including both fixed and fragile links (independently of the configuration). Note that we can only apply this LP after state v was "splitted" into a starting and a target state. Also note that the value of the target state, x_{v_t}, is fixed at zero.

3.4 Handling Dangling Nodes

Now, we are going to show that assumption (AD) can be omitted and our complexity result is independent of how dangling nodes are particularly handled.

Suppose that we have chosen a rule according to which the dangling nodes are handled, e.g., we take one of the rules discussed by Berkhin [2]. Then, in case (AD) is not satisfied, we can simply apply this rule to the dangling nodes before the optimization. However, we may still have problems with the nodes which only have fragile links, since they are latent dangling nodes, namely, they become dangling nodes if we deactivate all of their outgoing edges. We call them "fragile nodes". Notice that in each fragile node we can safely restrict the optimization in a way that maximum one of the fragile links can be activated. This does not affect the optimal PageRank of v, since the only one allowed link should point to a node that has the smallest expected hitting time to v. Even if there are several nodes with the same value, we can select one of them arbitrarily.

It may also be the case that deactivating all of the edges is the optimal solution, e.g., if the fragile links lead to nodes that have very large hitting times to v. In this case, we should have an action that has the same effect as the dangling node handling rule. Consequently, in case we have a fragile node that has m fragile links, we will have $m+1$ available actions: $u_1, \dots, u_{m+1}$. If u_j is selected, where $1 \leq j \leq m$, it means that only the j-th fragile link is activated and all other links are deactivated, while if u_{m+1} is selected, it means that all of the fragile links are deactivated and auxiliary links are introduced according to the selected dangling node handling rule. If we treat the fragile nodes this way, we still arrive at an MDP which has states and actions polynomial in n and d, therefore, Max-PageRank can be solved in polynomial time even if (AD) is not satisfied and independently of the applied rule. The modification of the LP formulation if fragile nodes are allowed is straightforward.

3.5 Damping and Personalization

Now, we are going to extend our refined SSP formulation, in order to handle *damping*, as well. For the sake of simplicity, we will assume (AD), but it is easy to remove it in a similar way as it was presented in Section 3.4. Note that assumption (AR) is always satisfied in case of damping (cf. Section 3.1).

Damping can be interpreted as follows: in each step we continue the random walk with probability $1-c$ and we restart it ("zapping") with probability c, where $c \in (0,1)$ is a given damping constant. In this latter case, we choose the new starting state of the random walk according to the probability distribution of a given positive and stochastic personalization vector $\boldsymbol{z}$. In order to model this, we introduce a new global auxiliary state, q, which we will call the *teleportation state*, since random restarting is sometimes referred to as "teleportation" [10].

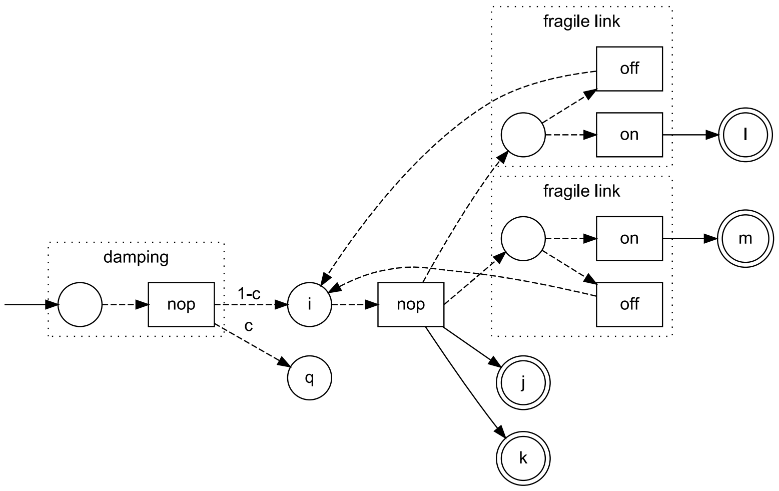


Fig. 2. An illustration of damping: the substructure of a node of the original digraph. Circles represent states and boxes represent actions. State q denotes the global "teleportation" state. Dashed edges help determining zero cost events: if a state-action-state path has only dashed edges, then this triple has zero cost, otherwise, its cost is one.

In order to take the effect of damping into account, we place a new auxiliary state h_i "before" each (non-auxiliary) state i (see Figure 2). Each action that leads to i in the previous formulation now leads to h_i. In h_i we have only one available action ("nop" abbreviating "no operation") which brings us to node i with probability $1-c$ and to q with probability c, except for the target state, v_t, for which h_{v_t} leads with probability one to v_t. In q, we have one available action which brings us with $\boldsymbol{z}$ distribution to the newly defined nodes,

$$p(\,h_i\,|\,q\,) \triangleq p(\,h_i\,|\,q,u\,) \triangleq \begin{cases} \boldsymbol{z}(i) & \text{if } \ i \neq v_s \ \text{ and } \ i \neq v_t \\ \boldsymbol{z}(v) & \text{if } \ i = v_t \\ 0 & \text{if } \ i = v_s. \end{cases} \tag{14}$$

All other transition-probabilities from q are zero. Regarding the cost function: it is easy to see that we should not count the steps when we move through h_i, therefore, $g(h_i, u, i) = 0$ and $g(h_i, u, q) = 0$. However, we should count when we move out from the teleportation state, i.e., $g(q, u, i) = 1$ for all i and u.

In this variant the size of the state space is $N = 2n + d + 2$ and we still have maximum 2 actions per state, therefore, it can also be solved in *polynomial time.*

In this case, the LP formulation of finding the optimal cost-to-go is

$$\text{maximize} \qquad \sum_{i\in\mathcal{V}} (x_i + \hat{x}_i) \,+\, \sum_{(i,j)\in\mathcal{F}} x_{ij} \,+\, x_q \tag{15a}$$

$$\text{subject to} \qquad x_{ij} \leq \hat{x}_j + 1\,, \qquad \text{and} \qquad \hat{x}_i \leq (1-c)\,x_i + c\,x_q\,, \tag{15b}$$

$$x_{ij} \leq x_i\,, \qquad \text{and} \qquad x_q \,\leq\, \sum_{i\in\mathcal{V}} \hat{z}_i\,(\hat{x}_i + 1)\,, \tag{15c}$$

$$x_i \,\leq\, \frac{1}{deg(i)}\left[\,\sum_{(i,j)\in\mathcal{E}\setminus\mathcal{F}} (\hat{x}_j + 1) \,+\, \sum_{(i,j)\in\mathcal{F}} x_{ij}\,\right], \tag{15d}$$

for all $i \in \mathcal{V} \setminus \{v_t\}$ and $(i,j) \in \mathcal{F}$, where $\hat{z}_i = p(\,h_i\,|\,q)$, $\hat{x}_i$ denotes the cost-to-go of state h_i and x_q is the value of the teleportation state, q. All other notations are the same as in LP (13) and we also have that x_{v_t} and $\hat{x}_{v_t}$ are fixed at zero.

We arrived at an LP problem with $O(n+d)$ variables and $O(n+d)$ constraints. Given an LP with k variables and $O(k)$ constraints, it can be solved in $O(k^3L)$, where L is the binary input size (for rational coefficients) or in $O(k^3 \log \frac{1}{\varepsilon})$, where ε is the desired precision [8]. Therefore, Max-PageRank can be solved using $O((n+d)^3L)$ operations under the *Turing model of computation.* Thus:

Theorem 1. *The* MAX-PAGERANK PROBLEM *can be solved in polynomial time even if the damping constant and the personalization vector are part of the input.*

Note that assumptions (AD) and (AR) are not needed for this theorem, since dangling and fragile nodes can be treated as discussed in Section 3.4 (without increasing the complexity) and, in case of damping, all policies are proper.

4 PageRank Optimization with Constraints

In this section we are going to investigate a variant of the PageRank optimization problem in which there are mutually exclusive constraints between the fragile links. More precisely, we will consider the case in which we are given a set of fragile link pairs, $\mathcal{C} \subseteq \mathcal{F} \times \mathcal{F}$, that cannot be activated simultaneously.

The Max-PageRank Problem under Exclusive Constraints

Instance: A digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a node $v \in \mathcal{V}$, a set of controllable edges $\mathcal{F} \subseteq \mathcal{E}$ and a set $\mathcal{C} \subseteq \mathcal{F} \times \mathcal{F}$ of those edge-pairs that cannot be activated together. A damping constant $c \in (0, 1)$ and a stochastic personalization vector z.

Task: Compute the maximum possible PageRank of v by activating edges in $\mathcal{F}$ and provide a configuration of edges in $\mathcal{F}$ for which the maximum is taken.

We will show that the Max-PageRank problem under exclusive constraints is already *NP-hard*, more precisely, that the decision version of it is NP-complete. In the decision version, one is given a number $p \in (0, 1)$ and is asked whether there is a configuration such that the PageRank is larger or equal to p.

Theorem 2. *The decision version of the* Max-PageRank Problem under Exclusive Constraints *is NP-complete.*

Proof. The problem is in NP because given a solution (viz., a configuration), it is easy to *verify* in polynomial time, e.g., via a simple matrix inversion, cf. equation (3), whether the corresponding PageRank is larger or equal than p.

We now reduce the 3SAT problem, whose NP-completeness is well known [7], to this problem. In an instance of the 3SAT problem, we are given a Boolean formula containing m disjunctive *clauses* of three *literals* that can be a variable or its negation, and one is asked whether there is a truth assignment to the variables so that the formula (or equivalently: each clause) is satisfied. Suppose now we are given an instance of 3SAT. We will construct an instance of Max-PageRank under exclusive constraints that solves this particular instance of 3SAT.

We construct a graph having $m + 2$ nodes in the following way: we first put a node s and a node t. Figure it as a source node and a sink node respectively. Each clause in the given 3SAT instance can be written as $y_{j,1} \vee y_{j,2} \vee y_{j,3}$, $1 \leq j \leq m$, where $y_{j,l}$ is a variable or its negation. For each such clause, we add a node v_j between s and t, we put an edge from v_j to itself (a self-loop), we put an edge from s to v_j, and we put three edges between v_j and t, labeled respectively with $y_{j,1}, y_{j,2}$, and $y_{j,3}$. We finally add an edge from t to s. We now define the set of exclusive constrains, $\mathcal{C}$, which concludes the reduction. For all pairs $(y_{j,l}, y_{j',l'})$ such that $y_{j,l} = \bar{y}_{j',l'}$ (i.e., $y_{j,l}$ is a variable and $\bar{y}_{j',l'}$ is its negation, or conversely), we forbid the corresponding pair of edges. Also, for all pairs of edges $(y_{j,l}, y_{j,l'})$ corresponding to a same clause node, we forbid the corresponding pair. This reduction is suitable, since the sizes of the graph and $\mathcal{C}$ are *polynomial* in the size of the 3SAT instance.

We claim that for c small enough, say $c = 1/(100m)$, it is possible to obtain an expected return time from t to itself which is smaller than 77 if and only if the instance of 3SAT is satisfiable. The reason for that is easy to understand with $c = 0$: if the instance is not satisfiable, there is a node v_j with no edge from it to t. In that case, the graph is not strongly connected, and the expected return time from t to itself is infinite. Now, if the instance is satisfiable, let us consider a particular satisfiable assignment. We activate all edges which correspond to a literal which is true and, if necessary, we deactivate some edges so that for all clause nodes, there is exactly one leaving edge to t. This graph, which is clearly satisfiable, is strongly connected, and so the expected return time to t is finite.

Now if $c \neq 0$ is small enough, one can still show by continuity that the expected return time is much larger if some clause node does not have an outgoing edge to t. To see this, let us first suppose that the instance is not satisfiable, and thus that a clause node (say, v_1), has no leaving edge. Then, for all $l \geq 3$, we describe a path of length l from t to itself: this path passes through s, and then remains during $l-2$ steps in v_1, and then jumps to t (with a zapping). This path has probability $(1-c)\frac{1}{m}(1-c)^{l-2}c$. Thus, the expected return time

$$E_1 \geq \sum_{l=3}^{\infty} l p(l) \geq \frac{c}{m} \sum_{l=3}^{\infty} l(1-c)^{l-1} \geq \frac{c}{m}\left[c^{-2}-3\right] \geq 99, \tag{16}$$

where we assumed that $c = 1/(100m)$ and the personalization vector is $z = (1/n)\,\mathbb{1}$. Note that c and z are part of the input, thus they can be determined.

Consider now a satisfiable instance, and build a corresponding graph so that for all clause nodes, there is exactly one leaving edge. It appears that the expected return time from t to itself satisfies $E_2 \leq 77$. To see this, one can aggregate all the clause nodes in one macro-node, and then define a Markov chain on three nodes that allows to derive a bound on the expected return time from v_t to itself. This bound does not depend on m because one can approximate the probabilities $m/(m+2)$ and $1/(m+2)$ that occur in the auxiliary Markov chain by one so that the bound remains true. Then, by bounding c with $1/8 > 1/(100m)$, one gets an upper bound on the expected return time. For the sake of conciseness, we skip the details of the calculations. To conclude the proof, it is possible to find an edge assignment in the graph so that the PageRank is greater than $p = 1/77$ if and only if the instance is satisfiable. $\square$

5 Conclusions

The task of ordering the nodes of a directed graph according to their *importance* arises in many applications. A promising and popular way to define such an ordering is to use the *PageRank* method [4]. The problem of optimizing the PageRank of a given node by changing some of the edges caused a lot of recent interest [1, 5, 9]. We considered the general problem of finding the extremal values of the PageRank a node can have in the case we are allowed to control (activate or deactivate) some of the edges, which we referred to as *fragile links.*

Our main contribution was that we proved that these problems could be efficiently formulated as stochastic shortest path problems (special Markov decision processes). This does not only imply that they can be solved in *polynomial time*, but also show that they are well-suited for *reinforcement learning* methods.

We note that we do not need to assume that the graph is simple, namely, it can have multiple edges (and self-loops). This allows the generalization of our results to *weighted graphs*, in case the weights are positive integers or rationals.

We also showed that slight modifications of the problem, as for instance adding mutual exclusive constraints between the activation of several fragile links, may turn the problem *NP-hard*. We conjecture that several other modified variants of the problem are also NP-hard, e.g., the Max-PageRank problem with restrictions on the number of fragile links that can be simultaneously activated.

Acknowledgments

The authors are grateful for the valuable discussions to Paul Van Dooren, Yurii Nesterov, Cristobald de Kerchove, Vincent Traag and Tzvetan Ivanov.

References

[1] Avrachenkov, K., Litvak, N.: The effect of new links on Google PageRank. Stochastic Models 22, 319–331 (2006)
[2] Berkhin, P.: A survey on PageRank computing. Internet Math. pp. 73–120 (2005)
[3] Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-Dynamic Programming. Athena Scientific, Belmont (1996)
[4] Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: Proc. of the 7th Intern. Conf. on World Wide Web, pp. 107–117 (1998)
[5] De Kerchove, C., Ninove, L., Van Dooren, P.: Maximizing PageRank via outlinks. Linear Algebra and its Applications 429, 1254–1276 (2008)
[6] Feinberg, E.A., Shwartz, A. (eds.): Handbook of Markov Decision Processes: Methods and Applications. Kluwer Academic Publishers, Dordrecht (2002)
[7] Garey, M.R., Johnson, D.S.: Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York (1990)
[8] Gonzaga, C.C.: An Algorithm for Solving Linear Programming Problems in $O(n^3L)$ operations. In: Progress in Mathematical Programming: Interior-Point and Related Methods, pp. 1–28. Springer, Heidelberg (1988)
[9] Ishii, H., Tempo, R.: Computing the PageRank variation for fragile web data. SICE J. of Control, Measurement, and System Integration 2(1), 1–9 (2009)
[10] Langville, A.N., Meyer, C.D.: Google's PageRank and Beyond: The Science of Search Engine Rankings. Princeton University Press, Princeton (2006)
[11] Levin, D.A., Peres, Y., Wilmer, E.L.: Markov Chains and Mixing Times. American Mathematical Society, Providence (2009)
[12] Littman, M.L., Dean, T.L., Kaelbling, L.P.: On the complexity of solving Markov decision problems. In: Proc. of the Eleventh International Conference on Uncertainty in Artificial Intelligence, pp. 394–402 (1995)
[13] Lovász, L.: Random walks on graphs: A survey. In: Combinatorics: Paul Erdős is Eighty, vol. 2, pp. 348–353. Bolyai Society Mathematical Studies, Budapest (1996)
[14] Papadimitriou, C.H., Tsitsiklis, J.N.: The complexity of Markov decision processes. Mathematics of Operations Research 12(3), 441–450 (1987)
[15] Sutton, R.S., Barto, A.G.: Reinforcement learning. MIT Press, Cambridge (1998)

Inferring Social Networks from Outbreaks

Dana Angluin[1,⋆], James Aspnes[1,⋆⋆], and Lev Reyzin[2,⋆⋆⋆]

[1] Department of Computer Science, Yale University
51 Prospect St., New Haven, CT 06511
{dana.angluin,james.aspnes}@yale.edu
[2] Yahoo! Research
111 West 40th St. 17th Fl., New York, NY 10018
lreyzin@yahoo-inc.com

Abstract. We consider the problem of inferring the most likely social network given connectivity constraints imposed by observations of outbreaks within the network. Given a set of vertices (or agents) V and constraints (or observations) $S_i \subseteq V$ we seek to find a minimum log-likelihood cost (or maximum likelihood) set of edges (or connections) E such that each S_i induces a connected subgraph of (V, E). For the offline version of the problem, we prove an $\Omega(\log(n))$ hardness of approximation result for uniform cost networks and give an algorithm that almost matches this bound, even for arbitrary costs. Then we consider the online problem, where the constraints are satisfied as they arrive. We give an $O(n \log(n))$-competitive algorithm for the arbitrary cost online problem, which has an $\Omega(n)$-competitive lower bound. We look at the uniform cost case as well and give an $O(n^{2/3} \log^{2/3}(n))$-competitive algorithm against an oblivious adversary, as well as an $\Omega(\sqrt{n})$-competitive lower bound against an adaptive adversary. We examine cases when the underlying network graph is known to be a star or a path, and prove matching upper and lower bounds of $\Theta(\log(n))$ on the competitive ratio for them.

1 Introduction

In the real world, we often observe patterns that expose information about an underlying structure that we are interested in discovering, and in this paper, we focus our attention on learning social networks by passively observing such patterns. Here, we consider inferring the structure of social networks by observing phenomena that give us information about their connectivity. Our observations may be limited, and we may not be able to infer the underlying networks precisely – in that case, we can try to find the most likely structure given prior beliefs.

⋆ Supported in part by the National Science Foundation under grant CCF-0916389.
⋆⋆ Supported in part by the National Science Foundation under grants CNS-0435201 and CCF-0916389.
⋆⋆⋆ Work supported in part by the National Science Foundation under a National Science Foundation Graduate Research Fellowship and in part under Grant # 0937060 to the Computing Research Association for the Computing Innovation Fellowship program.

© Springer-Verlag Berlin Heidelberg 2010

For example, in the United States, the Centers for Disease Control and Prevention release various data[1] on persons affected by illnesses – where, ideally, we would have information on exactly who is affected in each outbreak. From this information we can try to learn the network's underlying structure. In an idealized setting, we can consider persons infected during one outbreak as connected subsets in a population – for a disease spreads among persons in close proximity, and such data impose **constraints** on the topology of the network. Given such constraints, the problem would then be to find a maximum likelihood social network from the disease data.

Thus, each set imposes a constraint on the network – namely that it be connected in its induced subgraph. The goal of the learner is to infer the most probable network that satisfies the connectivity requirements presented to it.

The learner can also have a prior belief about the probability each edge appears in the network. Let $p_{(u,v)}$ be the a priori probability of an edge appearing between nodes u and v. If the prior distribution on edges is independent and each edge appears with low probability, the goal of finding a maximum likelihood social network given the constraints is to find a set of edges E that satisfies all of the constraints, for which the quantity

$$\prod_{\{u,v\}\in E} p_{(u,v)} \prod_{\{u,v\}\notin E} \left(1-p_{(u,v)}\right) = \prod_{\{u,v\}} \left(1-p_{(u,v)}\right) \prod_{\{u,v\}\in E} \frac{p_{(u,v)}}{\left(1-p_{(u,v)}\right)}$$

is maximized. Taking the logarithm, we want a set of edges E that minimizes the sum

$$\sum_{\{v,u\}\in E} -\log\left(\frac{p_{(u,v)}}{\left(1-p_{(u,v)}\right)}\right).$$

We assume that all quantities $p_{(u,v)} \leq 1/2$, meaning that we *a priori* do not expect any pair of given agents to be connected. This assumption implies that each term in the sum—the log-likelihood **cost** of the edge—is non-negative.

We can now think of the priors in terms of these costs. The goal of the learner becomes to construct the cheapest network (with respect to the prior costs) that satisfies the connectivity constraints.

This task presents various natural variations. We can consider what happens if the constraints are given to the learner in advance, and when the constraints arrive online. If they arrive online, they can be chosen adversarially or obliviously. We can imagine all edges in a network having the same cost, or that edges in a network have arbitrary costs. There are also cases when some information about the underlying social network is known, for example, that there exists a path that satisfies all the constraints.

1.1 Past Work

In the area of active learning, Angluin et al. [7] study the problem of reconstructing independent cascade social networks by activating and suppressing

[1] For more on CDC statistics, we direct the reader to www.cdc.gov/datastatistics/

various nodes in the network. The problem of actively discovering various networks from connectivity queries has been much studied in [2, 6, 8, 9, 14, 18]. In active learning of hidden networks, the object of the algorithm is to learn the network exactly. Our model is similar, except the learner is passive and has only the constraints it is given. Our task is to output the most likely network consistent with the constraints without the ability to query the network.

In a different passive learning model, Akutsu et al. [1] consider the problem of completing networks consistently with observations. Their model assumes the learner has more information and uses different target networks, but it is an example of a model that captures some of the spirit of our problem.

A variant of our problem was also considered by Korach and Stern [16] in another context where users in a **trusted set** in a network want to send messages among themselves without having the messages travel outside the group. Trusted sets of users can overlap, creating complicated structures, and these trusted sets form connectivity constraints in their subgraphs, imposing similar requirements to those in the social network inference problem.

In [16] Korach and Stern analyze the offline version of this problem for the case where the constraints can be satisfied by a tree. They give a polynomial time algorithm that finds the optimal solution in the tree-realizable case. In [17] Korach and Stern consider this problem for the case where the optimal solution forms a tree, and all of the connectivity constraints must be satisfied by stars. They pose as an open question the case of general graphs. Among our other results, we answer their question in this paper.

In a different line of work, Alon *et al.* [5] explore a wide range of network optimization problems, including generalized connectivity, cuts, facility location, and multicast. The connectivity problem they study involves ensuring a network with fractional edge weights has a flow of 1 over cuts specified by the constraints. In [3], Alon *et al.* also study approximation algorithms for the Online Set Cover problem, which has connections to Network Inference problems which we explore in this paper. In [15] Gupta *et al.* also consider a network design problem for pairwise vertex connectivity constraints.

1.2 Preliminaries

In this paper, we consider the following **Network Inference** problem. V is a set of vertices, and for each undirected edge $e = (v_i, v_j)$, c_e is the cost of constructing edge e. A collection of connectivity constraints $S = \{S_1, S_2, \ldots, S_r\}$ is given, where each S_i is a subset of V. The task is to construct a set E of edges between vertices of V such that for each i, the set S_i induces a connected subgraph of $G = (V, E)$. The quality of the solution is measured by comparing the sum of the costs of all the edges in E with the optimal cost of satisfying all the constraints.

In the offline version of the problem, the algorithm knows all of the constraints at the outset; in the **Online Network Inference** problem, the constraints are given to the algorithm one by one, and edges must be added to G to satisfy each new constraint. By default, we allow the edges to have **arbitrary costs**, but in the **uniform cost** version of the problem the edge costs are all equal to 1.

When we restrict the **underlying graph** in a problem to a smaller class of graphs, we mean that all constraints S_i can be satisfied (for the online case, in hindsight), by a graph from that class.

1.3 Our Results

In Section 2 we analyze the offline problem, where we show that the Uniform Cost Network Inference problem (and therefore the arbitrary cost one) has a hardness of approximation lower bound of $\Omega(\log(n))$ times the optimal solution. We give an $O(\log(r))$ approximation algorithm, where r is the number of constraints. This matches the lower bound when r is polynomial in n.

In Section 3, we look at the Online Network Inference Problem. First, in Subsection 3.1, we look at the case when the underlying uniform cost graph is a star or path. In both cases, we show that the optimal algorithm has an $\Omega(\log(n))$-competitive ratio and give a matching $O(\log(n))$-competitive algorithm. We also show that in the case when edges have non-uniform costs, there is no cn-competitive algorithm for any $c < 1$, even when the underlying graph is a path.

Then, in Subsection 3.2 we consider the general case of Online Network Inference, where the topology of the underlying graph is unrestricted. There we give an $O(n\log(n))$-competitive algorithm for the arbitrary cost case, that almost matches our lower bound. For the Uniform Cost Network Inference problem, we give an $\Omega(\sqrt{n})$-competitive lower bound, and in the case of an oblivious adversary, we give an $O(n^{2/3}\log^{2/3}(n))$-competitive algorithm.

2 Offline Network Inference

We first examine the offline Uniform Cost Network Inference problem.

Theorem 1. *If P≠NP, the approximation ratio for the* ***Uniform Cost Network Inference*** *problem on n nodes is $\Omega(\log n)$.*

Proof. We reduce from the Hitting Set problem. The inputs to Hitting Set are $U = \{v_1, v_2, \ldots, v_n\}$ and $\{C_1, C_2, \ldots, C_j\}$ with $C_i \subseteq U$. The **Hitting Set** problem is to minimize $|H|$, where $H \subseteq U$ such that $\forall C_i$, $H \cap C_i \neq \emptyset$. We define an instance of the Uniform Cost Network Inference problem with n^3 by n vertices $v_{(i,j)}$, for all $1 \leq i \leq n^3$ (rows) and $1 \leq j \leq n$ (columns). For each i, the vertices in row i, $\{v_{(i,1)}, v_{(i,2)}, \ldots, v_{(i,n)}\}$, correspond to the elements $\{v_1, \ldots, v_n\}$ in the Hitting Set instance.

Now we define the connectivity constraints for the Uniform Cost Network Inference problem. First we enforce that all pairs of vertices in each row i are connected, by adding a connectivity constraint for each pair of vertices $\{v_{(i,j)}, v_{(i,k)}\}$. For each constraint C_i in the Hitting Set problem, we create $\binom{n^3}{2}$ connectivity constraints. Without loss of generality, let $C_i = \{v_1, v_2, \ldots, v_k\}$. For each pair $l \neq j$ such that $1 \leq l, j \leq n^3$ we add a connectivity constraint

$$S_{C_i}^{l,j} = \{v_{(l,1)}, v_{(l,2)}, \ldots, v_{(l,k)}, v_{(j,1)}, v_{(j,2)}, \ldots, v_{(j,k)}\} \tag{1}$$

in the Uniform Cost Network Inference problem. This enforces the Hitting Set constraints pairwise between the n^3 rows of the network inference problem.

Each pair of rows in our new instance contains the original Hitting Set instance. First, the algorithm has no choice but to place a clique on each row. Then, let equation (1) be a constraint. To satisfy $S_{C_i}^{l,j}$, the algorithm must choose some edge between row l and row j among vertices $1, \ldots, k$. We observe that if the algorithm chooses an edge between two vertices corresponding to different elements in the two rows, it could do at least as well by choosing the edge going between two copies of one of the two elements. To see this, if edge $(v_{(l,x)}, v_{(j,y)})$, with $x \neq y$, is chosen to satisfy the constraint $S_{C_i}^{l,j}$, edge $(v_{(l,x)}, v_{(j,x)})$ would have also satisfied the constraint (and corresponds to choosing element x in the Hitting Set). Then, for any other constraint between the two rows, $(v_{(l,x)}, v_{(j,y)})$ will satisfy it only if $(v_{(l,x)}, v_{(j,x)})$ will. Hence an optimal algorithm may choose edges in one-to-one correspondence with the elements in the original Hitting Set instance.

Because Hitting Set is the complement of Set Cover, if P$\neq$NP, its optimal approximation ratio is $\Omega(\log(n))$ [13], and there are $\Theta\binom{n^3}{2}$ pairs of Hitting Set instances (or rows). The optimal solution has $\left(n^3\binom{n}{2} + \mathrm{OPT}\binom{n^3}{2}\right)$ edges – the first term counts the pairwise constraints in each row. So unless P=NP, the best polynomial time algorithm will require $\left(n^3\binom{n}{2} + \Omega\left(\log(n)\mathrm{OPT}\binom{n^3}{2}\right)\right)$ edges, giving us the result. □

Below, we give an algorithm that almost meets this lower bound, even in the arbitrary cost case when r is polynomial in n.

Theorem 2. *There is a polynomial time $O(\log(r))$-approximation algorithm for the **Network Inference** problem on n nodes and r constraints.*

Proof. The inputs are the vertices $V = \{v_1, v_2, \ldots, v_n\}$, the cost c_e of each edge $e = (v_i, v_j)$, and the constraints $\{S_1, S_2, \ldots, S_r\}$. Let $C(E)$ be a potential function that takes in a set of edges and sums over all constraints S_i, 1 minus the number of components S_i induces on (V, E). Let E be initially empty. Now, consider the following greedy algorithm: until all constraints are satisfied (while $C(E) < 0$), greedily add to E the edge that is $\arg\max_e \frac{C(E+e)-C(E)}{c_e}$.

We now notice that $C(E)$ is sub-modular in its edge set – as in, if $A \subseteq B$ then for all e, $C(A+e) - C(A) \geq C(B+e) - C(B)$. This is clear because A can induce additional components for e to reduce compared to B. Now we use the result of Wolsey [19] that says that a greedy algorithm for maximizing an integer-valued submodular set function f on elements x gives a $H(m)$ approximation to the optimum of f, where $m = \max_x f(\{x\})$ and $H(m) = \sum_i^m \left(\frac{1}{i}\right)$. Because each edge can increase the value of C by at most r, we have $m \leq r$, giving an $O(\log r)$ approximation. □

3 Online Network Inference

Oftentimes, the learner must commit to its choices as constraints arrive. For example, when the constraints represent diseases, we may wish to commit resources to fight an epidemic. This leads us to consider the natural extension of network inference to the online setting.

In the online setting, the collection of connectivity constraints $S_1, S_2, \ldots, S_r$ is now given one at a time, and we say that upon being presented S_i the algorithm is on **round** i. Also, let E_i be the edge set after the algorithm satisfies constraint S_i. To explore the worst-case performance of our algorithms, unless otherwise stated, we assume an **adaptive adversary**, meaning that the adversary can wait for the algorithm to satisfy constraint S_i before determining constraint S_{i+1}.

In this section, we are interested in competitive analysis. An algorithm is c-**competitive** if the cost of its solution is less than c times OPT, where OPT is the best solution in hindsight. In the case when we know that the underlying graph is, for instance, a uniform cost path or star, we know that $\text{OPT} = (n-1)$.

First, we prove a lemma helpful for analyzing online algorithms.

Lemma 1. *Let $n(G, S)$ be the number of connected components $S \subseteq V$ induces in G, and let $G_i = (V, E_i)$. For every algorithm for the* ***Online Network Inference*** *problem, there is an algorithm that performs at least as well and adds exactly $(n(G_i, S_{i+1}) - 1)$ edges on every round i.*

Proof. Let A be any algorithm for the online network inference problem. We can make a new algorithm called A_{lazy}, that on each round inserts only a subset of edges that A has inserted up to that round, enough to keep the constraints satisfied. Each edge that A inserts but A_{lazy} does not, A_{lazy} remembers as possible edges for future rounds and adds them as needed to satisfy future constraints. It is clear that A_{lazy} needs to put down a spanning tree on the components induced by constraint i, which is $(n(G_i, S_{i+1}) - 1)$ edges; any fewer edges would not satisfy the constraint. Thereby, A_{lazy} satisfies the constraints, and because A_{lazy} uses a subset of the edges of A, it performs at least as well. □

3.1 Stars and Paths

First, we examine the case when the underlying graph is a star. This represents the extreme case of one influential agent having many connections.

Theorem 3. *The optimal competitive ratio for the* ***Online Uniform Cost Network Inference*** *problem on n nodes when the algorithm knows the underlying graph is a* ***star*** *is $\Theta(\log(n))$.*

Proof. We first prove the *lower bound* – that the competitive ratio for any algorithm is $\Omega(\log(n))$. The adversary maintains a partition of the vertices into two sets: C, the possible centers, and $D = (V - C)$, the non-centers. Initially C has $(n-1)$ vertices and D has one vertex, and the initial two constraints given to the algorithm are V and C. At every step, the adversary looks for a vertex $v \in C$ that maximizes the number of edges (u, v) with $u \in D$ given by the algorithm, and moves v from C to D, that is, $C' = C \setminus \{v\}$ and $D' = D \cup \{v\}$. The new constraints given by the adversary are $C' \cup u$ for all $u \in D'$. Thus, the algorithm must ensure at least one edge from each element of D' to some element of C'. The adversary continues until it has moved all but one vertex from C to D.

To analyze, we consider the edges from elements of D to the element v moved from C to D when $|C| = i$. Each element of D must have at least one edge to an element of C, so the maximum number of edges from D to one element of C is at least the average: $(n-i)/i$. These edges are all distinct, so the algorithm must produce at least $\sum_{i=2}^{n-1}(n-i)/i = \Omega(n\log(n))$ edges in all. Yet, all these constraints can be satisfied by a star with $(n-1)$ edges. This completes the proof of the lower bound.

For the *upper bound*, we give an $O(\log(n))$-competitive algorithm. The algorithm will keep track of a set C_i of potential centers and $D_i = V - C_i$ known non-centers at round i. Any node not appearing in some constraint cannot be a center. The algorithm keeps nodes in C_i connected by a path, and each node in D_i is connected to some node in C_i, such that the number of edges going into each node in C_i from D_i is no more than $\lceil(|D_i|)/|C_i|\rceil$, meaning that all nodes in C_i have close to the same degree. Initially, $C_0 = V$ and is connected by an arbitrary path (costing $O(n)$ edges). At any stage of the algorithm, when a constraint S_i comes in, if it does not eliminate any potential centers, it is easy to see S_i is already satisfied. Otherwise, we remove any potential centers $R_i \subset C_{i-1}$ that are now known to be non-centers from C_{i-1} (to form C_i), and we add them to D_{i-1} (to form D_i). Further, we ignore all edges to nodes in R_i. We re-stitch the path connecting nodes in C_i, which takes at most $|R_i| + 1$ edges. Then, we connect (in such a way that keeps the degrees of the nodes in C_i about equal) all nodes in R_i to nodes in C_i, which takes $|R_i|$ edges, and also all nodes in D_{i-1} that became disconnected from C_i because were connected to nodes in R_i, which takes $O\left(\frac{|R||D_{i-1}|}{|C_{i-1}|}\right)$ edges. This clearly satisfies constraint S_i.

To see why this gives us the needed result, we notice that at most n centers can be removed from C, and therefore connections involving nodes in R_i take $\sum_{i=1}^{n} O(|R_i|) = O(n)$. The rest of the connections, by the analysis in the paragraph above, cost $\sum_i O\left(\frac{|R||D_{i-1}|}{|C_{i-1}|}\right) \leq \sum_i O\left(\frac{|R_i|n}{|C_{i-1}|}\right)$. If we consider removing one center at a time (as opposed to in groups R_i), we can bound this from above by $O(n\sum_{i=1}^{n}\frac{1}{n-i}) = O(n\log(n))$. □

Next, we examine another natural structure – when the underlying graph is a path.

Theorem 4. *The optimal competitive ratio for the* ***Online Uniform Cost Network Inference*** *problem on n nodes when the algorithm knows the underlying graph is a* ***path*** *is* $\Theta(\log(n))$.

Proof. First we prove the *lower bound*, that any algorithm has a competitive ratio of $\Omega(\log(n))$. We show an adversarial strategy that forces the algorithm to use $O(n\log(n))$ edges when the optimal solution in hindsight uses only $(n-1)$ edges. The adversary first shows all the nodes, which by Lemma 1 the optimal algorithm connects using $(n-1)$ edges. Then the adversary divides the nodes into two independent sets and presents each of them to the algorithm in arbitrary order. The optimal algorithm must connect the two subgraphs with trees (again by Lemma 1), and the adversary repeats this process recursively. We say that each depth in the recursion is a new level in this process. Because the algorithm puts down $O(n)$ edges per level, given this strategy for the adversary, the optimal algorithm needs to put down a path at each step so as to balance the sizes of two following independent sets and limit the algorithm to $O(\log(n))$ levels. Hence, the algorithm uses $\Omega(n\log(n))$ edges, but it is clear that knowing the sets in advance, one can satisfy the connectivity requirements using $O(n)$ edges - by simply connecting the smallest sets and then merging them accordingly into a path. This gives us the desired $\Omega(\log(n))$ gap.

Now we prove the *upper bound* by giving an $O(\log((n))$-competitive algorithm. We first observe that every constraint S_i is a sub-interval of the path, and the algorithm must put down enough edges to capture a permutation of the vertices consistent with the S_i's. The algorithm we introduce maintains a **pq-tree** – a data structure, introduced by Booth and Lueker in [10], that keeps track of all consistent orderings of nodes given contiguous intervals in a permutation. A pq-tree is a tree that consists of leaf nodes, p-nodes, and q-nodes. A **leaf node** is an element (or vertex in our case). A **p-node** (permutation node) has 2 or more children of any type, and its children form a contiguous interval, but can be ordered in any order. A **q-node** has 3 or more children of any type and its children form an interval in the given order or its reverse. Each new interval constraint updates the pq-tree, and then the algorithm adds edges to satisfy the new constraint.

We will show that the algorithm can satisfy the constraints using $O(n\log(n))$ edges by using a potential function to keep track of the evolution of the pq-tree. Let P be the set of p-nodes in a given tree and Q be the set of q-nodes. Also for any node p, let $c(p)$ count p's children. For constants a and b, our potential function is

$$\Phi = a\sum_{p\in P}((c(p)-1)(\log(c(p)-1)) + b|Q|. \tag{2}$$

We observe that the pq-tree before any constraints arrive has one p-node at the root, and all its children are leaf nodes. This corresponds to an arbitrary permutation of the vertices. So at the beginning, $\Phi = \Theta(n\log(n))$. In comparison, when the permutation is specified, the root is a q-node and the rest of the nodes are leaves. In that case, $\Phi = \Theta(1)$.

Now we look at what happens when a constraint comes in. We will argue that the number of edges we need to insert into our graph is a lower bound on the drop in the potential function, and because it is always the case that $\Phi \geq 0$, this will complete the proof.

We first analyze the most common type of update to a pq-tree. A constraint comes in and splits a known interval into two, that is, it splits a p-node with m

children into two p-nodes (one with at most $(k+1)$ children and the other with at most $(m-k)$ children), and attaches them to a q-node parent. So the drop in the potential function is as follows (where $H(p)$ is the binary entropy function.).

$$\begin{aligned}
-\Delta\Phi &= a\left((m-1)\log(m-1) - (k\log(k) + (m-k-1)\log(m-k-1))\right) - b \\
&= a(m-1)\left(\log(m-1) - \frac{k}{m-1}\log(k) - \frac{m-k-1}{m-1}\log(m-k-1)\right) - b \\
&= a(m-1)\left(-\frac{k}{m-1}\log\left(\frac{k}{m-1}\right) - \frac{m-k-1}{m-1}\log\left(\frac{m-k-1}{m-1}\right)\right) - b \\
&= a(m-1)H\left(\frac{k}{m-1}\right) - b \\
&\geq a(m-1)\min\left(\frac{k}{m-1}, \frac{m-k-1}{m-1}\right) - b \\
&= a\min(k, m-k-1) - b.
\end{aligned}$$

Now, $2\min(k, m-k-1)$ is exactly how much is required in the worst case to stitch up a split interval – because we have to connect up all of the nodes in the smaller new interval, and patch at most as many gaps in the larger interval (similar to the reasoning in the proof of the lower bound). It takes at most 4 more edges to connect up the ends of the two new intervals to the rest of the graph, and this can be paid for if $a = 10$ and $b = 4$. We remember $\min(k, m-k-1) \geq 1$, so we spend 2 on splitting the p-node, 4 on re-stitching, and 4 on the new q-node, and thus $a = 10$.

Booth and Lueker in [10] characterized all of the possible updates to the pq-tree using 10 patterns: L, P1, P2, P3, P4, P5, P6, Q1, Q2, and Q3, given in the Appendix. Neither L, Q1, nor P1 changes the number of p-nodes or q-nodes. P2-P6 split at most one p-node and create at most one q-node, and are covered by our analysis above. Q2 and Q3 require us to reconnect at most 2 pairs of endpoints (with 4 edges), but also reduce the number of q-nodes by 1 or 2 (this is why $b = 4$), and the edges are paid for by the drop in Φ. □

In the arbitrary cost case, the competitive ratio becomes considerably worse.

Theorem 5. *There is no (cn)-competitive algorithm for $c < 1$ for the* ***Online Network Inference*** *problem on n vertices, even when the underlying graph is a* ***path****.*

Proof. We let all edges among $(n-1)$ of the vertices have cost 0, and all edges from the remaining vertex, s, have cost 1. The adversary first tells the algorithm that all the vertices are connected. When the algorithm satisfies this constraint, the adversary excludes from the next constraint all vertices the algorithm has chosen to directly connect to s. This continues until the adversary forces the algorithm to use all the 1 edges. But because each constraint is a subset of the previous constraint, the optimal solution only needs to contain the final cost 1 edge, and can connect the remaining vertices using a path that goes through the vertices in the order they were excluded in the adversary's choice of constraints.

Hence, the algorithm was forced to pay a cost of $(n-1)$, while the optimal solution pays a cost of 1. □

3.2 General Graphs

We introduce the **Online Fractional Network Inference** problem, in which the algorithm is similarly given a set of vertices V and edge costs c_e for all $e = (v, w) \in V$, and sees a sequence of constraints $\{S_1, S_2, \ldots, S_r\}$. The task is to assign fractional weights w_e to the edges (or pairs of vertices), such that for each i, the maximum flow between each pair of vertices in S_i is at least 1, given the weights w_e (to be interpreted as edge capacities). The quality of the solution is measured by comparing $\sum c_e w_e$ with the optimal cost of satisfying all the connectivity constraints. In the online problem, the algorithm may not decrease any edge weights from round to round.

Lemma 2. *There is an $O(\log(n))$-competitive polynomial time algorithm for the* ***Online Fractional Network Inference*** *problem on n nodes.*

Proof. We give Algorithm 1 for the Online Fractional Network Inference problem. Algorithm 1 is a modification of the algorithm in **3.1** of Alon *et al.* [5], and this proof closely follows their logic.

Algorithm 1. An $O(\log(n))$-competitive Algorithm for the Online Fractional Network Inference Problem

```
Let $|V| = n$ and $|E| = m$
Upon seeing first constraint, set all $w_e = \frac{1}{m^2}$
for each constraint $S$ do
  for each pair $v, w \in S$ do
    if the flow from $v$ to $w$ in $S$ is at least 1 then
      do nothing
    else
      while the flow from $v$ to $w$ in $S$ is less than 1 do
        compute a min-weight cut $C$ between $v$ and $w$ in $S$.
        for each edge $e \in C$, $w_e = w_e(1 + 1/c_e)$
      end while
    end if
  end for
end for
```

We say that the optimal solution OPT has cost α. We assume the value of α is known, and we can then assume all edges have cost between 1 and m.[2] We now

[2] Alon *et al.* [5] argue that we can use all edges of cost less than α/m and stay within our bound, and we can ignore all edges with cost greater than α, and then rescale. They also show how to guess α to within a factor of 2, justifying the assumption that α is known in advance.

follow the argument in Alon *et al.* [5], which works for Algorithm 1 almost without modification. First we note that the algorithm generates a feasible solution. This is clear from its termination condition.

Now we will prove that the number of weight augmentation steps performed during the run of the algorithm is $O(\alpha \log(m))$. Consider the potential function

$$\Phi = \sum_{e \in E} c_e w_e^* \lg(w_e),$$

where w_e^* is the weight of edge e in OPT. It is clear from the initial edge weights that the potential function begins as $\Phi_0 = -O(\alpha \lg(m))$. Because the weight update rule ensures that no edge gets weight more than 2, the potential function never exceeds 2α. And the increase in the potential function with each weight augmentation step is at least 1:

$$\begin{aligned}\Delta\Phi &= \sum_{e \in E} c_e w_e^* \lg(w_e(1 + 1/c_e)) - \sum_{e \in E} c_e w_e^* \lg(w_e) \\ &= \sum_{e \in E} c_e w_e^* \lg(1 + 1/c_e) \\ &\geq \sum_{e \in E} w_e^* \\ &\geq 1.\end{aligned}$$

Finally, we look at the cost of our solution, $\sum_{e \in E} w_e c_e$, (which begins at ≤ 1) and notice that in a weight augmentation step, it does not exceed $\sum_{e \in E} \frac{w_e}{c_e} c_e \leq 1$. So, whenever Φ increases by at least 1, the cost of our solution increases by no more than 1. This gives us an $O(\log(m)) = O(\log(n))$ approximation to the Online Fractional Network Inference problem. □

We can now use Lemma 2 to develop an algorithm that almost matches the lower bound from Theorem 5.

Theorem 6. *There is an $O(n \log(n))$-competitive polynomial time algorithm for the* ***Online Network Inference*** *problem on n nodes.*

Proof. We use Algorithm 1 together with a rounding scheme similar to the one considered by Buchbinder [11] for solving linear programs, to get our result.

For each edge e, we choose $2n$ random variables $X(e, i)$ independently and uniformly from $[0, 1]$. For each edge, we let threshold $T(e) = \min_{i=1}^{2n} X(e, i)$. Then we run the algorithm for the Online Fractional Network Inference problem, and whenever $w_e \geq T(e)$, we add e to our integral solution, and continue. Now we claim the following.

1. The integral solution has expected cost $O(n)$ times the fractional solution.
2. The integral solution satisfies all the constraints with high probability.

To prove the first claim, for any edge e, the probability that $X(e, i) < w_e$ is w_e. The probability that e is chosen to be in the integral solution is the probability that some $X(e,i) < w_e$ – we call this event A_i. Hence, the probability of $\cup_{i=1}^{2n} A_i$ is $2nw_e$, and by linearity of expectation, on every round, the expected cost of our solution is $O(n)$ times the fractional solution, which is an $O(\log(n))$ approximation of OPT for the integral problem. Hence our solution is an $O(n\log(n))$ approximation of OPT in expectation.

To prove the second claim, we pick a constraint S. The constraint S is satisfied if and only if for every cut $C \in S$, there exists an edge crossing C in our solution. We fix a cut C. The probability the cut is not crossed is the probability we have not chosen any edge crossing the cut. This probability is $\prod_{e\in C}(1-w_e)^{2n} \leq e^{\left(-2n\sum_{e\in C} w_e\right)}$. And because the cut is crossed with a flow of 1 in the fractional solution (i.e. $\sum_{e\in C} w_e \geq 1$) at the time it is considered by the algorithm, we can bound this by $\frac{1}{e^{2n}}$. There are r constraints and at most 2^n cuts per constraint, so by the union bound, the probability our solution is not feasible is $\left(\frac{r2^n}{e^{2n}}\right)$. Because $r < 2^n < e^n$, the probability our solution is not feasible tends to 0 as n increases, completing the proof. □

We note that it is tempting to try to improve the bound for the Online Network Inference problem by reducing it to Online Set Cover with the hope of getting better bounds by using algorithms from [4] or [11]. In Online Set Cover, the sets to be covered are given in advance, and the elements come in online. In Online Network Inference, the graph is known in advance, but there are exponentially many constraints in the size of the graph that can arrive online. While a reduction that makes edges in the network graph correspond to sets and all cuts induced by the connectivity constraints correspond to elements in the Online Set Cover problem is possible, the resulting bound on the competitiveness ratio is $O(n\log(n))$, which yields no improvement over our algorithm.

We now make a simple observation for the uniform cost case.

Proposition 1. *There is an $O(n)$-competitive polynomial time algorithm for the* ***Online Uniform Cost Network Inference*** *problem on n nodes.*

Proof. Consider the algorithm that puts down a clique for each constraint presented to it. Let $q \leq n$ be the number of nodes that appear in at least one constraint. Our algorithm uses $O(q^2)$ edges, but the optimal algorithm must clearly use at least $\Omega(q)$ edges. □

We also present a lower bound for the online uniform cost case.

Theorem 7. *The* ***Online Uniform Cost Network Inference*** *problem on n nodes has an $\Omega(\sqrt{n})$-competitive lower bound.*

Proof. We divide the vertices into two sets Q and R, with $|Q| = \sqrt{n}$ and $|R| = n - \sqrt{n}$. For each $v_i \in R$, the adversary does the following. At stage $t = 1$ the adversary sets $Q_{(i,1)} = Q$. At stage t, the adversary gives the learner the constraint $S_{(i,t)} = Q_{(i,t)} \cup v_i$. Let $C_{(i,t)}$ be the set of vertices in Q which the learner connects to v_i in response to being presented $S_{(i,t)}$. The adversary sets $Q_{(i,t+1)} = Q_{(i,t)} \setminus C_{(i,t)}$ and continues to the next stage. The adversary stops when $Q_{(i,t)} = \emptyset$.

To analyze this strategy for the adversary, for each v_i, we order the edges from v_i to R by the stage in which the learner has placed them, breaking ties arbitrarily. It is clear that the last edge the learner places is sufficient to connect v_i to R for all constraints $S_{(i,t)}$. Hence, all of these constraints can be satisfied in retrospect by placing a clique on Q using $\binom{\sqrt{n}}{2} = O(n)$ edges and one edge per vertex in R, also using $O(n)$ edges. But the learner places $\Omega(n)$ edges per vertex in Q, amounting to $\Omega\left(n\sqrt{n}\right)$ edges in total, giving the desired result. □

We now consider the Online Network Inference problem with an **oblivious adversary** – an adversary who commits to the constraints $\{S_1, S_2, \ldots, S_r\}$ before presenting any of them to the algorithm.

Theorem 8. *There is a randomized polynomial time algorithm for the* ***Online Uniform Cost Network Inference*** *problem on n nodes that gives an expected $O(n^{2/3}\log^{2/3}(n))$-competitive ratio against an oblivious adversary.*

Proof. We assume that the optimal solution has $m = \Omega(n)$ edges (that each vertex appears in some constraint). We then create an Erdös Rényi random graph on our graph G, by putting in edges independently with a specified probability. Random graph connectivity has a sharp threshold of $\frac{c\log(n)}{n}$ for $c > 1$ [12]. When $p = \frac{c\log^{2/3}(n)}{n^{1/3}}$, G has $O(n^{5/3}\log^{2/3}(n))$ edges in expectation. Now, our algorithm is simple – for each constraint S_i such that $|S_i| \geq n^{1/3}\log^{1/3}(n)$, because of our choice of p, S_i is already connected with high probability in G. Because we assume that there are only polynomially many constraints (even in the offline case, as in Theorem 2), for large enough c, all such constraints are satisfied in expectation. For every constraint S_i of size $< n^{1/3}\log^{1/3}(n)$ that we see, we can put a clique with $O(n^{2/3}\log^{2/3}(n))$ edges on that constraint, and each time we do that, we are guaranteed to hit at least one edge in OPT. Hence, this costs us $O(n^{5/3}\log^{2/3}(n) + n^{2/3}\log^{2/3}(n)\mathrm{OPT})$ edges in expectation, and because $m = \Omega(n)$, we have an $O(n^{2/3}\log^{2/3}(n))$ approximation ratio. □

4 Discussion and Open Problems

In this paper we present a theoretical study of the Network Inference problem. This model allows us to estimate connections among populations from data that exposes certain constraints. One challenge in using this model to learn real-world networks is that oftentimes, due to issues of privacy, data is anonymized (for example disease data), and it is hard to tell when the same person participates in multiple constraints. However, there are other settings where this would not be an issue. For network construction problems, our algorithms give network designers methods of optimizing costs while satisfying their users' constraints.

We leave open some interesting questions. In the offline case, we give an $\Omega(\log(n))$ hardness of approximation lower bound and an $O(\log(r))$ approximation algorithm for both the arbitrary cost and uniform cost Network Inference problems. If r is polynomial in n these bounds match, but otherwise there can be a gap. We also have a $\log(n)$ asymptotic gap for the Online Network Inference problem. For the Online Uniform Cost Network Inference problem, we have an $\Omega(\sqrt{n})$ adversarial lower bound and an $O(n^{2/3}/log^{1/3}(n))$ algorithm for the oblivious case. Improving these bounds is an important problem.

Another open problem is to find tight bounds for trees in the uniform cost case. For stars and paths, the bounds are tight, and our arguments can be adapted to give a $\Omega(\log(n))$-competitive lower bounds against an oblivious adversary. Perhaps an $O(\log(n))$-competitive algorithm can be found for trees in general, but our algorithms for paths and stars rely on their specific properties and do not immediately generalize. Finally, one can consider generalizations of the Network Inference Problem, for example constraints could require the vertices to be k-connected in the induced subgraphs.

Acknowledgments

We thank the anonymous reviewers of this paper for helpful suggestions. We thank an anonymous reviewer of a previous version of this paper for helping improve our analysis of Theorem 2. Parts of this work were done while Lev Reyzin was in the Department of Computer Science at Yale University.

References

[1] Akutsu, T., Tamura, T., Horimoto, K.: Completing networks using observed data. In: ALT, pp. 126–140 (2009)
[2] Alon, N., Asodi, V.: Learning a hidden subgraph. SIAM J. Discrete Math. 18(4), 697–712 (2005)
[3] Alon, N., Awerbuch, B., Azar, Y., Buchbinder, N., Naor, J.: The online set cover problem. In: Proceedings of the 35th annual ACM symposium on Theory of computing, pp. 100–105 (2003)
[4] Alon, N., Awerbuch, B., Azar, Y., Buchbinder, N., Naor, J.: The online set cover problem. In: STOC, pp. 100–105 (2003)
[5] Alon, N., Awerbuch, B., Azar, Y., Buchbinder, N., Naor, J.: A general approach to online network optimization problems. ACM Transactions on Algorithms 2(4), 640–660 (2006)
[6] Alon, N., Beigel, R., Kasif, S., Rudich, S., Sudakov, B.: Learning a hidden matching. SIAM J. Comput. 33(2), 487–501 (2004)
[7] Angluin, D., Aspnes, J., Reyzin, L.: Optimally learning social networks with activations and suppressions. In: 19th International Conference on Algorithmic Learning Theory, pp. 272–286 (2008)
[8] Angluin, D., Chen, J.: Learning a hidden graph using $O(\log n)$ queries per edge. J. Comput. Syst. Sci. 74(4), 546–556 (2008)
[9] Beigel, R., Alon, N., Kasif, S., Apaydin, M.S., Fortnow, L.: An optimal procedure for gap closing in whole genome shotgun sequencing. In: RECOMB, pp. 22–30 (2001)
[10] Booth, K.S., Lueker, G.S.: Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. J. Comput. Syst. Sci. 13(3), 335–379 (1976)

[11] Buchbinder, N.: Designing Competitive Online Algorithms Via A Primal-Dual Approach. PhD thesis, Technion – Israel Institute of Technology, Haifa, Israel (2008)
[12] Erdös, P., Rényi, A.: On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci. 5, 17–61 (1960)
[13] Feige, U.: A threshold of ln for approximating set cover. J. ACM 45(4), 634–652 (1998)
[14] Grebinski, V., Kucherov, G.: Reconstructing a Hamiltonian cycle by querying the graph: Application to DNA physical mapping. Discrete Applied Mathematics 88(1-3), 147–165 (1998)
[15] Gupta, A., Krishnaswamy, R., Ravi, R.: Online and stochastic survivable network design. In: STOC, pp. 685–694 (2009)
[16] Korach, E., Stern, M.: The clustering matroid and the optimal clustering tree. Mathematical Programming 98(1-3), 345–414 (2003)
[17] Korach, E., Stern, M.: The complete optimal stars-clustering-tree problem. Discrete Applied Mathematics 156(4), 444–450 (2008)
[18] Reyzin, L., Srivastava, N.: Learning and verifying graphs using queries with a focus on edge counting. In: Hutter, M., Servedio, R.A., Takimoto, E. (eds.) ALT 2007. LNCS (LNAI), vol. 4754, pp. 285–297. Springer, Heidelberg (2007)
[19] Wolsey, L.A.: An analysis of the greedy algorithm for the submodular set covering problem. Combinatorica 2(4), 385–393 (1982)

Appendix: Updating a pq-Tree

We briefly describe the patterns in [10] for updating pq-trees, as broken down into 10 cases. This can be used as a guide for tracking the changes in Equation 2.

L This pattern simply relabels some leaf nodes.
P1 This pattern simply relabels a p-node.
P2 This pattern moves some children of a p-node into their own p-node.
P3 This pattern moves some children of a p-node into their own p-node and creates a parent q-node.
P4 This pattern moves some children of a p-node to be children of a newly created p-node, whose parent is a q-node that is a child of the original p-node.
P5 This pattern moves some children of a p-node into their own p-node that is the child of the original p-node, which becomes transformed to a q-node.
P6 This pattern moves some children of a p-node to their own p-node that is moved to be the child of a newly created q-node formed by merging two q-nodes.
Q1 This pattern simply relabels a q-node.
Q2 This pattern deletes a q-node and moves its children to become children of its parent q-node.
Q3 This pattern deletes two q-nodes and merges their children to become children of their parent q-node.

Distribution-Dependent PAC-Bayes Priors

Guy Lever[1], François Laviolette[2], and John Shawe-Taylor[1]

[1] University College London
[2] Université Laval

Abstract. We develop the idea that the PAC-Bayes prior can be informed by the data-generating distribution. We prove sharp bounds for an existing framework, and develop insights into function class complexity in this model and suggest means of controlling it with new algorithms. In particular we consider controlling capacity with respect to the *unknown* geometry of the data-generating distribution. We finally extend this localization to more practical learning methods.

1 Introduction

The paper takes its inspiration from Catoni (2007), who developed localised PAC-Bayes analysis by using a prior defined in terms of the data generating distribution. At first sight this might appear to be 'cheating', since we must define the prior before seeing the data. However, by defining in terms of the distribution we avoid this difficulty since the distribution itself can be considered as fixed before the sample is generated. PAC-Bayes bounds are one of the sharper analyses of the learning process. A weakness in the standard PAC-Bayes approach is that analysis is constrained by the choice of prior distribution, since the divergence between prior and posterior forms part of the bound. This choice of prior tends to be rather generic; typically not tailored to the particular problem, so that, in particular, good classifiers do not generally receive large prior weight. Thus the divergence term in the PAC-Bayes analysis can typically be large. By tuning the prior to the distribution Catoni is able to remove the Kullback-Leibler (KL) term from the bound hence significantly reducing the complexity penalty.

The paper begins by using Catoni's definition of the prior involving a Boltzmann distribution, but proves a new sharp bound (Theorem 3) using a new lemma (Lemma 1) and the re-use of the PAC-Bayes bound to remove the KL term (Lemma 2). The resulting bound suggests a new complexity parameter γ that enters as a $\gamma/m^{3/2}$ term (where m is the sample size). This opens a potential new direction in generalization analysis of learning machines.

In our context this suggests the need to regularize in this learning method. The flexibility of the framework we develop is that it allows us to encode our prior meta-assumptions about how we anticipate a good classifier will interact with the data; we can control capacity, for example, with respect to the smoothness of a classifier over the *unknown* data generating distribution thus giving high weight to classifiers that are smooth over the manifold defined by the support of the data distribution. The analysis is achieved with a novel PAC-Bayes bound on U-statistics estimation.

The final section of the paper covers the third main theme which is the extension of the data distribution dependent priors to the Gaussian prior and posterior PAC-Bayes

© Springer-Verlag Berlin Heidelberg 2010

bounds for, for example, SVMs (Langford & Shawe-taylor, 2002). Here we are able to remove the KL term again leaving a term that only involves a similar complexity parameter γ appearing as $O(\gamma/(\eta^2 m^2))$, where η is the regularization parameter, in contrast to the usual $O(\|\mathbf{w}\|^2/m)$. This again suggests a new measure of complexity for SVM classifiers with the possibility of using the bound to optimise the regularization parameter.

We now review the relation of our approach to earlier work. The luckiness framework explored the possibility that we could learn the hierarchy of classes of hypotheses from the data as part of the learning process giving rise to so-called data-dependent structural risk minimization (Shawe-Taylor et al., 1998). The most successful example of this approach was large margin classification such as support vector machines. However, although we cannot measure a margin without seeing the data, by moving to real-valued functions, we can equate large margin with small norm when we constrain $y_i f(\mathbf{x}_i) \geq 1$ on the training data, $i = 1, \dots, m$, resulting in a fixed prior. Nonetheless this is equivalent to placing a prior over the classifiers in terms of the data generating distribution, that is we favour hyperplanes that have low input density in the slab defined by shifting the decision boundary parallel to itself by $\pm\gamma$.

Further research in this direction has been developed by Balcan and Blum (Balcan & Blum, 2010) with their notion of compatibility, which is used to restrict the hypotheses considered in the learning process to those satisfying a given level of compatibility *estimated from the training data*, hence reducing the effective complexity of the class. Perhaps less well-known is work by Catoni (Catoni, 2007) where he introduces 'localised' PAC-Bayes analysis effectively defining the prior in terms of the data-generating distribution in a PAC-Bayes bound on generalization.

We should finally distinguish between distribution defined priors and using part of the data to learn a prior and the rest to learn the function (Ambroladze et al., 2006).

2 Preliminaries

We consider the general setting in which we are given a sample of labelled and unlabelled[1] points $\mathcal{S} = \{(X_1, Y_1), ...(X_m, Y_m)\} \cup \{X_{m+1}, ...X_n\}$ drawn i.i.d. from distribution D (with density d) over $\mathcal{X} \times \mathcal{Y}$, the product space of labelled inputs (or its marginalization to $\mathcal{X}$). Our analysis therefore includes the settings of supervised and semi-supervised learning and some transductive settings. We are interested in the case where $\mathcal{Y} = \{-1, +1\}$, and study binary classification.

We are interested in the notion of *risk* of a hypothesis $h \in \mathcal{H}$,

$$\mathrm{risk}^{\ell}(h) := \mathbb{E}_{(X,Y)\sim D}\,\ell(h(X), Y),$$

and its empirical counterpart on a labelled sample $\mathcal{S}$,

$$\widehat{\mathrm{risk}}^{\ell}_{\mathcal{S}}(h) := \frac{1}{|\mathcal{S}|} \sum_{(X,Y)\in\mathcal{S}} \ell(h(X), Y),$$

[1] Throughout, the unlabelled set may be empty.

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is some loss function. When $\ell(\cdot,\cdot)$ is the $0-1$ loss of binary classification, $\ell_{0-1}(y,y') := \begin{cases} 0 \text{ if } y = y' \\ 1 \text{ if } y \neq y' \end{cases}$, then for simplicity we denote the corresponding binary classification risk and its empirical counterpart by $\text{risk}(\cdot)$ and $\widehat{\text{risk}}_{\mathcal{S}}(\cdot)$ respectively. Our objective is to obtain a probabilistic guarantee on the true binary classification risk of a classifier by relating it to its empirical counterpart.

The PAC-Bayes bounds apply to a stochastic Gibbs classifier G_Q drawn from a distribution Q over a hypothesis class $\mathcal{H}$. We denote $\text{risk}(G_Q) := \mathbb{E}_{h\sim Q}\text{risk}(h)$. The following quantities feature in these bounds: the Kullback-Leibler divergence between distributions Q and P, and its specialization to Bernouilli distributions,

$$\text{KL}(Q||P) := \mathbb{E}_{h\sim Q} \ln \frac{dQ}{dP}(h), \quad \text{kl}(q,p) := q\ln\frac{q}{p} + (1-q)\ln\frac{1-q}{1-p} \quad q,p \in (0,1),$$

and we define

$$\xi(m) := \sum_{k=0}^{m} \binom{m}{k} \left(\frac{k}{m}\right)^k \left(1-\frac{k}{m}\right)^{m-k} \in [\sqrt{m}, 2\sqrt{m}].$$

The following is a generalization of (Germain et al., 2009, Th 2.1) and is proved using the same sequence of arguments.

Theorem 1. *For any functions $A(h)$, $B(h)$ over $\mathcal{H}$, either of which may be a statistic of the sample $\mathcal{S}$, any distributions P over $\mathcal{H}$, any $\delta \in (0,1]$, any $t > 0$, and a convex function $\mathcal{D} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ we have with probability at least $1-\delta$ over the draw of $\mathcal{S}$,*

$$\forall Q \,\text{on}\, \mathcal{H} : \mathcal{D}(\mathbb{E}_{h\sim Q}A(h), \mathbb{E}_{h\sim Q}B(h)) \leq \frac{1}{t}\left(\text{KL}(Q||P) + \ln\left[\frac{\mathcal{L}_P}{\delta}\right]\right),$$

where $\mathcal{L}_P := \mathbb{E}_{\mathcal{S}}\mathbb{E}_{h\sim P}\left[e^{t\mathcal{D}(A(h),B(h))}\right]$.

Theorem 1 is a recipe for generating a variety of PAC-Bayes bounds, by specializing to a choice for $\mathcal{D}(\cdot,\cdot)$, t, $A(\cdot)$ and $B(\cdot)$, and choosing P to be a "prior" (i.e. not sample-dependent) so that the order of expectation in the r.h.s. can be exchanged and evaluated. For example, one can derive Seeger's bound and a slightly relaxed version of Catoni's bound, which will be needed later:

Theorem 2. *(Seeger, 2002; Langford, 2005; Catoni, 2007) For any (unknown) distribution D, any set $\mathcal{H}$ of classifiers, any distribution P of support $\mathcal{H}$, any $\delta \in (0,1]$, and any positive constant C, we have, where $C^\star := \frac{C}{1-e^{-C}}$,*

$$\mathbb{P}_{\mathcal{S}}\left(\forall Q \,\text{on}\, \mathcal{H} : \text{kl}(\widehat{\text{risk}}_{\mathcal{S}}(G_Q), \text{risk}(G_Q)) \leq \tfrac{1}{m}\left(\text{KL}(Q||P) + \ln\tfrac{\xi(m)}{\delta}\right)\right) \geq 1-\delta$$
$$\mathbb{P}_{\mathcal{S}}\left(\forall Q \,\text{on}\, \mathcal{H} : \text{risk}(G_Q) \leq C^\star\left(\widehat{\text{risk}}_{\mathcal{S}}(G_Q) + \tfrac{1}{C\cdot m}\left(\text{KL}(Q||P) + \ln\tfrac{1}{\delta}\right)\right)\right) \geq 1-\delta$$

Note that the PAC-Bayes bound proposed by McAllester in his pioneer work on the subject (McAllester, 1999) can be retrieved from Seeger's bound using the inequality

$$2(q-p)^2 \leq \text{kl}(q,p). \tag{1}$$

Note also that the $\mathrm{KL}(Q||P)$ term can significantly deteriorate these bound if classifiers with small empirical risk receive low probability from the prior P, i.e. if the prior has been "badly" chosen. The data distribution-defined priors we investigate are specifically constructed to give large weight to classifiers with low true risk, and the KL-divergence between Q and P decays with the sample size.

2.1 Choosing a Distribution-Dependent Prior

Suppose an algorithm takes as input a training sample $\mathcal{S}$ from the distribution D^m over $\mathcal{Z}^m$ and outputs a posterior distribution Q over $\mathcal{H}$. Let $\mathcal{P}_{\mathcal{H}}$ be the space of probability distributions over $\mathcal{H}$, and in the interest of obtaining a good PAC-Bayes bound for Q, consider the minimizer of $\mathrm{KL}(Q||P)$ *in expectation*: $\mathrm{argmin}_{P\in\mathcal{P}_{\mathcal{H}}} \mathbb{E}_{\mathcal{S}}[\mathrm{KL}(Q||P)] = \mathbb{E}_{\mathcal{S}}[Q]$. The implication of this result is noted in this context in (Catoni, 2007) as is the immediate fact that the resulting divergence is equal to the mutual information, $I(h;\mathcal{S})$, between sample and classifier (considered as random variables drawn from the distribution $Q \times D^m$), $\mathbb{E}_{\mathcal{S}}[\mathrm{KL}(Q||\mathbb{E}_{\mathcal{S}}[Q])] = I(h;\mathcal{S})$. We note that PAC-Bayes analysis using this prior appears to be quite difficult since the prior can be difficult to manipulate. As suggested by Catoni we study other more flexible choices of prior which enable us to obtain very sharp PAC-Bayes bounds. We assume that there exists a reference measure μ on $\mathcal{H}$ (when $\mathcal{H}$ is of finite dimensionality this would typically be a uniform measure such as Lebesgue measure). We consider the case when the posterior and prior are from the exponential family, defined by their densities w.r.t. μ,

$$q(h) := \frac{dQ}{d\mu}(h) := \frac{1}{Z}e^{-F_Q(h)} \qquad\qquad p(h) := \frac{dP}{d\mu}(h) := \frac{1}{Z'}e^{-F_P(h)},$$

where F_Q, F_P are "energy functions", to be chosen, and $Z = \int_{\mathcal{H}} e^{-F_Q(h)}\mathrm{d}\mu$, $Z' = \int_{\mathcal{H}} e^{-F_P(h)}\mathrm{d}\mu$. We note the following upper bound on the KL divergence, which reduces obtaining a bound on the KL divergence to establishing a PAC-Bayesian concentration result for the energy functions.

Lemma 1

$$\mathrm{KL}(Q||P) \le (\mathbb{E}_{h\sim Q} - \mathbb{E}_{h\sim P})\,[F_P(h) - F_Q(h)]\,. \tag{2}$$

Proof

$$\begin{aligned}
\mathrm{KL}(Q||P) &= \mathbb{E}_{h\sim Q} \ln \frac{Z' e^{-F_Q(h)}}{Z e^{-F_P(h)}} \\
&= \mathbb{E}_{h\sim Q}\,[F_P(h) - F_Q(h)] - \ln \frac{\int e^{-F_Q(h)}\mathrm{d}\mu}{Z'} \\
&= \mathbb{E}_{h\sim Q}\,[F_P(h) - F_Q(h)] - \ln \int p(h) e^{F_P(h)-F_Q(h)}\mathrm{d}\mu \\
&\le (\mathbb{E}_{h\sim Q} - \mathbb{E}_{h\sim P})\,[F_P(h) - F_Q(h)]\,, \qquad (3)
\end{aligned}$$

where the final line follows from the convexity of $-\ln(\cdot)$. □

Note that the r.h.s. of (3) is precisely the type of quantity that PAC-Bayes theory provides a bound for. In particular, the choice $F_P = \mathbb{E}_{\mathcal{S}}[F_Q]$ seems natural and we remark that (3) is then reduced to obtaining a concentration inequality for F_Q to its expectation.

3 Stochastic Empirical Risk Minimization-Type Prediction

We first consider posterior and prior densities, w.r.t. μ, over $\mathcal{H}$ of the following forms:

$$q(h) = \frac{1}{Z} e^{-\left(\gamma\widehat{\text{risk}}_{\mathcal{S}}(h) + \eta F_Q(h)\right)} \tag{4}$$

$$p(h) = \frac{1}{Z'} e^{-(\gamma\text{risk}(h) + \eta F_P(h))}. \tag{5}$$

where the $F : \mathcal{H} \to \mathbb{R}$ are regularization functions, and Z a normalization constant. The unregularized case corresponds to "Gibbs algorithms", e.g. (Catoni, 2007). $F_Q(\cdot)$ and $F_P(\cdot)$ may be different and in particular we will consider the special case where $F_Q(\cdot)$ is a sample statistic, allowing us to perform data-dependent regularization.

We note that Lemma 1 implies the following upper bound on the KL divergence

$$\text{KL}(Q||P) \leq (\mathbb{E}_{h\sim Q} - \mathbb{E}_{h\sim P}) \left[\gamma\text{risk}(h) - \gamma\widehat{\text{risk}}_{\mathcal{S}}(h) + \eta F_P(h) - \eta F_Q(h)\right]. \tag{6}$$

As we will see later, for suitable choices of parameters γ and η, the divergence decays with the sample. We now consider several choices of $F_Q(\cdot)$ and $F_P(\cdot)$ and give PAC-Bayes bounds for the resulting Gibbs classifiers.

3.1 The Non-regularized Case: $\eta = 0$

We recall that the distribution D over $\mathcal{X} \times \mathcal{Y}$ is unknown, hence so is the prior distribution given by (5). To obtain a bound, we need to bound the KL divergence $\text{KL}(Q||P)$. With reference to (6), in the situation where $\eta = 0$ such an upper bound can be obtained given an upper bound for $\text{risk}(G_Q) - \widehat{\text{risk}}_{\mathcal{S}}(G_Q)$ and a lower bound for $\text{risk}(G_P) - \widehat{\text{risk}}_{\mathcal{S}}(G_P)$, and such bounds can obtained via Theorem 2.

Lemma 2. *Let P and Q be defined as in (4) and (5) with $\eta = 0$ then with probability at least $1 - \delta$, the following holds,*

$$\text{KL}(Q||P) \leq \frac{\gamma}{\sqrt{m}}\sqrt{\ln\frac{\xi(m)}{\delta}} + \frac{\gamma^2}{4m}.$$

Proof. From (1) and the Seeger bound of Theorem 2 (applied for the choices $Q = Q$ and $Q = P$) we obtain that, simultaneously,

$$\begin{aligned} \text{risk}(G_Q) - \widehat{\text{risk}}_{\mathcal{S}}(G_Q) &\leq \frac{1}{2\sqrt{m}}\sqrt{\text{KL}(Q||P) + \ln\frac{\xi(m)}{\delta}}, \\ -\left(\text{risk}(G_P) - \widehat{\text{risk}}_{\mathcal{S}}(G_P)\right) &\leq \frac{1}{2\sqrt{m}}\sqrt{\ln\frac{\xi(m)}{\delta}}. \end{aligned}$$

Together with (6) the last inequalities give

$$\begin{aligned}\mathrm{KL}(Q\|P) &\leq \gamma\left(\mathrm{risk}(G_Q)-\widehat{\mathrm{risk}}_{\mathcal{S}}(G_Q)\right)-\gamma\left(\mathrm{risk}(G_P)-\widehat{\mathrm{risk}}_{\mathcal{S}}(G_P)\right)\\ &\leq \frac{\gamma}{2\sqrt{m}}\sqrt{\mathrm{KL}(Q\|P)+\ln\frac{\xi(m)}{\delta}}+\frac{\gamma}{2\sqrt{m}}\sqrt{\ln\frac{\xi(m)}{\delta}}\,.\end{aligned}$$

If $\mathrm{KL}(Q\|P) \leq \frac{\gamma}{\sqrt{m}}\sqrt{\ln\frac{\xi(m)}{\delta}}$, we are done. Otherwise, by straightforward algebraic manipulations we then obtain the following inequality, which, together with the fact that $\mathrm{KL}(Q\|P) \geq 0$, directly implies the result.

$$\begin{aligned}(\mathrm{KL}(Q\|P))^2 - \frac{2\gamma}{2\sqrt{m}}\sqrt{\ln\frac{\xi(m)}{\delta}}\,\mathrm{KL}(Q\|P)+\frac{\gamma^2}{4m}\ln\frac{\xi(m)}{\delta}&\\ \leq \frac{\gamma^2}{4m}\mathrm{KL}(Q\|P)+\frac{\gamma^2}{4m}\ln\frac{\xi(m)}{\delta}&.\end{aligned}$$

□

Thus, Theorem 2 can be specialized to the following bound. (Note, for the first result no union bound is required since we need to apply Theorem 2 once only.)

Theorem 3. *Let P and Q be defined as in (4) and (5) with $\eta = 0$, then*

$$\mathbf{P}_{\mathcal{S}}\left(\mathrm{kl}(\widehat{\mathrm{risk}}_{\mathcal{S}}(G_Q),\mathrm{risk}(G_Q)) \leq \frac{1}{m}\left(\frac{\gamma}{\sqrt{m}}\sqrt{\ln\frac{\xi(m)}{\delta}}+\frac{\gamma^2}{4m}+\ln\frac{\xi(m)}{\delta}\right)\right) \geq 1-\delta,$$

$$\mathbf{P}_{\mathcal{S}}\left(\mathrm{risk}(G_Q) \leq C^{\star}\,\widehat{\mathrm{risk}}_{\mathcal{S}}(G_Q)+\frac{C^{\star}}{C\cdot m}\left(\frac{\gamma}{\sqrt{m}}\sqrt{\ln\frac{2\xi(m)}{\delta}}+\frac{\gamma^2}{4m}+\ln\frac{2}{\delta}\right)\right) \geq 1-\delta$$

Observe that for a large value of γ, the posterior Gibbs classifier G_Q will be concentrated on the classifiers of $\mathcal{H}$ with smallest empirical risk. Hence the two bounds of Theorem 3 are risk bounds for a type of stochastic empirical risk minimization algorithm. Since the KL-divergence term has been evaluated and is small, it appears that there is no component of the bound that depends on the complexity of the learning problem or the class of classifiers. In fact the parameter that controls the effective complexity is the "inverse temperature", γ (or γ^2 if we view it in the role of a VC dimension). If the problem is 'easy' in the sense that the measure of the set of classifiers with low empirical risk is not too small then a low value of γ will deliver low empirical risk for the Gibbs classifier. If, however, the measure of the classifiers that have low empirical risk is very small (as would be likely if the function class itself is large) then we require a larger value of γ before the Gibbs risk is controlled. The complexity that γ measures is related to the fit between input distribution and function class in that it will depend on the measure of the distribution Q on the low empirical risk functions.

In practice γ would need to be chosen from a grid Γ of values in response to the particular training problem. Hence, in order to apply the bound we would need to use the union bound over the $|\Gamma|$ applications of the bound resulting in an extra $\log(|\Gamma|)$ term in the right hand side brackets. Another possibility would be to make use the generalized union bound known as Occam's hammer (Blanchard & Fleuret, 2007).

3.2 Regularization with $F_Q(\cdot) = F_P(\cdot)$

Given the above argument it appears necessary to control function class capacity in this model in order to deliver low empirical Gibbs risk. We therefore consider the presence of regularization terms in (4), (5) which encode a preference for classifiers which satisfy some notion of simplicity. The flexibility of this model is such that, with reference to (6), when $F_Q(\cdot) = F_P(\cdot)$, the bounds of Theorem 3 hold for this case. We can therefore apply arbitrary (non data-dependent) regularization and attain the same bound of Theorem 3, and there are many natural possibilities. For example, if $\mathcal{H}$ is equipped with a norm $||\cdot||_{\mathcal{H}}$ we can choose $F_Q(\cdot) = F_P(\cdot) = ||\cdot||_{\mathcal{H}}$. This should permit learning with smaller γ.

3.3 Regularization in the Intrinsic Data Geometry

The flexibility of this model further permits, in a straightforward way, regularization w.r.t. the geometry defined by the *unknown* data-generating distribution, and we detail one way of achieving this. The regularization methods considered in Section 3.2 utilise a geometry which is extrinsic to the data, that is, determined by the ambient representation space rather than the intrinsic geometry of data. For example, if the data generating distribution has support on some submanifold of the ambient space, then encouraging smoothness on the manifold ought to be more suitable for learning (since if the structure of data is a key factor in the learnability of a task, it is the intrinsic geometry which will capture this relevant structure most accurately). In general, when using a regularizer informed by the intrinsic geometry of the data-generating distribution the prior and posterior regularizers must be different since the posterior regularizer will be an empirical quantity (here, chosen to be an estimate, based on the sample, of the prior regularizer).

Given a sample $\mathcal{S} = \{(X_1, Y_1), ...(X_m, Y_m)\} \cup \{X_{m+1}, ...X_n\}$, we consider regularizing via the following *smoothness functional* (typical in semi-supervised learning e.g. (Belkin et al., 2004; Zhu et al., 2003)) over functions from some function class $\mathcal{H}$:

$$\widehat{U}_{\mathcal{S}}(h) := \frac{1}{n(n-1)} \sum_{ij} (h(X_i) - h(X_j))^2 W(X_i, X_j) \tag{7}$$

where the symmetric $W : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ captures similarity or "weight" between data points, for example $W(\boldsymbol{x}, \boldsymbol{x}') = \begin{cases} 1 \text{ if } \boldsymbol{x}, \boldsymbol{x}' \text{ are a pair of } k\text{−nearest neighbours} \\ 0 \text{ otherwise} \end{cases}$ or $W(\boldsymbol{x}, \boldsymbol{x}') = e^{-||\boldsymbol{x}-\boldsymbol{x}'||^2}$ for some norm $||\cdot||$ over $\mathcal{X}$. Note that $\widehat{U}_{\mathcal{S}}(h) = \frac{2}{n(n-1)} \boldsymbol{h}^\top \boldsymbol{L} \boldsymbol{h}$ where $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ is the graph Laplacian of a graph $\mathcal{G}$ whose vertices are the sample instances and whose *edge weights* are controlled by W, and $D_{ij} = \delta_{ij} \sum_k W_{ik}$ and where $\boldsymbol{h} \in \mathbb{R}^n$ is the "point evaluation" of h on the sample, $h_i := h(\boldsymbol{x}_i)$. Minimizing (7) encourages functions to be smooth over the sample $\mathcal{S}$. Note that $\widehat{U}_{\mathcal{S}}(h)$ is a U-statistic of order 2 with *kernel* $f_h(X_i, X_j) := (h(X_i) - h(X_j))^2 W(X_i, X_j)$ indexed by $\mathcal{H}$. A family of U-statistics indexed by a function space is often called a U-process. We suppose that the weights are bounded, $|W(\boldsymbol{x}, \boldsymbol{x}')| \le w$, for example if $W(\boldsymbol{x}, \boldsymbol{x}') = e^{-||\boldsymbol{x}-\boldsymbol{x}'||^2}$ we have $w = 1$, and that $\sup_{h \in \mathcal{H}, \boldsymbol{x} \in \mathcal{X}} |h(\boldsymbol{x})| = b$.

A series of results (Hein et al., 2006) demonstrate that under certain conditions on the distribution of instances, certain constructions of graph Laplacian converge to a generalized Laplace operator on the support of the data generating distribution and the smoothness functional converges to a natural distribution-dependent measure of smoothness over functions defined over the data.

We choose $F_Q(\cdot) = \widehat{U}_{\mathcal{S}}(\cdot)$ so that,

$$q(h) = \frac{1}{Z} e^{-\left(\gamma \widehat{\mathrm{risk}}_{\mathcal{S}}(h) + \eta \widehat{U}_{\mathcal{S}}(h)\right)}. \tag{8}$$

The exponent simply minimizes empirical risk plus the smoothness on the graph formed on the sample, as is a typical methodology in semi-supervised learning (Belkin et al., 2006; Belkin et al., 2004).

We further choose $F_P(h) = U(h) := \mathbb{E}_{X,X'}[(h(X) - h(X'))^2 W(X,X')] = \mathbb{E}_{\mathcal{S}}[\widehat{U}_{\mathcal{S}}(h)]$, giving the prior $p(h) = \frac{1}{Z'} e^{-(\gamma \mathrm{risk}(h) + \eta U(h))}$.

Convergence of the smoothness functional. We consider PAC-Bayes convergence of the U-process (see (Ralaivola et al., 2009) for an alternative PAC-Bayes analysis of U-statistics). Let $\mathcal{S} = \{X_1, ... X_n\}$ be an i.i.d. sample. For any second-order U-statistic $\widehat{U}_{\mathcal{S}}(h) = \frac{1}{n(n-1)} \sum_{i \neq j} f_h(X_i, X_j)$ with expectation $U(h)$, and with kernel $f_h(x, x')$ indexed by $\mathcal{H}$ and bounded, $a \leq f_h(x, x') \leq b$, we have the following.

Theorem 4. *For all t, any prior P and simultaneously for all posteriors Q over $\mathcal{H}$,*

$$\mathbb{P}_{\mathcal{S}}\left(\mathbb{E}_{h\sim Q}[\widehat{U}_{\mathcal{S}}(h) - U(h)] \leq \frac{1}{t}\left(\mathrm{KL}(Q||P) + \frac{t^2(b-a)^2}{2n} + \ln\left(\frac{1}{\delta}\right)\right)\right) \geq 1 - \delta \tag{9}$$

$$\mathbb{P}_{\mathcal{S}}\left(\mathbb{E}_{h\sim Q}[U(h) - \widehat{U}_{\mathcal{S}}(h)] \leq \frac{1}{t}\left(\mathrm{KL}(Q||P) + \frac{t^2(b-a)^2}{2n} + \ln\left(\frac{1}{\delta}\right)\right)\right) \geq 1 - \delta. \tag{10}$$

In particular, choosing $t = \sqrt{n}$ gives $\mathcal{O}(\frac{1}{\sqrt{n}})$ convergence.

Proof. We note that Theorem 1 implies that with probability at least $1 - \delta$, $\forall Q$ on $\mathcal{H}$:

$$\mathbb{E}_{h\sim Q}[\widehat{U}_{\mathcal{S}}(h) - U(h)] \leq \frac{1}{t}\left(\mathrm{KL}(Q||P) + \ln\left(\frac{1}{\delta}\mathbb{E}_{h\sim P}\mathbb{E}_{\mathcal{S}}\left[e^{t(\widehat{U}_{\mathcal{S}}(h) - U(h))}\right]\right)\right),$$

so we simply need to bound $\mathbb{E}_{\mathcal{S}}\left[e^{t(\widehat{U}_{\mathcal{S}}(h) - U(h))}\right]$. Employing Hoeffding's canonical decomposition of U-statistics into forward martingales (e.g. (Serfling, 1980)), let,

$$V_k := \sum_{i=1}^{k} \left(\mathbb{E}[f_h(X_i, X) \mid X_i] - U(h)\right)$$

$$W_k := \sum_{j=1}^{k} \sum_{i=1}^{j-1} \left(f_h(X_i, X_j) + U(h) - \mathbb{E}[f_h(X_i, X) \mid X_i] - \mathbb{E}[f_h(X, X_j) \mid X_j]\right),$$

so that, $\widehat{U}_{\mathcal{S}}(h) - U(h) = \frac{2}{n}V_n + \frac{2}{n(n-1)}W_n$. We then have that

$$V_k - V_{k-1} = \mathbb{E}[f_h(X_k, X \mid X_k)] - U(h)$$
$$W_k - W_{k-1} = \sum_{i=1}^{k-1} f_h(X_i, X_k) + U(h) - \mathbb{E}[f_h(X_i, X) \mid X_i] - \mathbb{E}[f_h(X_k, X) \mid X_k],$$

and note the martingale structure $\mathbb{E}_{X_k}[V_k - V_{k-1}] = \mathbb{E}_{X_k}[W_k - W_{k-1}] = 0$. Note further that,

$$V_k - V_{k-1} + \frac{1}{n-1}(W_k - W_{k-1}) = \frac{n-k}{n-1}(\mathbb{E}[f_h(X_k, X) \mid X_k] - U(h)) + \frac{1}{n-1}\sum_{i=1}^{k-1} f_h(X_i, X_k) - \mathbb{E}[f_h(X_i, X) \mid X_i],$$

so that,

$$|V_k - V_{k-1} + \frac{1}{n-1}(W_k - W_{k-1})| \leq (b-a)\frac{n-k}{n-1} + (b-a)\frac{k-1}{n-1} = b-a. \quad (11)$$

Now,

$$\mathbb{E}_{\mathcal{S}}\left[e^{t(\widehat{U}_{\mathcal{S}}(h) - U(h))}\right] = \mathbb{E}_{\mathcal{S}}\left[e^{\frac{2t}{n}\sum_{i=1}^{n} V_i - V_{i-1} + \frac{1}{n-1}(W_i - W_{i-1})}\right]$$
$$= \mathbb{E}_{X_1,\ldots X_{n-1}}\left[\mathbb{E}_{X_n}\left[e^{\frac{2t}{n}\sum_{i=1}^{n} V_i - V_{i-1} + \frac{1}{n-1}(W_i - W_{i-1})} \mid X_1, \ldots X_{n-1}\right]\right]$$
$$= \mathbb{E}_{X_1,\ldots X_{n-1}}\left[e^{\frac{2t}{n}\sum_{i=1}^{n-1} V_i - V_{i-1} + \frac{1}{n-1}(W_i - W_{i-1})}\mathbb{E}_{X_n}\left[e^{\frac{2t}{n}\left(V_n - V_{n-1} + \frac{1}{n-1}(W_n - W_{n-1})\right)}\right]\right]$$
$$\leq \mathbb{E}_{X_1,\ldots X_{n-1}}\left[e^{\frac{2t}{n}\sum_{i=1}^{n-1} V_i - V_{i-1} + \frac{1}{n-1}(W_i - W_{i-1})}\right] e^{\frac{t^2(b-a)^2}{2n^2}}$$
$$\vdots$$
$$\leq \prod_{i=1}^{n} e^{\frac{t^2(b-a)^2}{2n^2}} = e^{\frac{t^2(b-a)^2}{2n}},$$

where in the final lines we used Hoeffding's lemma, Lemma 6, combined with (11) recursively. This proves (9), and (10) follows by a symmetrical argument. □

We can now give the following bound for the classification risk of the Gibbs classifier G_Q drawn from the distribution (8) over $\mathcal{H}$:

Theorem 5. *For $\eta < \sqrt{n}$,*

$$\mathbb{P}_{\mathcal{S}}\left(\mathrm{kl}(\widehat{\mathrm{risk}}_{\mathcal{S}}(G_Q), \mathrm{risk}(G_Q)) \leq \frac{1}{m}\left(A^2 + B + A\sqrt{2B + A^2} + \ln\frac{\xi(m)}{\delta}\right)\right) \geq 1 - \delta,$$

where

$$A := \frac{\gamma\sqrt{n}}{2\sqrt{m}(\sqrt{n} - \eta)} = \mathcal{O}\left(\frac{1}{\sqrt{m}}\right)$$
$$B := \frac{\sqrt{n}}{\sqrt{n} - \eta}\left(\gamma\sqrt{\frac{2}{m}\ln\frac{4\xi(m)}{\delta}} + \frac{2\eta}{\sqrt{n}}\left(32b^4w^2 + \ln\frac{4}{\delta}\right)\right) = \mathcal{O}\left(\sqrt{\frac{\ln m}{m}}\right).$$

Proof. From (6) we have

$$\mathrm{KL}(Q||P) \leq \gamma(\mathrm{risk}(G_Q) - \widehat{\mathrm{risk}}_{\mathcal{S}}(G_Q)) + \gamma(\widehat{\mathrm{risk}}_{\mathcal{S}}(G_P) - \mathrm{risk}(G_P)) + \eta \mathbb{E}_{h\sim Q}\left[U(h) - \widehat{U}_{\mathcal{S}}(h)\right] + \eta \mathbb{E}_{h\sim P}\left[\widehat{U}_{\mathcal{S}}(h) - U(h)\right]. \quad (12)$$

And now, recalling (1), and noting that, because $|h(\boldsymbol{x})| \leq b$, $W(\boldsymbol{x}, \boldsymbol{x}') \leq w$, the kernel satisfies $|f_h(\boldsymbol{x}, \boldsymbol{x}')| \leq 4b^2 w$, as in Theorem 3 we apply Seeger's bound of Theorem 2 and Theorem 4 to the relevant terms in (12), and apply the union bound, so that with probability at least $1 - \delta$ over the draw of $\mathcal{S}$,

$$\begin{aligned}
\mathrm{KL}(Q||P) &\leq \gamma\sqrt{\frac{1}{2m}\left(\mathrm{KL}(Q||P) + \ln\frac{4\xi(m)}{\delta}\right)} + \gamma\sqrt{\frac{1}{2m}\ln\frac{4\xi(m)}{\delta}} \\
&\quad + \frac{\eta}{\sqrt{n}}\left(\mathrm{KL}(Q||P) + 32b^4w^2 + \ln\frac{4}{\delta}\right) + \frac{\eta}{\sqrt{n}}\left(32b^4w^2 + \ln\frac{4}{\delta}\right) \\
&\leq \gamma\sqrt{\frac{1}{2m}\mathrm{KL}(Q||P)} + \frac{\eta}{\sqrt{n}}\mathrm{KL}(Q||P) + \gamma\sqrt{\frac{2}{m}\ln\frac{4\xi(m)}{\delta}} + \frac{2\eta}{\sqrt{n}}\left(32b^4w^2 + \ln\frac{4}{\delta}\right) \\
\left(\sqrt{\mathrm{KL}(Q||P)} - \frac{1}{\sqrt{2}}A\right)^2 &\leq B + \frac{A^2}{2} \\
\mathrm{KL}(Q||P) &\leq A^2 + B + A\sqrt{2B + A^2},
\end{aligned}$$

which we plug into the bound of Theorem 2. □

We remark that the ease with which we can obtain this bound for regularization w.r.t. the geometry defined by the unknown data-generating distribution, with apparently very little deterioration in the bound, is unusual and that in classical frameworks this type of structuring of a function class usually results in significant deterioration in the bound.

4 Prediction by RKHS Regularization

We now extend the localization framework to the more practical setting of predicting with a Gaussian process whose mean is the solution to a loss minimization with RKHS regularization, such as an SVM solution. We consider a separable[2] RKHS $\mathcal{H} = \overline{\mathrm{span}\{K(\boldsymbol{x}, \cdot) : \boldsymbol{x} \in \mathcal{X}\}}$, for some kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, of functions which are identified as binary classifiers via $h_{\mathrm{class}}(\boldsymbol{x}) = \mathrm{sgn}(h(\boldsymbol{x})) = \mathrm{sgn}(\langle h, K(\boldsymbol{x}, \cdot)\rangle_{\mathcal{H}})$. For any chosen loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, we are interested in the classifiers,

$$h^*_{\mathcal{S}} := \underset{h\in\mathcal{H}}{\mathrm{argmin}}\{\widehat{\mathrm{risk}}^{\ell}_{\mathcal{S}}(h) + \eta||h||^2_{\mathcal{H}}\} \quad \text{and} \quad h^* := \mathbb{E}_{\mathcal{S}}[h^*_{\mathcal{S}}],$$

where η is a regularization parameter and expectation is taken with respect to samples $\mathcal{S}$ with m labelled points. For our intended applications, typically $\widehat{\mathrm{risk}}^{\ell}_{\mathcal{S}}(\cdot)$ will be convex so that $h^*_{\mathcal{S}}$ is unique and h^* well-defined. Our posterior Q and prior P will be Gaussian processes on $\mathcal{X}$ with mean $h^*_{\mathcal{S}}$ and h^* respectively. To define this, denote

[2] This is a mild condition, an RKHS $\mathcal{H}$ is separable if $\mathcal{X}$ is and if the kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is continuous.

by $\mathcal{L}^2(\mathcal{X})$ the Hilbert space of square integrable real-valued functions on $\mathcal{X}$ with inner product $\langle h, g\rangle_{\mathcal{L}^2} = \int_{\mathcal{X}} h(\boldsymbol{x})g(\boldsymbol{x})\mathrm{d}\boldsymbol{x}$, and consider the countable orthonormal basis $\{\phi_i\}$ for $\mathcal{L}^2(\mathcal{X})$ provided by the eigenfunctions of the integral operator A_K defined by $(A_K h)(\boldsymbol{x}) := \int K(\boldsymbol{x}, \boldsymbol{x}')h(\boldsymbol{x}')\mathrm{d}\boldsymbol{x}'$, i.e. such that $A_K(\phi_i) = \lambda_i\phi_i$, for eigenvalues $\{\lambda_i\}$ and $\langle \phi_i, \phi_j\rangle_{\mathcal{L}^2} = \delta_{ij}$. Denote $h_i := \langle h, \phi_i\rangle_{\mathcal{L}^2}$ and consider the isomorphism $I : \mathcal{L}^2(\mathcal{X}) \to \ell^2$ given by $I(h) = (h_i)$ identifying $\mathcal{L}^2(\mathcal{X})$ with the space of square summable real-valued sequences. Denote by $N_{h_i, \frac{1}{\gamma}\lambda_i}$ the one-dimensional Gaussian measure on (the Borel σ-algebra on) $\mathbb{R}$ with mean h_i and variance $\frac{1}{\gamma}\lambda_i$. We then define the product measures[3],

$$Q := \prod_{i=1}^{\infty} N_{(h^*_{\mathcal{S}})_i, \frac{1}{\gamma}\lambda_i} \quad \text{and } P := \prod_{i=1}^{\infty} N_{h^*_i, \frac{1}{\gamma}\lambda_i}. \tag{13}$$

That Q and P define probability measures on $\mathcal{L}^2(\mathcal{X})$ is the subject of (Da Prato, 2006, Chapter 1). A full treatment of this subject requires more space and will be presented in a longer version of this paper (or see (Lever et al., 2010)). To build intuition, when the distributions are of finite dimensionality they have density (w.r.t. Lebesgue measure under the above isomorphism),

$$q(h) = \frac{1}{Z} e^{-\frac{\gamma}{2}||h - h^*_{\mathcal{S}}||^2_{\mathcal{H}}} \text{ and } \qquad p(h) = \frac{1}{Z'} e^{-\frac{\gamma}{2}||h - h^*||^2_{\mathcal{H}}} \tag{14}$$

where, Z, Z' enforce normalization. When the dimension of $\mathcal{H}$ is infinite any marginalization to a finite-dimensional linear subspace of $\mathcal{L}^2(\mathcal{X})$ has a similar density. Note that γ is a parameter of the algorithm which controls the variance of the Gaussian distribution.

We note, when predicting on a finite set of points, the equivalence between the Gibbs classifier drawn from the posterior (13) and a Gaussian process $\{G_{\boldsymbol{x}}\}_{\boldsymbol{x}\in\mathcal{X}}$ on $\mathcal{X}$ with mean $\mathbb{E}[G_{\boldsymbol{x}}] = h^*_{\mathcal{S}}(\boldsymbol{x})$ and covariance $\mathbb{E}[(G_{\boldsymbol{x}} - \mathbb{E}[G_{\boldsymbol{x}}])(G_{\boldsymbol{x}'} - \mathbb{E}[G_{\boldsymbol{x}'}])] = \frac{1}{\gamma}K(\boldsymbol{x}, \boldsymbol{x}')$.

To obtain a PAC-Bayes bound for the Gibbs classifier drawn from Q, we proceed to establish a bound for the relative entropy between Q and P. For any Mercer kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, we denote

$$\kappa(\boldsymbol{x}) := \sup_{h\in\mathcal{H}} \frac{|h(\boldsymbol{x})|}{||h||_{\mathcal{H}}} = \sqrt{K(\boldsymbol{x}, \boldsymbol{x})} \quad \text{and} \quad \kappa := \sup_{\boldsymbol{x}\in\mathcal{X}} \kappa(\boldsymbol{x}),$$

and define the distance $d_K(\boldsymbol{x}, \boldsymbol{x}') := ||K(\boldsymbol{x}, \cdot) - K(\boldsymbol{x}', \cdot)||_{\mathcal{H}}$. Note that $d_K(\boldsymbol{x}, \boldsymbol{x}') \leq 2\kappa$. Our analyses will make use of the following property of a loss function:

Definition 1. *(Bousquet & Elisseeff, 2002, Definition 19)* $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ *is* α*-admissible with respect to* $\mathcal{H}$ *if it is convex in its first argument and for all* $y \in \mathcal{Y}$,

$$|\ell(y_1, y) - \ell(y_2, y)| \leq \alpha|y_1 - y_2|,$$

for all y_1, y_2 *in the domain of the functions from* $\mathcal{H}$.

[3] These measures are defined on $\mathbb{R}^\infty$ but their support is precisely ℓ^2, i.e. $\mathcal{L}^2(\mathcal{X})$.

We recall the following definition of Bregman divergence on a Hilbert space $\mathcal{V}$: for differentiable convex $\Phi : \mathcal{V} \to \mathbb{R}$,

$$d_\Phi(u, v) := \Phi(u) - \Phi(v) - \langle \nabla \Phi(v), u - v \rangle_{\mathcal{V}}. \tag{15}$$

Lemma 3. $\mathrm{KL}(Q||P) = \frac{\gamma}{2}||h^*_{\mathcal{S}} - h^*||^2_{\mathcal{H}}$.

Proof. When Q and P are finite-dimensional distributions this is the well-known formula for the KL divergence between Gaussians. When the dimensionality is infinite some subtleties are required and due to lack of space we refer the reader to a longer version of this paper (Lever et al., 2010). □

We now proceed to upper bound this divergence via a method of bounded differences: consider a sample $\mathcal{S}$ and its "perturbation" $\mathcal{S}^{(i)}$,

$$\mathcal{S} := \{(X_1, Y_1), ...(X_m, Y_m)\}$$
$$\mathcal{S}^{(i)} := \{(X_1, Y_1), ...(X_{i-1}, Y_{i-1}), (X'_i, Y'_i), (X_{i+1}, Y_{i+1}), ...(X_m, Y_m)\}.$$

Lemma 4. *If $\ell(\cdot, \cdot)$ is α-admissible and differentiable*[4] *then*

$$||h^*_{\mathcal{S}^{(i)}} - h^*_{\mathcal{S}}||_{\mathcal{H}} \le \frac{\alpha}{2\eta m}(\kappa(X_i) + \kappa(X'_i)). \tag{16}$$

Proof. The method of proof is a stability argument which follows (Bousquet & Elisseeff, 2002, Theorem 22). Denote the "objectives"

$$\Omega(h) := \widehat{\mathrm{risk}}^\ell_{\mathcal{S}}(h) + \eta||h||^2_{\mathcal{H}},$$
$$\Omega^{(i)}(h) := \widehat{\mathrm{risk}}^\ell_{\mathcal{S}^{(i)}}(h) + \eta||h||^2_{\mathcal{H}}.$$

Since $\nabla\Omega(h^*_{\mathcal{S}}) = \nabla\Omega^{(i)}(h^*_{\mathcal{S}^{(i)}}) = 0$, we have,

$$\begin{aligned} d_\Omega(h^*_{\mathcal{S}^{(i)}}, h^*_{\mathcal{S}}) + d_{\Omega^{(i)}}(h^*_{\mathcal{S}}, h^*_{\mathcal{S}^{(i)}}) &= \Omega(h^*_{\mathcal{S}^{(i)}}) - \Omega(h^*_{\mathcal{S}}) + \Omega^{(i)}(h^*_{\mathcal{S}}) - \Omega^{(i)}(h^*_{\mathcal{S}^{(i)}}) \\ &= \frac{1}{m}(\ell(h^*_{\mathcal{S}^{(i)}}(X_i), Y_i) - \ell(h^*_{\mathcal{S}^{(i)}}(X'_i), Y'_i) \\ &\quad + \ell(h^*_{\mathcal{S}}(X'_i), Y'_i) - \ell(h^*_{\mathcal{S}}(X_i), Y_i)). \end{aligned}$$

Noting the additivity, $d_{\Phi+\Psi} = d_\Phi + d_\Psi$, and non-negativity of Bregman divergences and that $d_{\eta||\cdot||^2_{\mathcal{H}}}(h, g) = \eta||h - g||^2_{\mathcal{H}}$ we have,

$$\begin{aligned} &2\eta||h^*_{\mathcal{S}} - h^*_{\mathcal{S}^{(i)}}||^2_{\mathcal{H}} \\ &\quad\le \frac{1}{m}\big(\ell(h^*_{\mathcal{S}^{(i)}}(X_i), Y_i) - \ell(h^*_{\mathcal{S}^{(i)}}(X'_i), Y'_i) + \ell(h^*_{\mathcal{S}}(X'_i), Y'_i) - \ell(h^*_{\mathcal{S}}(X_i), Y_i)\big) \\ &\quad\le \frac{\alpha}{m}(|h^*_{\mathcal{S}}(X_i) - h^*_{\mathcal{S}^{(i)}}(X_i)| + |h^*_{\mathcal{S}}(X'_i) - h^*_{\mathcal{S}^{(i)}}(X'_i)|) \\ &\quad\le \frac{\alpha}{m}(||h^*_{\mathcal{S}} - h^*_{\mathcal{S}^{(i)}}||_{\mathcal{H}}(\kappa(X_i) + \kappa(X'_i))). \end{aligned}$$ □

[4] We note that for the case of the hinge loss or absolute loss this condition can be relaxed – we can define the derivative to be zero at the point at which they are non-differentiable. For general subdifferentiable convex loss functions we recover the results if we define the gradient to be zero at the minimum.

Lemma 5. *If $\ell(\cdot,\cdot)$ is α-admissible, differentiable*[4] *and $\mathcal{H}$ is separable then*

$$\mathbb{P}_{\mathcal{S}}\left(||h^*_{\mathcal{S}} - h^*||_{\mathcal{H}} \leq \frac{2\alpha\kappa}{\eta}\sqrt{\frac{1}{m}\ln\frac{4}{\delta}}\right) \geq 1-\delta. \tag{17}$$

Proof. Define the Doob martingale,

$$V_i = \mathbb{E}[h^*_{\mathcal{S}} - h^* \mid (X_1,Y_1),...(X_i,Y_i)],$$

and note that $V_0 = 0$, $V_m = h^*_{\mathcal{S}} - h^*$, and that

$$\begin{aligned}\mathbb{E}[V_i \mid (X_1,Y_1),...(X_{i-1},Y_{i-1})] &= \mathbb{E}[h^*_{\mathcal{S}} - h^* \mid (X_1,Y_1),...(X_{i-1},Y_{i-1})] \\ &= V_{i-1}.\end{aligned}$$

Thus $\{V_i\}_{i=1}^m$ is a martingale and we have further, by Lemma 4, that

$$\begin{aligned}||V_i - V_{i-1}||_{\mathcal{H}} &= ||\mathbb{E}[h^*_{\mathcal{S}} \mid (X_1,Y_1),...(X_i,Y_i)] - \mathbb{E}[h^*_{\mathcal{S}} \mid (X_1,Y_1),...(X_{i-1},Y_{i-1})]||_{\mathcal{H}} \\ &\leq \frac{\kappa\alpha}{\eta m}.\end{aligned}$$

Since $\mathcal{H}$ is separable it has a countable basis and so is isomorphic to either $\ell^2(\mathbb{R})$ or $\mathbb{R}^d$ and the result follows from the result of (Kallenberg & Sztencel, 1991, Theorem 3.1) (which gives a version of Azuma's inequality for ℓ^2-valued martingales, see the details in Theorem 7 and Corollary 1 of the Appendix). □

We can now give the PAC-Bayes bound for the classification risk of the Gibbs classifier, G_Q, drawn from $\mathcal{H}$ according to the distribution Q defined by (13).

Theorem 6. *If $\ell(\cdot,\cdot)$ is α-admissible, differentiable*[4] *and $\mathcal{H}$ is separable then*

$$\mathbb{P}_{\mathcal{S}}\left(\mathrm{kl}(\widehat{\mathrm{risk}}_{\mathcal{S}}(G_Q), \mathrm{risk}(G_Q)) \leq \frac{1}{m}\left(\frac{2\gamma\alpha^2\kappa^2}{\eta^2 m}\ln\frac{8}{\delta} + \ln\frac{2\xi(m)}{\delta}\right)\right) \geq 1-\delta.$$

Proof. Lemma 3 and Lemma 5 immediately imply that,

$$\mathbb{P}_{\mathcal{S}}\left(KL(Q||P) \leq \frac{2\gamma\alpha^2\kappa^2}{\eta^2 m}\ln\frac{8}{\delta}\right) \geq 1 - \frac{\delta}{2},$$

which we combine with Theorem 2 using the union bound. □

Note that the PAC-Bayes bounds for Gibbs classifiers presented here will provide sharp bounds on the mean classifier (which, with suitable choices for parameters, could be various types of SVM), with an additional factor of $1+\epsilon$, under a margin assumption, by standard techniques (Langford & Shawe-taylor, 2002). We also remark that it is straightforward to extend the analysis presented here to provide bounds for classifiers obtained by regularizing in the geometry defined by the data, in the manner of Section 3.3, such as LapSVM, and refer the reader to an extended version of this paper (Lever et al., 2010).

References

Ambroladze, A., Parrado-Hernández, E., Shawe-Taylor, J.: Tighter pac-bayes bounds. In: NIPS, pp. 9–16. MIT Press, Cambridge (2006)

Azuma, K.: Weighted sums of certain dependent random variables. Tohoku Mathematical Journal 68, 357–367 (1967)

Balcan, M., Blum, A.: A discriminative model for semi-supervised learning. JACM, 57 (2010)

Belkin, M., Matveeva, I., Niyogi, P.: Regularization and semi-supervised learning on large graphs. In: Shawe-Taylor, J., Singer, Y. (eds.) COLT 2004. LNCS (LNAI), vol. 3120, pp. 624–638. Springer, Heidelberg (2004)

Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. Journal of Machine Learning Research 7, 2399–2434 (2006)

Blanchard, G., Fleuret, F.: Occam's hammer. In: Bshouty, N.H., Gentile, C. (eds.) COLT. LNCS (LNAI), vol. 4539, pp. 112–126. Springer, Heidelberg (2007)

Bousquet, O., Elisseeff, A.: Stability and generalization. J. Mach. Learn. Res. 2, 499–526 (2002)

Catoni, O.: PAC-Bayesian surpevised classification: the thermodynamics of statistical learning. Monograph Series of the Institute of Mathematical Statistics (2007)

Da Prato, G.: An introduction to infinite-dimensional analysis. Springer, Heidelberg (2006)

Germain, P., Lacasse, A., Laviolette, F., Marchand, M.: Pac-bayesian learning of linear classifiers. In: ICML, p. 45. ACM, New York (2009)

Hein, M., Audibert, J.-Y., von Luxburg, U.: Graph laplacians and their convergence on random neighborhood graphs. CoRR (2006)

Kallenberg, O., Sztencel, R.: Some dimension-free features of vector-valued martingales. Probability Theory and Related Fields 88, 215–247 (1991)

Langford, J.: Tutorial on practical prediction theory for classification. Journal of Machine Learning Research 6, 273–306 (2005)

Langford, J., Shawe-taylor, J.: Pac-bayes and margins. In: Advances in Neural Information Processing Systems, vol. 15, pp. 439–446. MIT Press, Cambridge (2002)

Lever, G., Laviolette, F., Shawe-Taylor, J.: Distribution dependent pac-bayes priors. UCL technical report (2010), http://www.cs.ucl.ac.uk/staff/G.Lever/pubs/DDPB.pdf

McAllester, D.A.: Pac-bayesian model averaging. In: COLT, pp. 164–170 (1999)

Ralaivola, L., Szafranski, M., Stempfel, G.: Chromatic pac-bayes bounds for non-iid data: Applications to ranking and stationary β-mixing processes. CoRR, abs/0909.1993 (2009)

Seeger, M.: Pac-bayesian generalisation error bounds for gaussian process classification. Journal of Machine Learning Research 3, 233–269 (2002)

Serfling, R.: Approximation theorems of mathematical statistics. Wiley, Chichester (1980)

Shawe-Taylor, J., Bartlett, P.L., Williamson, R.C., Anthony, M.: Structural risk minimization over data-dependent hierarchies. IEEE Transactions on Information Theory 44, 1926–1940 (1998)

Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using gaussian fields and harmonic functions. In: ICML 2003, pp. 912–919 (2003)

A Technical Lemmas

Lemma 6. *(Hoeffding's lemma) Let X be a random variable with $\mathbb{E}[X] = 0$ and $a < X < b$ then for $t > 0$,*

$$\mathbb{E}[e^{tX}] \leq e^{\frac{t^2(b-a)^2}{8}}.$$

The following theorem demonstrates that many key properties of martingales are independent of their dimension. The authors note that it is true for any Hilbert space-valued martingale but the proof is just for martingales in ℓ^2.

Theorem 7. *(Kallenberg & Sztencel, 1991, Theorem 3.1) Let $\{V_t\}$ be a martingale in $\mathbb{R}^d$ or ℓ^2. Then there exists a martingale $\{U_t\}$ in $\mathbb{R}^2$ such that $||V_t|| = ||U_t||$ a.s. and $||V_t - V_{t-1}|| = ||U_t - U_{t-1}||$ a.s..*

Given the above result all that we must do to obtain a large deviation inequality for ℓ^2-valued martingales is to demonstrate a variation of Azuma-Hoeffding inequality for a martingale in $\mathbb{R}^2$, which is elementary if we are not concerned with obtaining the best constants.

Corollary 1. *For a martingale $\{V_i\}_{i=1}^m$ in $\mathbb{R}^d$ or ℓ^2, such that, for all i,*

$$||V_i - V_{i-1}|| \leq c_i,$$

we have for all $\delta > 0$,

$$\mathbb{P}\left(||V_m - V_0|| \leq 2\sqrt{\sum_{i=1}^m c_i^2 \ln\frac{4}{\delta}}\right) \geq 1 - \delta.$$

Proof. Consider a martingale $\{U_i\}_{i=1}^m$ in $\mathbb{R}^2$ such that,

$$||U_i - U_{i-1}|| \leq c_i. \tag{18}$$

Let $U_i = (U_i^{(1)}, U_i^{(2)})$, so that we have that $\{U_i^{(1)}\}_{i=1}^n$ and $\{U_i^{(2)}\}_{i=1}^n$ are clearly martingales and that,

$$|U_i^{(1)} - U_{i-1}^{(1)}| \leq c_i$$
$$|U_i^{(2)} - U_{i-1}^{(2)}| \leq c_i.$$

Now,

$$\begin{aligned}\mathbb{P}\left(||U_m - U_0|| \geq \epsilon\right) &= \mathbb{P}\left((U_m^{(1)} - U_0^{(1)})^2 + (U_m^{(2)} - U_0^{(2)})^2 \geq \epsilon^2\right) \\ &\leq \mathbb{P}\left(|U_m^{(1)} - U_0^{(1)}| \geq \frac{\epsilon}{\sqrt{2}}\right) + \mathbb{P}\left(|U_m^{(2)} - U_0^{(2)}| \geq \frac{\epsilon}{\sqrt{2}}\right) \\ &\leq 4\exp\left(-\frac{\epsilon^2}{4\sum_{i=1}^m c_i^2}\right),\end{aligned}$$

where the last line follows by the Azuma-Hoeffding inequality (Azuma, 1967). The result then follows by Theorem 7. □

PAC Learnability of a Concept Class under Non-atomic Measures: A Problem by Vidyasagar

Vladimir Pestov

Department of Mathematics and Statistics, University of Ottawa, 585 King Edward Avenue, Ottawa, Ontario, Canada K1N 6N5

Abstract. In response to a 1997 problem of M. Vidyasagar, we state a necessary and sufficient condition for distribution-free PAC learnability of a concept class $\mathscr{C}$ under the family of all non-atomic (diffuse) measures on the domain Ω. Clearly, finiteness of the classical Vapnik–Chervonenkis dimension of $\mathscr{C}$ is a sufficient, but no longer necessary, condition. Besides, learnability of $\mathscr{C}$ under non-atomic measures does not imply the uniform Glivenko–Cantelli property with regard to non-atomic measures. Our learnability criterion is stated in terms of a combinatorial parameter $\mathrm{VC}(\mathscr{C} \bmod \omega_1)$ which we call the VC dimension of $\mathscr{C}$ modulo countable sets. The new parameter is obtained by "thickening up" single points in the definition of VC dimension to uncountable "clusters". Equivalently, $\mathrm{VC}(\mathscr{C} \bmod \omega_1) \leq d$ if and only if every countable subclass of $\mathscr{C}$ has VC dimension $\leq d$ outside a countable subset of Ω. The new parameter can be also expressed as the classical VC dimension of $\mathscr{C}$ calculated on a suitable subset of a compactification of Ω. We do not make any measurability assumptions on $\mathscr{C}$, assuming instead the validity of Martin's Axiom (MA).

1 Introduction

A fundamental result of statistical learning theory says that for a concept class $\mathscr{C}$ the three conditions are equivalent: (1) $\mathscr{C}$ is distribution-free PAC learnable over the family $P(\Omega)$ of all probability measures on the domain Ω, (2) $\mathscr{C}$ is a uniform Glivenko–Cantelli class with regard to $P(\Omega)$, and (3) the Vapnik–Chervonenkis dimension of $\mathscr{C}$ is finite [VC, BEHW]. In this paper we are interested in the problem, discussed by Vidyasagar in both editions of his book [V1, V2] as problem 12.8, of giving a similar combinatorial description of concept classes $\mathscr{C}$ which are PAC learnable under the family $P_{na}(\Omega)$ of all non-atomic probability measures on Ω. (A measure μ is *non-atomic*, or *diffuse,* if every set A of strictly positive measure contains a subset B with $0 < \mu(B) < \mu(A)$.)

The condition $\mathrm{VC}(\mathscr{C}) < \infty$, while of course sufficient for $\mathscr{C}$ to be learnable under $P_{na}(\Omega)$, is not necessary. Let a concept class $\mathscr{C}$ consist of all finite and all cofinite subsets of a standard Borel space Ω. Then $\mathrm{VC}(\mathscr{C}) = \infty$, and moreover $\mathscr{C}$ is clearly not a uniform Glivenko-Cantelli class *with regard to non-atomic measures.* At the same time, $\mathscr{C}$ is PAC learnable under non-atomic measures: any learning rule $\mathcal{L}$ consistent with the subclass $\{\emptyset, \Omega\}$ will learn $\mathscr{C}$. Notice that $\mathscr{C}$ is not *consistently* learnable under non-atomic measures: there are consistent learning rules mapping every training sample to a finite set, and they will not learn any cofinite subset of Ω.

© Springer-Verlag Berlin Heidelberg 2010

The point of this example is that PAC learnability of a concept class $\mathscr{C}$ under non-atomic measures is not affected by adding to $\mathscr{C}$ symmetric differences $C \triangle N$ for each $C \in \mathscr{C}$ and every countable set N.

A version of VC dimension oblivious to this kind of set-theoretic "noise" is obtained from the classical definition by "thickening up" individual points and replacing them with uncountable clusters (Figure 1).

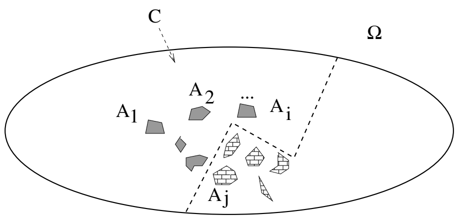


Fig. 1. A family $A_1, A_2, \ldots, A_n$ of uncountable sets shattered by $\mathscr{C}$

Define the *VC dimension of a concept class $\mathscr{C}$ modulo countable sets* as the supremum of natural n for which there exists a family of n uncountable sets, $A_1, A_2, \ldots, A_n \subseteq \Omega$, shattered by $\mathscr{C}$ in the sense that for each $J \subseteq \{1, 2, \ldots, n\}$, there is $C \in \mathscr{C}$ which contains all sets A_i, $i \in J$, and is disjoint from all sets A_j, $j \notin J$. Denote this parameter by $\mathrm{VC}(\mathscr{C} \bmod \omega_1)$. Clearly, for every concept class $\mathscr{C}$

$$\mathrm{VC}(\mathscr{C} \bmod \omega_1) \leq \mathrm{VC}(\mathscr{C}).$$

In our example above, one has $\mathrm{VC}(\mathscr{C} \bmod \omega_1) = 1$, even as $\mathrm{VC}(\mathscr{C}) = \infty$.

Here is our main result.

Theorem 1. *Let $(\Omega, \mathscr{A})$ be a standard Borel space, and let $\mathscr{C} \subseteq \mathscr{A}$ be a concept class. Under the Martin's Axiom (MA), the following are equivalent.*

1. *$\mathscr{C}$ is PAC learnable under the family of all non-atomic measures.*
2. *$\mathrm{VC}(\mathscr{C} \bmod \omega_1) = d < \infty$.*
3. *Every countable subclass $\mathscr{C}' \subseteq \mathscr{C}$ has finite VC dimension on the complement to some countable subset of Ω (which depends on $\mathscr{C}'$).*
4. *There is d such that for every countable $\mathscr{C}' \subseteq \mathscr{C}$ one has $\mathrm{VC}(\mathscr{C}') \leq d$ on the complement to some countable subset of Ω (depending on $\mathscr{C}'$).*
5. *Every countable subclass $\mathscr{C}' \subseteq \mathscr{C}$ is a uniform Glivenko–Cantelli class with regard to the family of non-atomic measures.*
6. *Same, with sample complexity $s(\epsilon, \delta)$ which only depends on $\mathscr{C}$ and not on $\mathscr{C}'$.*

If $\mathscr{C}$ is universally separable [P], the above are also equivalent to:

7. *VC dimension of $\mathscr{C}$ is finite outside of a countable subset of Ω.*
8. *$\mathscr{C}$ is a uniform Glivenko-Cantelli class with respect to the family of non-atomic probability measures.*

Martin's Axiom (MA) [F] is one of the most often used and best studied additional set-theoretic assumptions beyond the standard Zermelo-Frenkel set theory with the Axiom of Choice (ZFC). In particular, Martin's Axiom follows from the Continuum Hypothesis (CH), but it is also compatible with the negation of CH, and in fact it is namely the combination MA+¬CH that is really interesting.

The concept class in our initial simple example (which is even image admissible Souslin [D]) shows that in general (7) and (8) are not equivalent to the remaining conditions. Notice that for universally separable classes, (1), (7) and (8) are equivalent without additional set-theoretic assumptions.

The core of the theorem — and the main technical novelty of our paper — is the proof of the implication (3)⇒(1). It is based on a special choice of a consistent learning rule $\mathcal{L}$ having the property that for every concept $C \in \mathscr{C}$, the image of all learning samples of the form $(\sigma, C \cap \sigma)$ under $\mathcal{L}$ forms a uniform Glivenko–Cantelli class. It is for establishing this property of $\mathcal{L}$ that we need Martin's Axiom.

Most of the remaining implications are relavely straightforward adaptations of the standard techniques of statistical learning. Nevertheless, (2)⇒(3) requires a certain technical dexterity, and we study this implication in the setting of Boolean algebras.

We begin the paper by reviewing a general formal setting, followed by a dicussion of Boolean algebras which seem like a natural framework for the problem at hand, especially in view of possible generalizations to learning under other intermediate families of measures.

In particular, we will show that our version of the VC dimension modulo countable sets, $\mathrm{VC}(\mathscr{C} \bmod \omega_1)$, is just the usual VC dimension of the class $\mathscr{C}$ of concepts extended over a suitable compactification of Ω and restricted to a certain subdomain of the compactification.

Now the part of Theorem 1 for universally separable concept classes follows easily. Afterwards, we discuss Martin's Axiom, prove the existence of a learning rule with the above special property, and deduce Theorem 1 for arbitrary concept classes.

2 The Setting

We need to fix a precise setting, which is mostly standard. The *domain* (*instance space*) $\Omega = (\Omega, \mathscr{A})$ is a *measurable space,* that is, a set Ω equipped with a sigma-algebra of subsets $\mathscr{A}$. Typically, Ω is assumed to be a *standard Borel space,* that is, a complete separable metric space equipped with the sigma-algebra of Borel subsets. We will clarify the assumption whenever necessary.

A *concept class* is a family, $\mathscr{C}$, of measurable subsets of Ω. (Equivalently, $\mathscr{C}$ can be viewed as a family of measurable $\{0, 1\}$-valued functions on Ω.).

In the learning model, a set $\mathcal{P}$ of probability measures on Ω is fixed. Usually either $\mathcal{P} = P(\Omega)$ is the set of all probability measures (distribution-free learning), or $\mathcal{P} = \{\mu\}$ is a single measure (learning under fixed distribution). In our article, the case of interest is the family $\mathcal{P} = P_{na}(\Omega)$ of all non-atomic measures.

Every probability measure μ on Ω defines a distance d_μ on $\mathscr{A}$ as follows:

$$d_\mu(A, B) = \mu(A \vartriangle B)\,.$$

We will not distinguish between a measure μ and its Lebesgue completion, that is, an extension of μ over the larger sigma-algebra of Lebesgue measurable subsets of Ω. Consequently, we will sometimes use the term *measurability* meaning *Lebesgue measurability*. No confusion can arise here.

Often it is convenient to approximate the concepts from $\mathscr{C}$ with elements of the *hypothesis space, $\mathscr{H}$*, which is, technically, a subfamily of $\mathscr{A}$ whose closure with regard to each (pseudo)metric d_μ, $\mu \in \mathcal{P}$, contains $\mathscr{C}$. However, in our article we make no distinction between $\mathscr{H}$ and $\mathscr{C}$.

A *learning sample* is a pair $s = (\sigma, \tau)$ of finite subsets of Ω, where $\tau \subseteq \sigma$. It is convenient to assume that elements $x_1, x_2, \ldots, x_n \in \sigma$ are ordered, and thus the set of all samples (σ, τ) with $|\sigma| = n$ can be identified with $(\Omega \times \{0,1\})^n$. A *learning rule* (for $\mathscr{C}$) is a mapping

$$\mathcal{L}\colon \bigcup_{n=1}^{\infty} \Omega^n \times \{0,1\}^n \to \mathscr{C}$$

which satisfies the following measurability condition: for every $C \in \mathscr{C}$ and $\mu \in \mathcal{P}$, the function

$$\Omega \ni \sigma \mapsto \mu\left(\mathcal{L}(\sigma, C \cap \sigma) \bigtriangleup C\right) \in \mathbb{R} \tag{1}$$

is measurable.

A learning rule $\mathcal{L}$ is *consistent* (with $\mathscr{C}$) if for every $C \in \mathscr{C}$ and each $\sigma \in \Omega^n$ one has

$$\mathcal{L}(\sigma, C \cap \sigma) \cap \sigma = C \cap \sigma.$$

A learning rule $\mathcal{L}$ is *probably approximately correct* (*PAC*) *under* $\mathcal{P}$ if for every $\epsilon > 0$

$$\sup_{\mu \in \mathcal{P}} \sup_{C \in \mathscr{C}} \mu^{\otimes n} \{\sigma \in \Omega^n \colon \mu\left(\mathcal{L}(\sigma, C \cap \sigma) \bigtriangleup C\right) > \epsilon\} \to 0 \text{ as } n \to \infty. \tag{2}$$

Here $\mu^{\otimes n}$ denotes the (Lebesgue extension of the) product measure on Ω^n. Now the origin of the measurability condition (1) on the mapping $\mathcal{L}$ is clear: it is implicit in (2).

Equivalently, there is a function $s(\epsilon, \delta)$ (*sample complexity* of $\mathcal{L}$) such that for each $C \in \mathscr{C}$ and every $\mu \in \mathcal{P}$ an i.i.d. sample σ with $\geq s(\epsilon, \delta)$ points has the property $\mu(C \bigtriangleup \mathcal{L}(\sigma, C \cap \sigma)) < \epsilon$ with confidence $\geq 1 - \delta$.

A concept class $\mathscr{C}$ consisting of measurable sets is *PAC learnable under* $\mathcal{P}$, if there exists a PAC learning rule for $\mathscr{C}$ under $\mathcal{P}$. A class $\mathscr{C}$ is *consistently learnable* (under $\mathcal{P}$) if every learning rule consistent with $\mathscr{C}$ is PAC under $\mathcal{P}$. If $\mathcal{P} = P(\Omega)$ is the set of all probability measures, then $\mathscr{C}$ is said to be (distribution-free) *PAC learnable*. At the same time, learnability under intermediate families of measures on Ω has received considerable attention, cf. Chapter 7 in [V2].

Notice that in this paper, we only talk of *potential* PAC learnability, adopting a purely information-theoretic viewpoint.

A closely related concept is that of a *uniform Glivenko–Cantelli* concept class *with regard to a family of measures* $\mathcal{P}$, that is, a concept class $\mathscr{C}$ such that for each $\epsilon > 0$

$$\sup_{\mu \in \mathcal{P}} \mu^{\otimes n} \left\{ \sup_{C \in \mathscr{C}} |\mu(C) - \mu_n(C)| \geq \epsilon \right\} \to 0 \text{ as } n \to \infty. \tag{3}$$

(Cf. [D], Ch. 3; [M].) Here μ_n stands for the empirical (uniform) measure on n points, sampled in an i.i.d. fashion from Ω according to the distribution μ. One also says that $\mathscr{C}$ has the property of *uniform convergence of empirical measures* (*UCEM* property) *with regard to* $\mathcal{P}$ [V2].

Every uniform Glivenko–Cantelli class (with regard to $\mathcal{P}$) is PAC learnable (under $\mathcal{P}$), and in the distribution-free situation, the converse is true as well. Already in the case of learning under a single measure, it is not so: a PAC learnable class under a single distribution μ need not be uniform Glivenko-Cantelli with regard to μ (cf. Chapter 6 in [V2]). Not every PAC learnable class under non-atomic measures is uniform Glivenko–Cantelli with regard to non-atomic measures either: the class consisting of all finite and all cofinite subsets of Ω is a counter-example.

We say, following Pollard [P], that a concept class $\mathscr{C}$ consisting of measurable sets is *universally separable* if it contains a countable subfamily $\mathscr{C}'$ with the property that every $C \in \mathscr{C}$ is a pointwise limit of a suitable sequence $(C_n)_{n=1}^{\infty}$ of sets from $\mathscr{C}'$: for every $x \in \Omega$ there is N with the property that, for all $n \geq N$, $x \in C_n$ if $x \in C$, and $x \notin C_n$ if $x \in C$. Such a family $\mathscr{C}'$ is said to be *universally dense* in $\mathscr{C}$.

Probably the main source of uniform Glivenko–Cantelli classes is the finiteness of VC dimension. Assume that $\mathscr{C}$ satisfies a suitable measurability condition, for instance, $\mathscr{C}$ is image admissible Souslin, or else universally separable. (In particular, a countable $\mathscr{C}$ satisfies either condition.) If $\mathrm{VC}(\mathscr{C}) = d < \infty$, then $\mathscr{C}$ is uniform Glivenko–Cantelli, with a sample complexity bound that does not depend on $\mathscr{C}$, but only on ϵ, δ, and d. The following is a typical (and far from being optimal) such estimate, which can be deduced, for instance, along the lines of [M]:

$$s(\epsilon, \delta, d) \leq \frac{128}{\epsilon^2}\left(d \log\left(\frac{2e^2}{\epsilon} \log \frac{2e}{\epsilon}\right) + \log \frac{8}{\delta}\right). \tag{4}$$

For our purposes, we will fix any such bound and refer to it as a *"standard"* sample complexity estimate for $s(\epsilon, \delta, d)$.

A subset $N \subseteq \Omega$ is *universal null* if for every non-atomic probability measure μ on $(\Omega, \mathscr{A})$ one has $\mu(N') = 0$ for some Borel set N' containing N. Universal null Borel sets are just countable sets.

3 VC Dimension and Boolean Algebras

Recall that a *Boolean algebra*, $B = \langle B, \wedge, \vee, \neg, 0, 1\rangle$, consists of a set, B, equipped with two associative and commutative binary operations, $\wedge$ ("meet") and $\vee$ ("join"), which are distributive over each other and satisfy the absorption principles $a \vee (a \wedge b) = a$, $a \wedge (a \vee b) = a$, as well as a unary operation $\neg$ (complement), and two elements 0 and 1, satisfying $a \vee \neg a = 1$, $a \wedge \neg a = 0$.

For instance, the family 2^{Ω} of all subsets of a set Ω, with the union as join, intersection as meet, the empty set as 0 and Ω as 1, as well as the set-theoretic complement $\neg A = A^c$, forms a Boolean algebra. In fact, every Boolean algebra can be realized as an algebra of subsets of a suitable Ω. Even better, according to the Stone representation theorem, a Boolean algebra B is isomorphic to the Boolean algebra formed by all open-and-closed subsets of a suitable compact space, $S(B)$, called the *Stone space* of B, where the Boolean algebra operations are interpreted set-theoretically as above.

The space $S(B)$ can be obtained in different ways. For instance, one can think of elements of $S(B)$ as Boolean algebra homomorphisms from B to the two-element Boolean algebra $\{0,1\}$ (the algebra of subsets of a singleton). In this way, $S(B)$ is a closed topological subspace of the compact zero-dimensional space $\{0,1\}^B$ with the usual Tychonoff product topology.

The Stone space of the Boolean algebra $B = 2^{\Omega}$ is known as the *Stone-Čech compactification of* Ω, and is denoted $\beta\Omega$. The elements of $\beta\Omega$ are *ultrafilters* on Ω. A collection ξ of non-empty subsets of Ω is an ultrafilter if it is closed under finite intersections and if for every subset $A \subseteq \Omega$ either $A \in \xi$ or $A^c \in \xi$. To every point $x \in \Omega$ there corresponds a *trivial* (*principal*) *ultrafilter,* $\bar{x}$, consisting of all sets A containing x. However, if Ω is infinite, the Axiom of Choice assures that there exist non-principal ultrafilters on Ω. Basic open sets in the space $\beta\Omega$ are of the form $\bar{A} = \{\zeta \in \beta\Omega\colon A \in \zeta\}$, where $A \subseteq \Omega$. It is interesting to note that each $\bar{A}$ is at the same time closed, and in fact $\bar{A}$ is the closure of A in $\beta\Omega$. Moreover, every open and closed subset of $\beta\Omega$ is of the form $\bar{A}$.

A one-to-one correspondence between ultrafilters on Ω and Boolean algebra homomorphisms $2^{\Omega} \to \{0,1\}$ is this: think of an ultrafilter ξ on Ω as its own indicator function χ_{ξ} on 2^{Ω}, sending $A \subseteq \Omega$ to 1 if and only if $A \in \xi$. It is not difficult to verify that χ_{ξ} is a Boolean algebra homomorphism, and that every homomorphism arises in this way.

The book [Jo] is a standard reference to the above topics.

Given a subset $\mathscr{C}$ of a Boolean algebra B, and a subset X of the Stone space $S(B)$, one can regard $\mathscr{C}$ as a set of binary functions restricted to X, and compute the VC dimension of $\mathscr{C}$ over X. We will denote this parameter $\mathrm{VC}(\mathscr{C} \upharpoonright X)$.

A subset I of a Boolean algebra B is an *ideal* if, whenever $x, y \in I$ and $a \in B$, one has $x \vee y \in I$ and $a \wedge x \in I$. Define a *symmetric difference* on B by the formula $x \bigtriangleup y = (x \vee y) \vee \neg(x \wedge y)$. The *quotient Boolean algebra* B/I consists of all equivalence classes modulo the equivalence relation $x \sim y \iff x \bigtriangleup y \in I$. It can be easily verified to be a Boolean algebra on its own, with operations induced from B in a unique way.

The Stone space of B/I can be identified with a compact topological subspace of $S(B)$, consisting of all homomorphisms $B \to \{0,1\}$ whose kernel contains I. For instance, if $B = 2^{\Omega}$ and I is an ideal of subsets of Ω, then the Stone space of $2^{\Omega}/I$ is easily seen to consist of all ultrafilters on Ω which do not contain sets from I.

Theorem 2. *Let $\mathscr{C}$ be a concept class on a domain Ω, and let I be an ideal of sets on Ω. The following conditions are equivalent.*

1. *The VC dimension of the (family of closures of the) concept class $\mathscr{C}$ restricted to the Stone space of the quotient algebra $2^{\Omega}/I$ is at least n: $VC(\mathscr{C} \upharpoonright S(2^{\Omega}/I)) \geq n$.*
2. *There exists a family $A_1, A_2, \ldots, A_n$ of measurable subsets of Ω not belonging to I, which is shattered by $\mathscr{C}$ in the sense that if $J \subseteq \{1,2,\ldots,n\}$, then there is $C \in \mathscr{C}$ which contains all sets A_i, $i \in J$, and is disjoint from all sets A_i, $i \notin J$.*

Proof. (1)⇒(2). Choose ultrafilters $\xi_1, \ldots, \xi_n$ in the Stone space of the Boolean algebra $2^{\Omega}/I$, whose collection is shattered by $\mathscr{C}$. For every $J \subseteq \{1,2,\ldots,n\}$, select $C_J \in \mathscr{C}$

which carves the subset $\{\xi_i \colon i \in J\}$ out of $\{\xi_1, \dots, \xi_n\}$. This means $C_J \in \xi_i$ if and only if $i \in J$. For all $i = 1, 2, \dots, n$, set

$$A_i = \bigcap_{J \ni i} C_J \bigcap \bigcap_{J \not\ni i} C_J^c. \tag{5}$$

Then $A_i \in \xi_i$ and hence $A_i \notin I$. Furthermore, if $i \in J$, then clearly $A_i \subseteq C_J$, and if $i \notin J$, then $A_i \cap C_J = \emptyset$. The sets A_i are measurable by their definition.

(2)⇒(1). Let $A_1, A_2, \dots, A_n$ be a family of subsets of Ω not belonging to the set ideal I and shattered by $\mathscr{C}$ in sense of the lemma. For every i, the family of sets of the form $A_i \cap B^c$, $B \in I$ is a filter and so is contained in some free ultrafilter ξ_i, which is clearly disjoint from I and contains A_i. If $J \subseteq \{1, 2, \dots, n\}$ and $C_J \in \mathscr{C}$ contains all sets A_i, $i \in J$ and is disjoint from all sets A_i, $i \notin J$, then the closure $\bar{C}_J$ of C_J in the Stone space contains ξ_i if and only if $i \in J$. We conclude: the collection of ultrafilters ξ_i, $i = 1, 2, \dots, n$, which are all contained in the Stone space of $2^\Omega/I$, is shattered by the closed sets $\bar{C}_J$.

It follows in particular that the VC dimension of a concept class does not change if the domain Ω is compactified.

Corollary 1. *$VC(\mathscr{C} \upharpoonright \Omega) = VC(\mathscr{C} \upharpoonright \beta\Omega)$.*

Proof. The inequality $\mathrm{VC}(\mathscr{C} \upharpoonright \Omega) \leq \mathrm{VC}(\mathscr{C} \upharpoonright \beta\Omega)$ is trivial. To establish the converse, assume there is a subset of $\beta\Omega$ of cardinality n shattered by $\mathscr{C}$. Choose sets A_i as in Theorem 2,(2). Clearly, any subset of Ω meeting each A_i at exactly one point is shattered by $\mathscr{C}$.

Definition 1. *Given a concept class $\mathscr{C}$ on a domain Ω and an ideal I of subsets of Ω, we define the VC dimension of $\mathscr{C}$ modulo I,*

$$VC(\mathscr{C} \bmod I) = VC(\mathscr{C} \upharpoonright S(2^\Omega/I)).$$

That is, $VC(\mathscr{C} \bmod I) \geq n$ if and only if any of the equivalent conditions of Theorem 2 are met.

Definition 2. *Let $\mathscr{C}$ be a concept class on a domain Ω. If I is the ideal of all countable subsets of Ω, we denote the $VC(\mathscr{C} \bmod I)$ by $VC(\mathscr{C} \bmod \omega_1)$ and call it the* VC dimension modulo countable sets.

4 Finiteness of VC Dimension Modulo Countable Sets Is Necessary for Learnability

Lemma 1. *Every uncountable Borel subset of a standard Borel space supports a non-atomic Borel probability measure.*

Proof. Let A be an uncountable Borel subset of a standard Borel space Ω, that is, Ω is a Polish space equipped with its Borel structure. According to Souslin's theorem (see e.g. Theorem 3.2.1 in [A]), there exists a Polish (complete separable metric) space X

and a continuous one-to-one mapping $f\colon X \to A$. The Polish space X must be therefore uncountable, and so supports a diffuse probability measure, ν. The direct image measure $f_*\nu = \nu(f^{-1}(B))$ on Ω is a Borel probability measure supported on A, and it is diffuse because the inverse image of every singleton is a singleton in X and thus has measure zero.

The following result makes no measurability assumptions on the concept class.

Theorem 3. *Let $\mathscr{C}$ be a concept class on a domain $(\Omega, \mathscr{B})$ which is a standard Borel space. If $\mathscr{C}$ is PAC learnable under non-atomic measures, then the VC dimension of $\mathscr{C}$ modulo countable sets is finite.*

Proof. This is just a minor variation of a classical result for distribution-free PAC learnability (Theorem 2.1(i) in [BEHW]; we will follow the proof as presented in [V2], Lemma 7.2 on p. 279).

Suppose $\mathrm{VC}(\mathscr{C} \bmod \omega_1) \geq d$. According to Theorem 2, there is a family of uncountable Borel sets A_i, $i = 1, 2, \ldots, d$, shattered by $\mathscr{C}$ in our sense. Using Lemma 1, select for every $i = 1, 2, \ldots, d$ a non-atomic probability measure μ_i supported on A_i, and let $\mu = \frac{1}{d}\sum_{i=1}^{d}\mu_i$. This μ is a non-atomic Borel probability measure, giving each A_i equal weight $1/d$.

For every d-bit string σ there is a concept $C_\sigma \in \mathscr{C}$ which contains all A_i with $\sigma_i = 1$ and is disjoint from A_i with $\sigma_i = 0$. If A and B take constant values on all the sets A_i, $i = 1, 2, \ldots, d$, then $d_\mu(A, B)$ is just the normalized Hamming distance between the corresponding d-bit strings. Now, given $A \in \mathscr{C}$ and $0 \leq k \leq d$, there are

$$\sum_{k \leq 2\epsilon d} \binom{d}{k}$$

concepts B with $d_\mu(A, B) \leq 2\epsilon$. This allows to get the following lower bound on the number of pairwise 2ϵ-separated concepts:

$$\frac{2^d}{\sum_{k \leq 2\epsilon d} \binom{d}{k}}.$$

The Chernoff–Okamoto bound allows to estimate the above expression from below by $\exp[2(0.5 - 2\epsilon)^2 d]$. We conclude: the metric entropy of $\mathscr{C}$ with regard to μ is bounded below as:

$$M(2\epsilon, \mathscr{C}, \mu) \geq \exp[2(0.5 - 2\epsilon)^2 d].$$

The assumption $\mathrm{VC}(\mathscr{C} \bmod \omega_1) = \infty$ now implies that for every $0 < \epsilon < 0.25$,

$$\sup_{P \in \mathcal{P}} M(2\epsilon, \mathscr{C}, \mu) = \infty,$$

where $\mathcal{P}$ denotes the family of all non-atomic measures on Ω. By Lemma 7.1 in [V2], p. 278, the class $\mathscr{C}$ is not PAC learnable under $\mathcal{P}$.

5 The Universally Separable Case

Lemma 2. *Let $\mathscr{C}$ be a universally separable concept class, and let $\mathscr{C}'$ be a universally dense countable subset of $\mathscr{C}$. Then*

$$VC(\mathscr{C}) = VC(\mathscr{C}').$$

Proof. For every $C \in \mathscr{C}$ there is a sequence (C_n) of elements of $\mathscr{C}'$ with the property that for each $x \in \Omega$ there is N such that if $n \geq N$ and $x \in C$, then $x \in C_n$, and if $x \notin C$, then $x \notin C_n$. Equivalently, for every finite $A \subseteq \Omega$, there is an N so that whenever $n \geq N$, one has $C_n \cap A = C \cap A$. This means that if A is shattered by $\mathscr{C}$, it is equally well shattered by $\mathscr{C}'$. This established the inequaity $\mathrm{VC}(\mathscr{C}) \leq \mathrm{VC}(\mathscr{C}')$, while the converse inequality is obviously true.

Theorem 4. *For a universally separable concept class $\mathscr{C}$, the following conditions are equivalent.*

1. *$VC(\mathscr{C} \bmod \omega_1) \leq d$.*
2. *There exists a countable subset $A \subseteq \Omega$ such that $VC(\mathscr{C} \upharpoonright (\Omega \setminus A)) \leq d$.*

Proof. (1)⇒(2): Choose a countable universally dense subfamily $\mathscr{C}'$ of $\mathscr{C}$. Let $\mathscr{B}$ be the smallest Boolean algebra of subsets of Ω containing $\mathscr{C}'$. Denote by A the union of all elements of $\mathscr{B}$ that are countable sets. Clearly, $\mathscr{B}$ is countable, and so A is a countable set.

Let a finite set $B \subseteq \Omega \setminus A$ be shattered by $\mathscr{C}$. Then, by Lemma 2, it is shattered by $\mathscr{C}'$. Select a family $\mathscr{S}$ of $2^{|B|}$ sets in $\mathscr{C}'$ shattering B. For every $b \in B$ the set

$$[b] = \bigcap_{b \in C \in \mathscr{S}} C \bigcap \bigcap_{b \notin C \in \mathscr{S}} C^c$$

is uncountable (for it belongs to $\mathscr{B}$ yet is not contained in A), and the collection of sets $[b]$, $b \in B$ is shattered by $\mathscr{C}'$. This establishes the inequality $\mathrm{VC}(\mathscr{C} \upharpoonright (\Omega \setminus A)) \leq \mathrm{VC}(\mathscr{C} \bmod \omega_1)$.

(2)⇒(1): Fix an $A \subseteq \Omega$ so that $\mathrm{VC}(\mathscr{C} \bmod A^c) \leq d$. Suppose a collection of n uncountable sets A_i, $i = 1, 2, \ldots, n$ is shattered by $\mathscr{C}$ in our sense. The sets $A_i \setminus A$ are non-empty; pick a representative $a_i \in A_i \setminus A$, $i = 1, 2, \ldots, n$. The resulting set $\{a_i\}_{i=1}^n$ is shattered by $\mathscr{C}$, meaning $n \leq d$.

Corollary 2. *Let $\mathscr{C}$ be a universally separable concept class on a Borel domain Ω. If $d = VC(\mathscr{C} \bmod \omega_1) < \infty$, then $\mathscr{C}$ is a universal Glivenko-Cantelli class with regard to non-atomic measures and consistently PAC learnable under non-atomic measures.*

Proof. The class $\mathscr{C}$ has finite VC dimension in the complement to a suitable countable subset A of Ω, hence $\mathscr{C}$ is a universal Glivenko-Cantelli class (in the classical sense) in the standard Borel space $\Omega \setminus A$. But A is a universal null set in Ω, hence clearly $\mathscr{C}$ is universal Glivenko-Cantelli with regard to non-atomic measures.

The class $\mathscr{C}$ is distribution-free consistently PAC learnable in the domain $\Omega \setminus A$, with the standard sample complexity $s(\epsilon, \delta, d)$. Let $\mathcal{L}$ be any consistent learning rule for $\mathscr{C}$ in Ω. The restriction of $\mathcal{L}$ to $\Omega \setminus A$ (more exactly, to $\cup_{n=1}^{\infty} ((\Omega \setminus A)^n \times \{0,1\}^n))$ is a consistent learning rule for $\mathscr{C}$ restricted to the standard Borel space $\Omega \setminus A$, and together with the fact that A has measure zero with regard to any non-atomic measure, it implies that $\mathcal{L}$ is a PAC learning rule for $\mathscr{C}$ under non-atomic measures, with the same sample complexity function $s(\epsilon, \delta, d)$.

6 Martin's Axiom and Learnability

Martin's Axiom (MA) in one of its equivalent forms says that no compact Hausdorff topological space with the countable chain condition is a union of strictly less than continuum nowhere dense subsets. Thus, it can be seen as a strengthening of the statement of the Baire Category Theorem. In particular, the Continuum Hypothesis (CH) implies MA. However, MA is compatible with the negation of CH, and this is where the most interesting applications of MA are to be found. We will be using just one particular consequence of MA.

Theorem 5 (Martin-Solovay). *Let (Ω, μ) be a standard Lebesgue non-atomic probability space. Under MA, the Lebesgue measure is $2^{\aleph_0}$-additive, that is, if $\kappa < 2^{\aleph_0}$ and A_α, $\alpha < \kappa$ is family of pairwise disjoint measurable sets, then $\cup_{\alpha<\kappa} A_\alpha$ is Lebesgue measurable and*

$$\mu\left(\bigcup_{\alpha<\kappa} A_\alpha\right) = \sum_{\alpha<\kappa} \mu(A_\alpha).$$

In particular, the union of less than continuum null subsets of Ω is a null subset. □

For the proof and more on MA, see [K], Theorem 2.21, or [F], or [Je], pp. 563–565.

Lemma 3. *Let $\mathscr{C}$ be an infinite concept class on a measurable space Ω. Denote $\kappa = |\mathscr{C}|$ the cardinality of $\mathscr{C}$. There exists a consistent learning rule $\mathcal{L}$ for $\mathscr{C}$ with the property that for every $C \in \mathscr{C}$ and each n, the set*

$$\{\mathcal{L}(\sigma, C \cap \sigma) \colon \sigma \in \Omega^n\} \subseteq \mathscr{C} \tag{6}$$

has cardinality $< \kappa$. Under MA the rule $\mathcal{L}$ satisfies the measurability condition (1).

Proof. Choose a minimal well-ordering of elements of $\mathscr{C}$:

$$\mathscr{C} = \{C_\alpha \colon \alpha < \kappa\},$$

and set for every $\sigma \in \Omega^n$ and $\tau \in \{0,1\}^n$ the value $\mathcal{L}(\sigma, \tau)$ equal to C_β, where

$$\beta = \min\{\alpha < \kappa \colon C_\alpha \cap \sigma = \tau\},$$

provided such a β exists. Clearly, for each $\alpha < \kappa$ one has

$$\mathcal{L}(\sigma, C_\alpha \cap \sigma) \subseteq \{C_\beta \colon \beta \leq \alpha\},$$

which assures (6). Besides, the learning rule $\mathcal{L}$ is consistent.

Fix $C = C_\alpha \in \mathscr{C}$, $\alpha < \kappa$. For every $\beta \le \alpha$ define $D_\beta = \{\sigma \in \Omega^n \colon C \cap \sigma = C_\beta \cap \sigma\}$. The sets D_β are measurable, and the function

$$\Omega^n \ni \sigma \mapsto \mu(\mathcal{L}(C \cap \sigma) \bigtriangleup C) \in \mathbb{R}$$

takes a constant value $\mu(C_\beta \bigtriangleup C_\alpha)$ on each set $D_\beta \setminus \cup_{\gamma<\beta} D_\gamma$, $\beta \le \alpha$. Such sets, as well as all their possible unions, are measurable under MA by force of Martin–Solovay's Theorem 5, and their union is Ω^n. This implies the condition (1) for $\mathcal{L}$.

We again recall that a set $A \subseteq \Omega$ is *universal null* if it is Lebesgue measurable with regard to every non-atomic Borel probability measure μ on Ω and $\mu(A) = 0$.

Lemma 4 (Assuming MA). *Let $\mathscr{C}$ be a class of Borel subsets on a standard Borel space Ω. Suppose there is a natural d such that every countable subclass $\mathscr{C}' \subseteq \mathscr{C}$ has VC dimension $\le d$ outside of an universal null set (which depends on $\mathscr{C}$). Then every subclass of $\mathscr{C}$ of cardinality $< 2^{\aleph_0}$ has the same property.*

Proof. By induction on the cardinality of $\mathscr{C}$, which we denote α (notice that it never exceeds $2^{\aleph_0}$, and so the proof only makes sense under the negation of the Continuum Hypothesis). Suppose the result is true for all β, $\aleph_0 \le \beta < \alpha$. Choose a minimally well-ordered chain $\mathscr{C}_\gamma, \gamma < \alpha$ of subclasses of $\mathscr{C}$ whose union is $\mathscr{C}$. For every γ, let $\mathcal{N}_\gamma$ be a universal null subset of Ω with the property that $\mathscr{C}_\gamma$ has VC dimension $\le d$ outside of $\mathcal{N}_\gamma$. Martin–Sollovay's Theorem implies that $\mathcal{N} = \cup_{\gamma<\alpha} \mathcal{N}_\gamma$ is universal null. Consequently, each $\mathscr{C}_\gamma$ has VC dimension $\le d$ outside of $\mathcal{N}$, and the same applies to the union of the chain.

Lemma 5 (Assuming MA). *Let $\mathscr{C}$ be a concept class of cardinality $\kappa = |\mathscr{C}| < 2^{\aleph_0}$ on a standard Borel space Ω. If $d = VC(\mathscr{C})$ is finite, then $\mathscr{C}$ is a uniform Glivenko–Cantelli class, with a standard sample complexity estimate $s(\epsilon, \delta, d)$.*

Proof. A transfinite induction on κ. For $\kappa = \aleph_0$ the result is classical. Else, represent $\mathscr{C}$ as a union of an increasing transfinite chain of concept classes $\mathscr{C}_\alpha$, $\alpha < \kappa$, for each of which the statement of Lemma holds. For every $\epsilon > 0$ and $n \in \mathbb{N}$, the set

$$\left\{\sigma \in \Omega^n \colon \sup_{C \in \mathscr{C}} |\mu_n(\sigma) - \mu(C)| < \epsilon\right\} = \bigcap_{\alpha<\kappa} \left\{\sigma \in \Omega^n \colon \sup_{C \in \mathscr{C}_\alpha} |\mu_n(\sigma) - \mu(C)| < \epsilon\right\}$$

is measurable by Martin-Solovay's Theorem 5. Given $\delta > 0$ and $n \ge s(\epsilon, \delta, d)$, another application of the same result leads to conclude that for every $\mu \in P(\Omega)$:

$$\begin{aligned}\mu^{\otimes n}\left\{\sigma \in \Omega^n \colon \sup_{C \in \mathscr{C}} |\mu_n(\sigma) - \mu(C)| < \epsilon\right\} &= \mu^{\otimes n}\left(\bigcap_{\alpha<\kappa} \left\{\sigma \in \Omega^n \colon \sup_{C \in \mathscr{C}_\alpha} |\mu_n(\sigma) - \mu(C)| < \epsilon\right\}\right) \\ &= \inf_{\alpha<\kappa} \mu^{\otimes n}\left\{\sigma \in \Omega^n \colon \sup_{C \in \mathscr{C}_\alpha} |\mu_n(\sigma) - \mu(C)| < \epsilon\right\} \\ &\ge 1 - \delta,\end{aligned}$$

as required.

The following is an immediate consequence of two previous lemmas.

Lemma 6 (Assuming MA). *Under the assumptions of Lemma 4, every subclass of $\mathscr{C}$ of cardinality $< 2^{\aleph_0}$ is uniform Glivenko-Cantelli with regard to the family of non-atomic measures on Ω. The sample complexity of this class is the usual sample complexity $s(\delta,\epsilon,d)$ of concept classes of VC dimension $\leq d$.*

Lemma 7 (Assuming MA). *Let $\mathscr{C}$ be a concept class consisting of Borel subsets of a standard Borel space Ω. Assume that for some natural d, every countable subclass of $\mathscr{C}$ has VC dimension $\leq d$ outside of some universal null subset of Ω. Then the class $\mathscr{C}$ is PAC learnable under the family of all non-atomic measures on Ω, with the usual sample complexity $s(\delta,\epsilon)$ of distribution-free PAC learning concept classes of VC dimension $\leq d$.*

Proof. Using Lemma 3, choose a learning rule $\mathcal{L}$ for $\mathscr{C}$ with the property in Eq. (6). Since the family of all Borel subsets of Ω is well-known to have cardinality continuum, for every concept C and each n the cardinality of the image $\mathscr{L}_C = \mathcal{L}\{C\cap\sigma\colon \sigma\in\Omega^n\} \subseteq \mathscr{C}$ is strictly less than $2^{\aleph_0}$. By Lemma 6, $\mathscr{L}_C$ is a uniform Glivenko-Cantelli class with regard to non-atomic measures on Ω, satisfying the standard sample complexity bound. The proof is now concluded in a standard way.

7 The Proof of the Main Theorem

(1)⇒(2): this is Theorem 3.
(2)⇒(3): follows from Theorem 4.
(3)⇒(4): assume that for every d there is a countable subclass $\mathscr{C}_d$ of $\mathscr{C}$ with the property that the VC dimension of $\mathscr{C}_d$ is $\geq d$ after removing any countable subset of Ω. Clearly, the countable class $\cup_{d=1}^{\infty}\mathscr{C}_d$ will have infinite VC dimension outside of every countable subset of Ω, a contradiction.
(4)⇒(6): as a consequence of a classical result of Vapnik and Chervonenkis, every countable subclass $\mathscr{C}'$ is universal Glivenko-Cantelli with regard to all probability measures supported outside of some countable subset of Ω, and a standard bound for the sample complexity $s(\delta,\epsilon)$ only depends on d, from which the statement follows.
(6)⇒(5): trivial.
(5)⇒(3): modelling the classical argument that the uniform Glivenko-Cantelli property implies finite VC dimension, in exactly the same spirit as in the proof of our Theorem 3, one shows that the uniform Glivenko-Cantelli property of a concept class with regard to non-atomic measures implies a finite VC dimension modulo countable sets. But for a countable (more generally, universally separable) class $\mathscr{C}'$ this means finite VC dimension after a removal of a countable set, cf. Theorem 4.
(3)⇒(1): this is Lemma 7, and the only implication requiring Martin's Axiom.

The equivalence of (1), (7) and (8) in the universally separable case follows from Theorem 4 and Corollary 2. □

8 Conclusion

We have characterized concept classes $\mathscr{C}$ that are distribution-free PAC learnable under the family of all non-atomic probability measures on the domain. The criterion is obtained without any measurability conditions on the concept class, but at the expense of making a set-theoretic assumption in the form of Martin's Axiom. In fact, assuming MA makes things easier, and as this axiom is very natural, perhaps it deserves its small corner within the foundations of statistical learning.

It seems that generalizing the result from concept to function classes, using a version of the fat shattering dimension modulo countable sets, will not pose particular technical difficulties, and we plan to perform this extension in a full journal version of the paper, in order to keep the conference submission short. The Boolean algebras will however have to give way to commutative C^*-algebras [A].

It would be still interesting to know if the present results hold without Martin's Axiom, under the assumption that the concept class $\mathscr{C}$ is image admissible Souslin ([D], pages 186–187). The difficulty here is selecting a measurable learning rule $\mathcal{L}$ with the property that the images of all learning samples $(\sigma, C \cap \sigma)$, $\sigma \in \Omega^n$, are uniform Glivenko-Cantelli. An obvious route to pursue is the recursion on the Borel rank of $\mathscr{C}$, but we were unable to follow it through.

Now, a concept class $\mathscr{C}$ will be learnable under diffuse measures provided there is a hypothesis class $\mathscr{H}$ which has finite VC dimension and such that every $C \in \mathscr{C}$ differs from a suitable $H \in \mathscr{H}$ by a null set. If $\mathscr{C}$ consists of all finite and all cofinite subsets of Ω, this $\mathscr{H}$ is given by $\{\emptyset, \Omega\}$. One may conjecture that $\mathscr{C}$ is learnable under diffuse measures if and only if it admits such a "core" $\mathscr{H}$ having finite VC dimension. Is this true?

Another natural question is: can one characterize concept classes that are uniformly Glivenko–Cantelli with regard to all non-atomic measures? Apparently, this task requires yet another version of shattering dimension, which is strictly intermediate between Talagrand's "witness of irregularity" [T] and our VC dimension modulo countable sets. We do not have a viable candidate.

Finally, our investigation open up a possibility of linking learnability and VC dimension to Boolean algebras and their Stone spaces. This could be a glib exercise in generalization for its own sake, or maybe something deeper if one manages to invoke model theory and forcing.

References

[A] Arveson, F.: An Invitation to C^*-Algebras. Graduate Texts in Mathematics, vol. 39. Springer, Heidelberg (1976)

[BEHW] Blumer, A., Ehrenfeucht, A., Haussler, D., Warmuth, M.K.: Learnability and the Vapnik-Chervonenkis dimension. Journal of the ACM 36(4), 865–929 (1989)

[D] Dudley, R.M.: Uniform Central Limit Theorems. In: Cambridge Studies in Advanced Mathematics, Cambridge University Press, Cambridge (1999)

[F] Fremlin, D.H.: Consequences of Martin's Axiom. Cambridge Tracts in Mathematics, vol. 84. Cambridge University Press, Cambridge (1984)

[Je] Jech, T.: Set theory. Academic Press, New York (1978)

[Jo] Johnstone, P.T.: Stone Spaces. Cambridge Studies in Advanced Mathematics, vol. 3. Cambridge University Press, Cambridge (1986), Reprint of the 1982 edition

[K] Kunen, K.: Set Theory. North-Holland, Amsterdam (1980)

[M] Mendelson, S.: A few notes on statistical learning theory. In: Mendelson, S., Smola, A.J. (eds.) Advanced Lectures on Machine Learning. LNCS (LNAI), vol. 2600, pp. 1–40. Springer, Heidelberg (2003)

[P] Pollard, D.: Convergence of Stochastic Processes. Springer, New York (1984)

[T] Talagrand, M.: The Glivenko-Cantelli problem, ten years later. J. Theoret. Probab. 9(1996), 371–384 (1996)

[VC] Vapnik, V.N., Chervonenkis, A.Y.: On the uniform convergence of relative frequencies of events to their probabilities. Theory Probab. Appl. 16(2), 264–280 (1971)

[V1] Vidyasagar, M.: A theory of learning and generalization. With applications to neural networks and control systems. Communications and Control Engineering Series. Springer, London (1997)

[V2] Vidyasagar, M.: Learning and Generalization, with Applications to Neural Networks, 2nd edn. Springer, Heidelberg (2003)

A PAC-Bayes Bound for Tailored Density Estimation

Matthew Higgs and John Shawe-Taylor

Center for Computational Statistics and Machine Learning,
University College London
{mhiggs,jst}@cs.ucl.ac.uk

Abstract. In this paper we construct a general method for reporting on the accuracy of density estimation. Using variational methods from statistical learning theory we derive a PAC, algorithm-dependent bound on the distance between the data generating distribution and a learned approximation. The distance measure takes the role of a loss function that can be tailored to the learning problem, enabling us to control discrepancies on tasks relevant to subsequent inference. We apply the bound to an efficient mixture learning algorithm. Using the method of localisation we encode properties of both the algorithm and the data generating distribution, producing a tight, empirical, algorithm-dependent upper risk bound on the performance of the learner. We discuss other uses of the bound for arbitrary distributions and model averaging.

Keywords: Tailored density estimation, PAC-Bayes bounds, localisation.

1 Introduction

Estimating probability densities lies at the heart of statistical inference. Theoretical analyses of learning methods have generally focused on induction; learning a "rule" from a sequence of examples. More general cases of probabilistic inference involve two phases; the learning of a distribution and subsequently inference based on the learned model. This type of learning utilises underlying structure that is inherent in the data. It also produces machines that can be asked a variety of questions. If these questions are predefined then we can avoid accurately approximating the data distribution in the sense of the traditional metrics used in probability theory for problems of convergence. Instead, we tailor the measure of accuracy to the learning problem. We therefore consider a family of tasks and ask that the learned approximation be accurate on this class.

The supervised learning framework for pattern recognition has proved very effective in explaining the performance of principled, and even "off the shelf" algorithms. But many real world problems require a more complex modeling process composed of multiple stages of learning and inference. The accuracy of each step affects the performance of the learning machine, and therefore a formal theory for this type of inference must utilise all information available. The question is how to incorporate this information into a learning theory framework.

© Springer-Verlag Berlin Heidelberg 2010

Here we consider the problem of *tailored* density estimation, an idea formalised by [1]. If density estimation is part of some larger inference process, then we need to consider how this knowledge affects how we measure the accuracy of the density estimate. To construct error bounds we formalise our notion of generalisation to be the performance of an approximation, against the true distribution, on a predefined set of tasks. We assume that tasks lie in a reproducing kernel Hilbert space and use the technique of [2]; embedding distributions in the RKHS and performing moment matching in that space. It is then through the choice of kernel that we can tailor the density estimate to a function class of interest.

To construct the error bounds we use a toolbox of methods that is well established within a small group of the statistical learning community. PAC-Bayes theory is a framework for deriving some of the tightest generalisation bounds available. The traditional style bounds of [3], [4] and [5] all focus on classification. Extensions to regression [6] and [7] exist, but density estimation is only examined briefly in [8] and restricted to finite domains in [9]. In [8] an information theoretic loss is used. Here we derive a novel PAC-Bayes bound for density estimation over general domains, with a loss function that can be tailored to focus on functions of interest. We use techniques from [10] to commute the averaging inside the loss. We then apply the chain rule for relative entropy to give an upper risk bound on the performance of a non random algorithm. The upper bound for the relative entropy works by a novel choice of prior and posterior and is applicable to any random algorithm that learns a multinomial.

At the core of PAC-Bayes bounds lies a deviation inequality that restricts our freedom when choosing risk functionals and performance measures. To allow us to upper bound the complex Laplace transform we take a slight detour and derive a second order U-statistic PAC-Bayes bound that extends the bounds of [11] and [12] to more general convex performance measures. This detour allows us to commute both the sample average and the posterior averaging inside the loss at only the cost of a small multiplicative factor and a bias term of order $\mathcal{O}((n-1)^{-1})$. We show how the U-statistic bound implies our bound and specialise our bound to an efficient kernel moment matching algorithm that learns the mixing coefficients of a mixture model. Using the method of localisation we encode properties of both the algorithm and the data generating distribution, producing a tight, empirical, algorithm-dependent upper risk bound on the performance of the learner.

The paper is outlined as follows. Section 2 provides the necessary background on PAC-Bayes theory and the idea of localisation. In section 3 we state the required second order U-statistics bound and give its proof. In section 4 the reproducing Kernel Moment Matching (KMM) method is explained and we apply the U-statistic bound, with some tweaking, to the performance of an approximate distribution. In Section 5 we localise the bound for a finite mixture learning algorithm. This comprises of a subsection on stability analysis and a subsection on the probabilistic aspect of the localisation. Section 6 draws conclusions and discusses future work.

2 Technical Background

We assume all mathematical objects adhere to the measurability issues of the computations in which they are used. At the core of the analysis is the following PAC form of Markov's inequality.

Lemma 1 (The PAC lemma). *For any real-valued random variable V s.t. $\mathbb{E}e^V \leq 1$ and for any $\delta \in (0,1]$,*

$$\Pr\big\{V \leq \ln(\delta^{-1})\big\} \geq 1 - \delta. \tag{1}$$

We are given a random sample S and follow a PAC-Bayes learning paradigm where we are equipped with two distributions ρ and π on hypotheses h. Let $\phi(h,S)$ denote a measurable function. Define the random variable

$$V = \mathbb{E}_{h\sim\rho}\phi(h,S) - \mathrm{KL}(\rho||\pi) - \ln \mathcal{L}_\phi \tag{2}$$

where $\mathcal{L}_\phi = \mathbb{E}_S\mathbb{E}_{h\sim\pi}e^{\phi(h,S)}$ is the Laplace transform of ϕ with h drawn from π, and $\mathrm{KL}(\rho||\pi) = \mathbb{E}_{h\sim\rho}\ln\frac{d\rho(h)}{d\pi(h)}$ is the relative entropy between ρ and π. Using Jensen's inequality we have

$$\begin{aligned}
\mathbb{E}e^V &= \mathbb{E}_S e^{\mathbb{E}_{h\sim\rho}\phi(h,S) - \mathrm{KL}(\rho||\pi) - \ln\mathcal{L}_\phi} && (3)\\
&\leq \mathbb{E}_S\mathbb{E}_{h\sim\rho}\frac{d\pi(h)}{d\rho(h)}e^{\phi(h,S)-\ln\mathcal{L}_\phi} && (4)\\
&= \mathbb{E}_S\mathbb{E}_{h\sim\pi}e^{\phi(h,S)-\ln\mathcal{L}_\phi} && (5)\\
&= 1 && (6)
\end{aligned}$$

and the PAC Lemma is applicable. This proves the following.

Lemma 2 (The PAC-Bayes lemma). *For any two distributions π and ρ on hypotheses h, for any measurable $\phi(h,S)$, for any $\delta \in (0,1]$,*

$$\Pr_S\left\{\mathbb{E}_{h\sim\rho}\phi(h,S) \leq \mathrm{KL}(\rho||\pi) + \ln\frac{\mathcal{L}_\phi}{\delta}\right\} \geq 1-\delta, \tag{7}$$

where $\mathcal{L}_\phi = \mathbb{E}_S\mathbb{E}_{h\sim\pi}e^{\phi(h,S)}$.

Our choice of $\phi(h,S)$ describes the learning problem. Generally $\phi(h,S)$ will be some measure of how close a sample statistic is to its mean, such as the squared loss between a classifier's empirical and generalisation errors. It is ρ that encodes our choice of algorithm, e.g. letting ρ be a spherical gaussian centered at the output of an SVM. But there is always some ambiguity over the choice of π. For $\mathrm{KL}(\rho||\pi)$ to have a closed form π should belong to the same class as ρ and for $\mathcal{L}_\phi$ to be tractable it is required that π be independent of the sample. Essentially, π is a free parameter with constraints, that can be used to tailor and tighten a particular bound. An effective choice is to use a localised prior described below.

2.1 Localisation

The first notion of localisation was developed in [13] and occurs in related works [14], [8], [15] and [16]. The idea is to choose a prior π that is informative about the data-generating distribution and preferably one that allows us to upper bound the relative entropy $\mathrm{KL}(\rho||\pi)$. Let $R(h)$ denote some measure of risk and let $r_S(h)$ denote its empirical approximation and let $\rho(R)$ and $\rho(r_S)$ denote their expectations over ρ. A natural choice for ρ is the Gibbs estimator with density proportional to $e^{-\alpha r_S(h)}d\pi(h)$ for some flat prior π. This implies that the relative entropy $\mathrm{KL}(\rho||\pi)$ is likely to be small for the Gibbs estimator if we replace π by a distribution with density proportional to $\mathbb{E}_S e^{-\alpha r_S(h)}d\pi(h)$. In [13], [15] and [16] this is approximated by $e^{-\beta R(h)}d\pi(h)$, $\beta \in \mathbb{R}$, $\beta \leq \alpha$[1], and they obtain empirical bounds on the corresponding relative entropy (see [13], [15] and [16]).

In [12] they construct a Gaussian processes on the hypothesis space, in their case a Hilbert space $\mathcal{H}$, using posterior and prior with respective densities

$$q(h) := \frac{1}{Z}e^{-\frac{\gamma}{2}||h-\lambda_S||^2_{\mathcal{H}}} \quad ; \quad p(h) := \frac{1}{Z'}e^{-\frac{\gamma}{2}||h-\lambda||^2_{\mathcal{H}}} \tag{8}$$

where $\lambda_S := \mathrm{argmin}_h\, \Lambda(h,S)$ and $\lambda = \mathbb{E}_S[\mu_S]$, for some convex risk functional $\Lambda(h,S)$. In [12] they consider the case when $\Lambda(h,S) := r_S(h) + \eta||h||^2_{\tilde{\mathcal{H}}_S}$ where $||\cdot||_{\tilde{\mathcal{H}}_S}$ is some possibly data dependent norm on hypotheses. This is localisation, but without mention of a Gibbs estimator. It is a novel approach and we build on it in the application of our bound to the problem of mixture learning. We have the benefit of a finite dimensional problem, but the hassle of additional constraints. All the above methods of localisation output a randomised predictor, whereas here we upper bound an optimal value. Our KL term is well defined because this value is a multinomial and makes the method unique to the problem of mixture learning. We do not use $\mathbb{E}_S[\lambda_S]$ as a localised prior, instead we take $\mu := \mathrm{argmin}_\nu\, \mathbb{E}_S \Lambda(\nu, S)$. This is a novel localised prior and allows us to use a simple stability argument to upper bound $\mathrm{KL}(\lambda_S||\mu)$ and incorporate algorithmic properties.

3 Second Order *U*-Statistic PAC-Bayes' Bound

In this section we derive the required second order U-statistic PAC-Bayes bound, using ideas from [11] and Hoeffding's decomposition method, e.g. [17]. In [11] they focus on the Bernoulli relative entropy and work in a very general setting for non iid data. Here we specifically work with second order U-statistics and the proof holds for more general convex functions. From this point on we assume that $S = (x_1, \ldots, x_n)$ is a sample of data drawn iid from D.

Let $\{h\}$ denote a measurable set of bounded U-statistic kernels, $h : \mathbb{R}^d \times \mathbb{R}^d \to [a,b] \subset \mathbb{R}$ for all $h \in \{h\}$. For $S \sim D^n$, $S = (x_1, \ldots, x_n)$, let $U_S(h)$ denote the

[1] β is smaller than α due to the mixing in $\mathbb{E}_S e^{-\alpha r_S(h)}d\pi(h)$.

normalised 2^{nd} order U-statistic given by

$$U_S(h) = \frac{1}{n(n-1)} \sum_{i \neq j} \frac{h(x_i, x_j) - a}{b - a}, \tag{9}$$

and let $U(h) := \mathbb{E}_{S \sim D^m}[U_S(h)]$ denote its mean. By definition, $U_S(h)$ is an unbiased estimator of $U(h)$, i.e. $U(h) = \mathbb{E}_{(x,x') \sim D^2}[h(x, x')]$. Let $\mathcal{M}^1_+(\{h\})$ denote the set of probability measures on $\{h\}$. We have the following theorem.

Theorem 1 (2^{nd}-Order U-Process Bound). *For any $\pi \in \mathcal{M}^1_+(\{h\})$, for any convex $\mathcal{D} : [0,1]^2 \to \mathbb{R}$, for any $\delta \in (0,1]$, with probability $\geq 1 - \delta$ over the draw of S, for any $\rho \in \mathcal{M}^1_+(\{h\})$,*

$$\mathcal{D}\Big(\mathbb{E}_{h\sim\rho} U_S(h)), \mathbb{E}_{h\sim\rho} U(h)\Big) \leq \frac{1}{\lfloor n/2 \rfloor}\left[\mathrm{KL}(\rho||\pi) + \ln \frac{\mathcal{L}_\mathcal{D}}{\delta}\right], \tag{10}$$

where $\mathcal{L}_\mathcal{D} = \mathbb{E}_{h\sim\pi}\mathbb{E}_S e^{\lfloor n/2 \rfloor \mathcal{D}\left(U_S(h)), U(h)\right)}$.

Proof (Theorem 1). Define the random variable V given by

$$V := \lfloor n/2 \rfloor \mathcal{D}\big(\mathbb{E}_{h\sim\rho} U_S(h)), \mathbb{E}_{h\sim\rho} U(h)\big) - \mathrm{KL}(\rho||\pi) - \ln \mathcal{L}_\mathcal{D}. \tag{11}$$

For any convex $\mathcal{D} : [0,1]^2 \to \mathbb{R}$, using Jensen's inequality, we have

$$\begin{aligned} \mathbb{E} e^V &= \mathbb{E}_S e^{\lfloor n/2 \rfloor \mathcal{D}\left(\mathbb{E}_{h\sim\rho} U_S(h), \mathbb{E}_{h\sim\rho} U(h)\right) - \mathrm{KL}(\rho||\pi) - \ln \mathcal{L}_\mathcal{D}} & (12) \\ &\leq \mathbb{E}_S \mathbb{E}_{h\sim\rho} \tfrac{d\pi(h)}{d\rho(h)} e^{\lfloor n/2 \rfloor \mathcal{D}\left(U_S(h)), U(h)\right) - \ln \mathcal{L}_\mathcal{D}} & (13) \\ &= \mathbb{E}_S \mathbb{E}_{h\sim\pi} e^{\lfloor n/2 \rfloor \mathcal{D}\left(U_S(h)), U(h)\right) - \ln \mathcal{L}_\mathcal{D}} & (14) \\ &= 1. & (15) \end{aligned}$$

The theorem then follows from PAC lemma 1.

A common choice for $\mathcal{D}(a, b)$ is the Bernoulli relative entropy $\mathrm{kl}(a||b)$, or the closely related $\mathcal{F}_C(a, b) = \Phi_C(b) - C \cdot a$, for some $C \in \mathbb{R}$, where

$$\mathrm{kl}(a||b) := a \ln \frac{a}{b} + (1-a) \ln \frac{1-a}{1-b} \tag{16}$$

$$\Phi_C(b) := \ln \frac{1}{1 - [1 - e^{-C}]b}. \tag{17}$$

Here, these two choices lead to the following Corollary.

Corollary 1. *For any $\pi \in \mathcal{M}^1_+(\{h\})$, for any $C \in \mathbb{R}^+$, for any $\delta \in (0,1]$, with probability $\geq 1 - \delta$ over the draw of S, for any $\rho \in \mathcal{M}^1_+(\{h\})$,*

$$\mathrm{kl}\Big(\mathbb{E}_{h\sim\rho} U_S(h))\Big|\Big|\mathbb{E}_{h\sim\rho} U(h)\Big) \leq \frac{1}{\lfloor n/2 \rfloor}\left[\mathrm{KL}(\rho||\pi) + \ln \frac{\xi(\lfloor n/2 \rfloor)}{\delta}\right] \tag{18}$$

where $\xi(m) = \mathcal{O}(\sqrt{m})$, and

$$\mathcal{F}_C\Big(\mathbb{E}_{h\sim\rho} U_S(h)), \mathbb{E}_{h\sim\rho} U(h)\Big) \leq \frac{1}{\lfloor n/2 \rfloor}\left[\mathrm{KL}(\rho||\pi) + \ln \frac{1}{\delta}\right]. \tag{19}$$

Proof (Corollary 1). Let Σ_n denote the set of permutations on $\{1, \ldots, n\}$. Let $\bar{h}(\cdot) := \frac{h(\cdot)-a}{b-a}$. For $\sigma \in \Sigma_n$, let $B_{S,\sigma}(h)$ denote the iid block given by

$$B_{S,\sigma}(h) := \frac{1}{\lfloor n/2 \rfloor} \sum_{i=1}^{\lfloor n/2 \rfloor} \bar{h}(x_{\sigma(i)}, x_{\sigma(\lfloor n/2 \rfloor + i)}), \tag{20}$$

where each $\bar{h}(x_{\sigma(i)}, x_{\sigma(\lfloor n/2 \rfloor + i)}) \in [0, 1]$ and we have $U_S(h) = \frac{1}{n!} \sum_{\sigma \in \Sigma_n} B_{S,\sigma}(h)$. Using Jensen's inequality, we have

$$\mathcal{L}_{\mathcal{D}} = \mathbb{E}_{h \sim \pi} \mathbb{E}_S e^{\lfloor n/2 \rfloor \mathcal{D}\left(U_S(h), U(h)\right)} \tag{21}$$

$$\leq \frac{1}{n!} \sum_{\sigma \in \Sigma_n} \mathbb{E}_{h \sim \pi} \mathbb{E}_S e^{\lfloor n/2 \rfloor \mathcal{D}\left(B_{S,\sigma}(h), \mathbb{E}_S B_{S,\sigma}(h)\right)}. \tag{22}$$

Let $Z_{x_{\sigma(i)}}$ denote the unique, $\{0, 1\}$ valued random variable such that

$$\mathbb{E} Z_{x_{\sigma(i)}} = \mathbb{E}_S \bar{h}(x_{\sigma(i)}, x_{\sigma(\lfloor n/2 \rfloor + i)}), \tag{23}$$

and let $\widehat{Z} = \frac{1}{\lfloor n/2 \rfloor} \sum_{i=1}^{\lfloor n/2 \rfloor} Z_{x_{\sigma(i)}}$ denote its empirical mean. Note that $\widehat{Z}$ is a sum of $\lfloor n/2 \rfloor$ iid random variables. Using (Lemma 3) from [18], and the fact that $e^{\mathcal{D}(\cdot,\cdot)}$ is convex, we have

$$\mathbb{E}_S e^{\mathcal{D}(B_{S,\sigma}(h), \mathbb{E}_S B_{S,\sigma}(h))} \leq \mathbb{E} e^{\mathcal{D}(\widehat{Z}, \mathbb{E}\widehat{Z})}. \tag{24}$$

Therefore we can upper bound the Laplace transform $\mathcal{L}_{\mathcal{D}}$ in theorem 1 using the binomial deviation inequalities, (see [19], for example)

$$\mathbb{E} e^{\lfloor n/2 \rfloor \mathrm{kl}(\widehat{Z} || \mathbb{E}\widehat{Z})} \leq \xi(\lfloor n/2 \rfloor), \tag{25}$$

$$\mathbb{E} e^{\lfloor n/2 \rfloor \mathcal{F}_C(\widehat{Z}, \mathbb{E}\widehat{Z})} \leq 1, \tag{26}$$

where (assuming $0^0 = 1$)

$$\xi(m) := \sum_{k=0}^{m} \binom{m}{k} \left(\frac{k}{m}\right)^k \left(1 - \frac{k}{m}\right)^{m-k} = \mathcal{O}(\sqrt{m}). \tag{27}$$

4 Reproducing Kernel Moment Matching

Exposed in [20] and developed further in [21] is the novel technique for comparing distributions by mapping them to a reproducing kernel Hilbert space. Let k denote a reproducing kernel and let $\mathcal{H}$ denote its Hilbert space and let $C_k = \sup_{x,x'} k(x, x')$. Let P be any input distribution and define the mapping $P \mapsto k_P$, where $k_P(\cdot) = \mathbb{E}_{x \sim P}[k(x, \cdot)]$ and importantly $k_P \in \mathcal{H}$, (see [21]). For particular kernels this mapping is injective, meaning to differentiate between distributions we only need to examine the distance between their mappings as measured by the norm $|| \cdot ||_{\mathcal{H}}$. Therefore we can formalise density estimation as "the minimisation of the distance $||k_Q - k_D||_{\mathcal{H}}$ between an approximation Q and the true distribution D". As usual we only have access to a sample S drawn from D, and look to minimise the proxy distance $||k_Q - k_S||_{\mathcal{H}}$ where $k_S := \frac{1}{|S|}\sum_{i=1}^{|S|}[k(x_i,\cdot)]$. It is then natural to consider how close this proxy distance is to the true, and more importantly how close a minimiser of the proxy is to the true distribution.

It may not be required that the mapping $P \mapsto k_P$ be injective. For example, low order polynomial kernels are not dense in the space of continuous bounded functions and therefore the mapping of distributions to a low order polynomial feature space is not injective, but in [2, Table 4] they show how low order polynomial kernels are sufficient for the kernel moment matching algorithm to perform better on average on polynomial type tasks than various other density estimators. The bounds below hold for a large number of kernels and therefore gives us the freedom to choose a kernel that better suits a particular inference problem.

Here we consider parametric density estimation with a parameter space Θ. We construct an approximate input distribution Q_θ for $\theta \in \Theta$. Let k_{Q_θ} denote the two stage mapping $\theta \mapsto Q_\theta \mapsto k_{Q_\theta}$ and let $\mathcal{M}_+^1(\Theta)$ denote the set of probability measures on Θ. For $\rho \in \mathcal{M}_+^1(\Theta)$ define $\rho[k_{Q_\theta}] := \mathbb{E}_{\theta\sim\rho}[k_{Q_\theta}]$.

Theorem 2 (RKMM PAC-Bayes Bound). *For any $\pi \in \mathcal{M}_+^1(\Theta)$, for any convex $\mathcal{D} : [0,1]^2 \to \mathbb{R}$, for any $\delta \in (0,1]$, with probability $\geq 1-\delta$ over the draw of S, for any $\rho \in \mathcal{M}_+^1(\Theta)$,*

$$\mathcal{D}_{1/2}\left(\frac{n||\rho[k_{Q_\theta}] - k_S||_{\mathcal{H}}^2 - V(\rho)}{2C_k(n-1)}, \frac{||\rho[k_{Q_\theta}] - k_D||_{\mathcal{H}}^2}{2C_k}\right) \leq \frac{1}{\lfloor n/2 \rfloor}\left[2\mathrm{KL}(\rho||\pi) + \ln\frac{\mathcal{L}_{\mathcal{D}}}{\delta}\right]$$

where

$$V(\rho) := \frac{1}{n}\sum_{k=1}^{n}||\mathbb{E}_{\theta\sim\rho}[k_{Q_\theta}] - k_{x_k}||_{\mathcal{H}}^2 \quad ; \quad \mathcal{D}_{1/2}(a,b) := \mathcal{D}\Big(\tfrac{1}{2}a + \tfrac{1}{2}, \tfrac{1}{2}b + \tfrac{1}{2}\Big).$$

Proof (Theorem 2). For $(\theta,\theta') \in \Theta^2$ define the second order U-statistic kernel $h_{(\theta,\theta')}(x,x')$ such that

$$|h_{(\theta,\theta')}(x,x')| := |\langle k_{Q_\theta} - k_x, k_{\theta'} - k_{x'}\rangle_{\mathcal{H}}| \leq 2C_k. \tag{28}$$

We have

$$\begin{aligned}\mathbb{E}_{(\theta,\theta')\sim\rho^2}U\big(h_{(\theta,\theta')}\big) &= \frac{\mathbb{E}_{(\theta,\theta')\sim\rho^2}}{4C_k}\mathbb{E}_{(x,x')\sim D^2}\langle k_{Q_\theta} - k_x, k_{\theta'} - k_{x'}\rangle_{\mathcal{H}} + \tfrac{1}{2} \\ &= \frac{\mathbb{E}_{(\theta,\theta')\sim\rho^2}}{4C_k}\langle k_{Q_\theta} - k_D, k_{\theta'} - k_D\rangle_{\mathcal{H}} + \tfrac{1}{2} \\ &= \frac{1}{4C_k}||\mathbb{E}_{\theta\sim\rho}[k_{Q_\theta}] - k_D||_{\mathcal{H}}^2 + \tfrac{1}{2}.\end{aligned}$$

For any $\rho^2 \in \mathcal{M}_+^1(\Theta^2)$ we have

$$\begin{aligned}
\mathbb{E}_{(\theta,\theta')\sim\rho^2} U_S\big(h_{(\theta,\theta')}\big) &= \frac{\mathbb{E}_{(\theta,\theta')\sim\rho^2}}{n(n-1)} \sum_{i\neq j} \frac{\langle k_{Q_\theta} - k_{x_i}, k_{\theta'} - k_{x_j}\rangle_{\mathcal{H}} + 2C_k}{4C_k} \\
&= \frac{(4C_k)^{-1}}{n(n-1)} \sum_{i\neq j} \langle \mathbb{E}_{\theta\sim\rho}[k_{Q_\theta}] - k_{x_i}, \mathbb{E}_{\theta'\sim\rho}[k_{\theta'}] - k_{x_j}\rangle_{\mathcal{H}} + \tfrac{1}{2} \\
&= \frac{(4C_k)^{-1}}{n(n-1)} \Bigg[\sum_{i=1}^n \sum_{j=1}^n \langle \mathbb{E}_{\theta\sim\rho}[k_{Q_\theta}] - k_{x_i}, \mathbb{E}_{\theta'\sim\rho}[k_{\theta'}] - k_{x_j}\rangle_{\mathcal{H}} \\
&\qquad - \sum_{k=1}^n ||\mathbb{E}_{\theta\sim\rho}[k_{Q_\theta}] - k_{x_k}||_{\mathcal{H}}^2 \Bigg] + \tfrac{1}{2} \\
&= \frac{(4C_k)^{-1}}{n(n-1)} \Bigg[\Bigg\langle n\mathbb{E}_{\theta\sim\rho}[k_{Q_\theta}] - \sum_{i=1}^n k_{x_i}, n\mathbb{E}_{\theta'\sim\rho}[k_{\theta'}] \\
&\qquad - \sum_{j=1}^n k_{x_j} \Bigg\rangle_{\mathcal{H}} - \sum_{k=1}^n ||\mathbb{E}_{\theta\sim\rho}[k_{Q_\theta}] - k_{x_k}||_{\mathcal{H}}^2 \Bigg] + \tfrac{1}{2} \\
&= \frac{1}{4C_k(n-1)} \Big[n||\mathbb{E}_{\theta\sim\rho}[k_{Q_\theta}] - k_S||_{\mathcal{H}}^2 - V(\rho) \Big] + \tfrac{1}{2}.
\end{aligned}$$

For any $\rho^2, \pi^2 \in \mathcal{M}_+^1(\Theta^2)$ it holds that

$$\mathrm{KL}\big(\rho^2||\pi^2\big) = 2 \cdot \mathrm{KL}(\rho||\pi). \tag{29}$$

Inserting $\mathbb{E}_{(\theta,\theta')\sim\rho^2} U_S\big(h_{(\theta,\theta')}\big)$, its mean $\mathbb{E}_{(\theta,\theta')\sim\rho^2} U\big(h_{(\theta,\theta')}\big)$ and the relative entropy $\mathrm{KL}\big(\rho^2||\pi^2\big)$ into Theorem 1 completes the proof.

The preamble of the theorem has changed little from theorem 1, but there are important properties we wish to highlight. Firstly, note that $\mathcal{D}_{1/2}(a,b)$ is convex in both arguments and zero for $a = b$ and therefore not too different from $\mathcal{D}$. In the bound we pay roughly a multiplicative factor of 4, this is for commuting the sample and the posterior average inside the squared loss $||\cdot||_{\mathcal{H}}^2$. The introduction of the bias term $V(\rho)$ is a bi-product of the detour to U-statistics. In the worst case we have $V(\rho) \leq 2C_k$, but it is likely that $\mathbb{E}_{\theta\sim\rho}[k_{Q_\theta}]$ will be chosen to be close to k_S and therefore $V(\rho)$ will be small if the variance of the sample is not too large. Future investigations will look at removing this object from the bound, but note how it decays with $(n-1)^{-1}$ in a similar fashion to the bias between a U-statistic and a V-statistic (see, e.g. [17, Chapter 5]). The denominator C_k normalises the loss to $[0,1]$ so that the Laplace transform $\mathcal{L}_{\mathcal{D}}$ is easier to upper bound, but it is the same constant that occurs in Rademacher arguments for the convergence of empirical processes, where it is used as a measure of function class complexity. Thus, with regards to tailoring the loss function to a particular RKHS, the rescaling by $1/C_k$ means we can choose any kernel we wish and still measure performance on a generic $[0,1]$ scale. In the following section we show how the above bound can be specialised to a deterministic algorithm.

5 Localised Bound for Finite Mixture Models

Assume the distribution Q_θ is a mixture of the form

$$Q_\theta = \sum_{i=1}^{m+1} \theta_i Q_i, \quad \theta \in \triangle^m \tag{30}$$

where $\{Q_i\}_{i=1}^{m+1}$ a fixed set of component distributions and $\triangle^m$ is the unit simplex of dimension m. In this section we define a class of prior and posterior that result in a single parameter bound. This allows us to measure the performance of algorithms that focus on learning the best mixing coefficients θ for components $\{Q_i\}_{i=1}^{m+1}$. Let $\{\mathbf{e}_i\}_{i=1}^{m+1}$ be the canonical basis in $\mathbb{R}^{m+1}$ and for $\lambda, \mu \in \triangle^m$ let $\rho_\lambda, \pi_\mu \in \mathcal{M}_+^1(\triangle^m)$ be point-mass mixtures of the form

$$\rho_\lambda := \sum_{i=1}^{m+1} \lambda_i \delta_{\mathbf{e}_i} \quad ; \quad \pi_\mu := \sum_{i=1}^{m+1} \mu_i \delta_{\mathbf{e}_i}. \tag{31}$$

By the chain rule for relative entropy [22] it holds that

$$\mathrm{KL}\Big(\sum_{i=1}^{m+1} \lambda_i \delta_{\mathbf{e}_i} \Big|\Big| \sum_{i=1}^{m+1} \mu_i \delta_{\mathbf{e}_i} \Big) \leq \mathrm{KL}(\lambda || \mu). \tag{32}$$

For $\lambda, \mu \in \triangle^m$, we have $\mathbb{E}_{\theta \sim \rho_\lambda}[\theta] = \lambda$ and $\mathbb{E}_{\theta \sim \pi_\mu}[\theta] = \mu$, and importantly

$$\begin{aligned} \mathbb{E}_{\theta \sim \rho_\lambda}[k_{Q_\theta}(\cdot)] &= \mathbb{E}_{\theta \sim \rho_\lambda}\Big[\mathbb{E}_{x \sim Q_\theta}[k(x, \cdot)] \Big] & (33) \\ &= \mathbb{E}_{\theta \sim \rho_\lambda}\left[\sum_{i=1}^{m+1} \theta_i \mathbb{E}_{x \sim Q_i}[k(x, \cdot)] \right] & (34) \\ &= \sum_{i=1}^{m+1} \lambda_i \mathbb{E}_{x \sim Q_i}[k(x, \cdot)] & (35) \\ &= k_{Q_\lambda}(\cdot). & (36) \end{aligned}$$

Let Q_θ be a mixture distribution of the form of (30). For some regularisation parameter $r > 0$, define

$$\lambda_S := \underset{\theta \in \triangle^m}{\arg\min} \Big\{ ||k_{Q_\theta} - k_S||_{\mathcal{H}}^2 + r||\theta||_2^2 \Big\}. \tag{37}$$

Let λ_S define a posterior ρ_{λ_S} and define a localised prior π_{μ_D} where ρ_{λ_S} and π_{μ_D} are point-mass mixtures (31) and μ_D is given by

$$\mu_D := \underset{\theta \in \triangle^m}{\arg\min} \Big\{ ||k_{Q_\theta} - k_D||_{\mathcal{H}}^2 + r||\theta||_2^2 \Big\}. \tag{38}$$

Obviously π_{μ_D} is independent of the sample and therefore a valid prior. Using the reproducing property of the kernel and the mixture form of Q_θ it is easy to write both (37) and (38) as quadratic programs (something highlighted in [2]). Formally, we can expand the squared norm in (37) and (38), ignore constants and write them both using the quadratic function

$$\Omega_u(\theta) := \frac{1}{2}\theta^\top(\mathrm{A}+r\mathrm{I})\theta - u^\top\theta \tag{39}$$

where $r \geq 0$, $u \in \mathbb{R}^{m+1}$ and $A(i,j) := k_{Q_iQ_j} = \mathbb{E}_{x\sim Q_i, x'\sim Q_j}[k(x,x')]$, $i,j = 1,\ldots,m+1$. Taking $v_S, v_D \in \mathbb{R}^{m+1}$ such that $v_S(i) = k_{Q_iS}$ and $v_D(i) = k_{Q_iD}$ we have respectively $\lambda_S := \mathrm{argmin}_{\theta\in\triangle^m}\Omega_{v_S}(\theta)$ and $\mu_D := \mathrm{argmin}_{\theta\in\triangle^m}\Omega_{v_D}(\theta)$. By definition the ridged component matrix $(\mathrm{A}+r\mathrm{I})$ is positive definite and therefore both λ_S and μ_D are unique and well defined. We have the following theorem.

Theorem 3. *For any $r > 0$, let $\lambda_S \in \triangle^m$ be the optimal value in (37) and let $\sigma_r^{\min}(A)$ be the minimal eigenvalue of $(\mathrm{A}+r\mathrm{I})$. For any convex $\mathcal{D} : [0,1]^2 \to \mathbb{R}$ and any $\delta \in (0,1]$, with probability $\geq 1-\delta$ over the draw of S, it holds that*

$$\mathcal{D}_{1/2}\left(\frac{n||k_{Q_{\lambda_S}} - k_S||_{\mathcal{H}}^2 - V(\lambda_S)}{2C_k(n-1)}, \frac{||k_{Q_{\lambda_S}} - k_D||_{\mathcal{H}}^2}{2C_k}\right) \leq \frac{2B^* + \ln\frac{2\mathcal{L}_{\mathcal{D}}}{\delta}}{\lfloor n/2\rfloor}$$

where

$$B^* := \max_{\theta\in\triangle^m}\left\{\mathrm{kl}(\lambda_S||\theta) : \frac{||\lambda_S - \theta||_2}{\sqrt{m+1}} \leq \frac{C_k\left(2+\sqrt{\frac{1}{2}\ln\frac{2}{\delta}}\right)}{\sqrt{n}\cdot\sigma_r^{\min}(A)}\right\}. \tag{40}$$

Essentially, we have removed the complexity term $\mathrm{KL}(\rho||\pi)$ and replaced it with an empirical, algorithm dependent upper bound. The form of prior and posterior used has a big advantage over using Dirichlet of log-Normal distributions for ρ and π, where if the mean of either distribution approaches the boundary, $\mathrm{KL}(\rho||\pi)$ quickly approaches infinity. The value B^* is the optimum of a convex maximisation problem over convex constraints and the KKT conditions can be solved numerically. It is important to note that for certain values of δ the bound B^* can be infinite. This occurs when δ is small enough that the set $\left\{\theta : \frac{||\lambda_S-\theta||_2}{\sqrt{m+1}} \leq \frac{C_k\left(2+\sqrt{\frac{1}{2}\ln\frac{2}{\delta}}\right)}{\sqrt{n}\cdot\sigma_r^{\min}(A)}\right\}$ intersects the boundary of $\triangle^m$. Note, for a fixed δ, there will always exist a value of n large enough that this does not happen. This PAC-type value of n depends on the position of λ_S, the component matrix A and the regularisation parameter r, therefore the bound is fully data and algorithm dependent. Also, we would expect the complexity of the bound to increase as the parameter space $\triangle^m$ increases in dimension, but the normalising constant of $\sqrt{m+1}$ in the 2-norm essentially makes the bound independent of m. The proof of theorem 3 is split into the following two subsections.

5.1 Stability Analysis

In this subsection we examine the sensitivity of the reproducing KMM algorithm to changes in the data. These changes can be viewed as perturbations in the data vector u of (39). We apply techniques from [23, Section 4.4.1].

Proposition 1. *For any $v, w \in \mathbb{R}^{m+1}_{+}$, let $\widehat{\theta}_v$, $\widehat{\theta}_w$ be solutions to the quadratic program* $\min_{\theta\in\triangle^m} \Omega_u(\theta)$ *of form* (39) *for respectively $u = v$ and $u = w$. Let $\sigma_r^{\min}(A) > 0$ denote the minimal eigenvalue of the matrix* $(\mathrm{A} + r\mathrm{I})$. *It holds that*

$$||\widehat{\theta}_w - \widehat{\theta}_v||_2 \leq \frac{||w - v||_2}{\sigma_r^{\min}(A)}. \tag{41}$$

For the proof we require the following lemma.

Lemma 3. *Let $\sigma_r^{\min}(A) > 0$ denote the minimal eigenvalue of the matrix* $(\mathrm{A} + r\mathrm{I})$ *and let $\widehat{\theta}_u$ denote the minimiser of $\Omega_u(\theta)$ over $\triangle^m$. We have*

$$\Omega_u(\theta) - \Omega_u(\widehat{\theta}_u) \geq \sigma_r^{\min}(A)||\widehat{\theta}_u - \theta||_2^2, \quad \forall\theta \in \triangle^m. \tag{42}$$

Proof (Lemma 3). As $\widehat{\theta}_u$ is the minimiser of $\Omega_u(\theta)$, it holds that $\Omega_u(\widehat{\theta}_u) - \Omega_u(\theta) \leq 0 \; \forall\theta \in \triangle^m$ and therefore

$$0 \leq (\widehat{\theta}_u - \theta)^\top(\mathrm{A} + r\mathrm{I})(\widehat{\theta}_u - \theta) \leq u^\top(\widehat{\theta}_u - \theta), \quad \forall\theta \in \triangle^m, \tag{43}$$

where the first inequality follows from the fact that $(\mathrm{A} + r\mathrm{I})$ is positive definite. From (43) and definition of the Rayleigh quotient, we have

$$\begin{aligned} \Omega_u(\theta) - \Omega_u(\widehat{\theta}_u) &= (\theta - \widehat{\theta}_u)^\top(\mathrm{A} + r\mathrm{I})(\theta - \widehat{\theta}_u) + u^\top(\widehat{\theta}_u - \theta) &(44)\\ &\geq \sigma_r^{\min}(A)||\theta - \widehat{\theta}_u||_2^2. &(45)\end{aligned}$$

Proof (Proposition 1). Define $\Lambda(\theta) := \Omega_v(\theta) - \Omega_w(\theta)$. From Lemma 3, using Holder's inequality and the fact that $\widehat{\theta}_w$ minimises $\Omega_w(\theta)$ over $\triangle^m$, it holds that

$$\begin{aligned} \sigma_r^{\min}(A)||\widehat{\theta}_w - \widehat{\theta}_v||_2^2 &\leq \Omega_v(\widehat{\theta}_w) - \Omega_v(\widehat{\theta}_v) &(46)\\ &= \Lambda(\widehat{\theta}_w) - \Lambda(\widehat{\theta}_v) + \Omega_w(\widehat{\theta}_w) - \Omega_w(\widehat{\theta}_v) &(47)\\ &= (w - v)^\top(\widehat{\theta}_w - \widehat{\theta}_v) + \Omega_w(\widehat{\theta}_w) - \Omega_w(\widehat{\theta}_v) &(48)\\ &\leq ||w - v||_2||\widehat{\theta}_w - \widehat{\theta}_v||_2. &(49)\end{aligned}$$

Dividing through by $||\widehat{\theta}_w - \widehat{\theta}_v||_2$ completes the proof.

Remark 1. Note that the constraint $\widehat{\theta}_w - \widehat{\theta}_v \in \{\theta - \theta' \in \mathbb{R}^{m+1} : \theta, \theta' \in \triangle^m\}$ makes the above bound vacuous for $\frac{||w-v||_2}{\sigma_r^{\min}(A)} \geq \sqrt{2}$.

5.2 Norm Convergence

In this subsection we adapt a proof from [24] to bound the norm $||k_S - k_D||^2_{\mathcal{H}}$ with high probability. As we only require a bound on $||v_S - v_D||_2$ where $v_S(i) = k_{Q_iS}$ and $v_D(i) = k_{Q_iD}$, we reduce the problem to a finite dimensional one.

Proposition 2. *Let $v_S, v_D \in \mathbb{R}^{m+1}$ be be define as above. For any $\delta \in (0, 1]$, with probability $\geq 1 - \delta$ over the draw of S,*

$$||v_S - v_D||_2 \leq C_k\sqrt{\frac{m+1}{n}}\left(2 + \sqrt{\frac{1}{2}\ln\frac{1}{\delta}}\right). \tag{50}$$

Proof (Proposition 2). All norms in the proof are 2-norms. Define $g(S) = ||v_S - v_D||$. At the core of the proof is Mcdiarmid's inequality, therefore we first examine the stability of $g(S)$ with respect to changes in the sample. Let $S^{(i)}$ denote S with x_i replaced by x_i'. Then we have

$$\begin{aligned} |g(S) - g(S^{(i)})| &= ||v_S - v_D|| - ||v_{S^{(i)}} - v_D|| \leq ||v_S - v_{S^{(i)}}|| && (51)\\ &= \frac{1}{n}\left[\sum_{j=1}^{m+1} \left|k_{Q_j}(x_i) - k_{Q_j}(x_i')\right|^2\right]^{1/2} && (52)\\ &\leq \frac{\sqrt{m+1}C_k}{n}. && (53) \end{aligned}$$

Applying Mcdiarmid's inequality we obtain

$$\Pr\left\{g(S) - \mathbb{E}_S[g(s)] \geq \epsilon\right\} \leq \exp\left(-\frac{2n\epsilon^2}{(m+1)C_k^2}\right). \quad (54)$$

Let $\sigma = (\sigma_1, \ldots, \sigma_n)$ be Rademacher random variables. Using a symmetrisation argument on a second sample S' and Jensen's inequality and the concavity of the square root function, it holds that

$$\begin{aligned} \mathbb{E}_S[g(S)] &= \mathbb{E}_S||v_S - v_D|| = \mathbb{E}_S||v_S - \mathbb{E}_{S'}[v_{S'}]|| && (55)\\ &= \mathbb{E}_S||\mathbb{E}_{S'}[v_S - v_{S'}]|| \leq \mathbb{E}_{SS'}||v_S - v_{S'}|| && (56)\\ &= \mathbb{E}_{SS'\sigma}\left[\frac{1}{n}\left(\sum_{j=1}^{m+1}\left(\sum_{i=1}^{n}\sigma_i(k_{Q_j}(x_i) - k_{Q_j}(x_i'))\right)^2\right)^{1/2}\right] && (57)\\ &\leq 2\mathbb{E}_{S\sigma}\left[\frac{1}{n}\left(\sum_{j=1}^{m+1}\left(\sum_{i=1}^{n}\sigma_i k_{Q_j}(x_i)\right)^2\right)^{1/2}\right] && (58)\\ &\leq 2\mathbb{E}_S\left[\frac{1}{n}\left(\sum_{j=1}^{m+1}\mathbb{E}_\sigma\sum_{i,l=1}^{n}\sigma_i\sigma_l k_{Q_j}(x_i)k_{Q_j}(x_l)\right)^{1/2}\right] && (59)\\ &= 2\mathbb{E}_S\left[\frac{1}{n}\left(\sum_{j=1}^{m+1}\sum_{i=1}^{n}k_{Q_j}(x_i)k_{Q_j}(x_i)\right)^{1/2}\right] && (60)\\ &\leq 2\sqrt{\frac{m+1}{n}}C_k. && (61) \end{aligned}$$

Inserting the upper bound on $\mathbb{E}_S[g(S)]$ into (54) and setting $\delta = e^{-\frac{2n\epsilon^2}{(m+1)C_k^2}}$ completes the proof.

5.3 Proof of Theorem 3

Combining proposition 1 and 2, for any $\delta \in (0,1]$, with probability $\geq 1-\delta$ we have

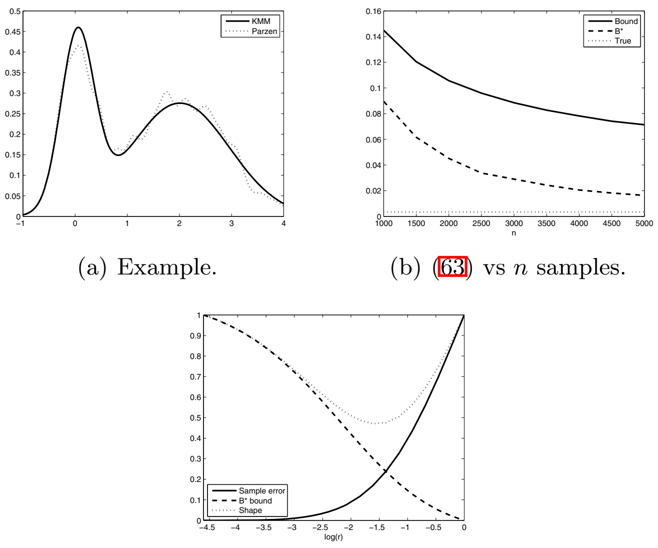
$$\frac{||\lambda_S - \mu_D||_2}{\sqrt{m+1}} \leq \frac{C_k\left(2 + \sqrt{\frac{1}{2}\ln\frac{2}{\delta}}\right)}{\sqrt{n}\cdot\sigma_r^{\min}(A)}, \quad (62)$$

where λ_S and μ_D are the minimisers of (37) and (38) respectively. Inserting into theorem 2 posterior and prior ρ_{λ_S} and π_{μ_D} respectively, of the form of (31), ensures that $\mathbb{E}_{\theta\sim\rho_{\lambda_S}}[k_{Q_\theta}] = k_{Q_{\lambda_S}}$ and, with high probability, $\mathrm{KL}(\rho_\lambda||\pi_\mu) \leq \mathrm{KL}(\lambda||\mu) \leq B^*$.

A simple numerical example for theorem 3 is shown in figure 1. Samples are drawn from a mixture of two Gaussians. Means and variances of two prototype Gaussians are constructed from the data using the EM algorithm. Mixing coefficients are then found using the KMM algorithm of (37). Note that the example is shown not to measure the performance of the algorithm, but to exemplify the validity of the bound[2]. Choosing $\mathcal{D}(a,b) = 2(a-b)^2$ we have

$$||k_{Q_{\lambda_S}} - k_D||^2_{\mathcal{H}} \leq \frac{n||k_{Q_{\lambda_S}} - k_S||^2_{\mathcal{H}} - V(\lambda_S)}{(n-1)} + \sqrt{\frac{2B^* + \ln\frac{2\mathcal{L}_{\mathcal{D}}}{\delta}}{\lfloor n/2 \rfloor}} \tag{63}$$

and for $m = 1$ we can compute B^* explicitly. Note in figure 1(b) how B^* decays with increasing sample size. This is the primary reason for using the localisation method. Also note how the bias term is negligible for reasonable sample sizes. Figure 1(c) shows a rescaled plot of the sample error $||k_{Q_{\lambda_S}} - k_S||^2_{\mathcal{H}}$ and the upper bound of theorem 3 against the regularisation parameter r. The shape plot is not exact due to rescaling and additional constants, but its shape indicates the expected trade off between data-fit and entropy.



(a) Example.

(b) (63) vs n samples.

(c) Increasing r.

Fig. 1. Numerical example for theorem 3

[2] We obviously benefit greatly from using the correct model.

6 Conclusions and Future Work

In this paper we have derived a PAC-Bayes bound for density estimation with a loss function that allows us to tailor the density estimate to a function class of interest. This places the work of [2] in a more principled framework and connects the theory of [1] to something more practical and applicable. We specialise the bound to finite mixture learning algorithms that minimise the kernel moment matching loss function and use methods of localisation to derive fully empirical and algorithm dependent upper risk bounds. Theorem 2 covers a huge number of learning scenarios, one of which is outlined below. Future work will look at extending the method to sparse solutions and considering the possibility of incorporating properties of the component matrix A into the norm convergence analysis of subsection 5.2. There is also the possibility of deriving the U-statistic bound directly for V-statistics; removing the need for a bias term.

If we let $\{\delta_x\}$, where δ_x is a unit mass at x, be the set of candidate distributions in theorem 2, then ρ and π become probability measures directly on the input space. The proof of the following corollary is omitted.

Corollary 2. *For any input distribution P, for any convex $\mathcal{D} : [0,1]^2 \to \mathbb{R}$, for any $\delta \in (0,1]$, with probability $\geq 1-\delta$ over the draw of S, for any input distribution Q,*

$$\mathcal{D}_{1/2}\left(\frac{n||k_Q - k_S||^2_{\mathcal{H}} - V(Q)}{2C_k(n-1)}, \frac{||k_Q - k_D||^2_{\mathcal{H}}}{2C_k}\right) \leq \frac{2\cdot \mathrm{KL}(Q||P) + \ln\frac{\mathcal{L}_{\mathcal{D}}}{\delta}}{\lfloor n/2 \rfloor}.$$

where $V(Q) := \frac{1}{n}\sum_{k=1}^{n} ||k_Q - k_{x_k}||^2_{\mathcal{H}}$.

The above corollary exposes the generality of the bounding method. Let $\mathcal{M}^1_+$ denote the space of all input distributions. We can define Q_S such that

$$Q_S := \underset{Q\in M\subseteq\mathcal{M}^1_+}{\arg\min} \left\{||k_Q - k_S||^2_{\mathcal{H}} + r\Gamma(Q)\right\} \tag{64}$$

where $\Gamma(Q)$ is a regularisation operator and M is a subset of $\mathcal{M}^1_+$. How to analyses the loss $||k_Q - k_S||^2_{\mathcal{H}}$ in a practical manner will form future work in this direction.

References

[1] Shawe-Taylor, J., Dolia, A.: A framework for probability density estimation. In: ICML (2007)

[2] Song, L., Zhang, X., Smola, A., Gretton, A., Schölkopf, B.: Tailoring density estimation via reproducing kernel moment matching. In: ICML 2008: Proceedings of the 25th international conference on Machine learning, pp. 992–999. ACM, New York (2008)

[3] McAllester, D.A.: PAC-Bayesian model averaging. In: Proceedings of the Twelfth Annual Conference on Computational Learning Theory, pp. 164–170. ACM Press, New York (1999)

[4] Seeger, M.: Pac-Bayesian generalisation error bounds for gaussian process classification. J. Mach. Learn. Res. 3, 233–269 (2003)
[5] Langford, J.: Tutorial on practical prediction theory for classification. J. Mach. Learn. Res. 6, 273–306 (2005)
[6] Audibert, J.Y.: Aggregated estimators and empirical complexity for least square regression. Annales de l'Institut Henri Poincare (B) Probability and Statistics 40(6), 685–736 (2004)
[7] Dalalyan, A., Tsybakov, A.B.: Aggregation by exponential weighting, sharp pac-bayesian bounds and sparsity. Mach. Learn. 72(1-2), 39–61 (2008)
[8] Zhang, T.: Information-theoretic upper and lower bounds for statistical estimation. IEEE Transactions on Information Theory 52(4), 1307–1321 (2006)
[9] Seldin, Y., Tishby, N.: A PAC-Bayesian approach to unsupervised learning with application to co-clustering analysis. Journal of Machine Learning Research, 1–46 (03 2010)
[10] Germain, P., Lacasse, A., Laviolette, F., Marchand, M.: A PAC-Bayes risk bound for general loss functions. In: Schölkopf, B., Platt, J., Hoffman, T. (eds.) Advances in Neural Information Processing Systems, vol. 19, pp. 449–456. MIT Press, Cambridge (2007)
[11] Ralaivola, L., Szafranski, M., Stempfel, G.: Chromatic PAC-Bayes Bounds for Non-IID Data. In: Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics AISTATS 2009. JMLR Workshop and Conference Proceedings, vol. 5, pp. 416–423 (2009)
[12] Lever, G., Laviolette, F., Shawe-Taylor, J.: Distribution dependent PAC-Bayes priors. Technical report, University College London (2010)
[13] Catoni, O.: A PAC-Bayesian approach to adaptive classification. Technical report, Laboratoire de Probabilités et Modéles Aléatoires, Universités Paris 6 and Paris 7 (2003)
[14] Audibert, J.Y.: A better variance control for PAC-Bayesian classification. Technical report, Laboratoire de Probabilités et Modéles Aléatoires, Universités Paris 6 and Paris 7 (2004)
[15] Alquier, P.: PAC-Bayesian bounds for randomized empirical risk minimizers. In: Mathematical Methods of StatisticS (2007)
[16] Catoni, O.: Pac-Bayesian supervised classification: The thermodynamics of statistical learning (2007)
[17] Serfling, R.J.: Approximation Theorems of Mathematical Statistics. John Wiley and Sons, Chichester (1980)
[18] Maurer, A.: A note on the PAC Bayesian theorem (2004)
[19] Germain, P., Lacasse, A., Laviolette, F., Marchand, M.: PAC-Bayesian learning of linear classifiers. In: ICML (2009)
[20] Smola, A., Gretton, A., Song, L., Schölkopf, B.: A Hilbert space embedding for distributions. In: Hutter, M., Servedio, R.A., Takimoto, E. (eds.) ALT 2007. LNCS (LNAI), vol. 4754, pp. 13–31. Springer, Heidelberg (2007)
[21] Sriperumbudur, B.K., Gretton, A., Fukumizu, K., Lanckriet, G.R.G., Shoelkopf, B.: Injective Hilbert space embeddings of probability measures. In: COLT, pp. 111–122. Omnipress (2008)
[22] Cover, T.M., Thomas, J.A.: Elements of information theory. Wiley Interscience, New York (1991)
[23] Bonnans, J., Shapiro, A.: Perturbation Analysis of Optimization Problems. Springer Series in Statistics. Springer, Heidelberg (2000)
[24] Shawe-Taylor, J., Cristianini, N.: Estimating the moments of a random vector. In: Proceedings of GRETSI 2003 Conference, vol. 1, p. 47–52 (2003)

Compressed Learning with Regular Concept[*]

Jiawei Lv[1], Jianwen Zhang[1], Fei Wang[2], Zheng Wang[1], and Changshui Zhang[1]

[1] State Key Laboratory on Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology(TNList)
Department of Automation, Tsinghua University, Beijing 100084, China
{lvjw05,jw-zhang06,w-z04}@mails.thu.edu.cn, zcs@mail.thu.edu.cn
[2] Department of Statistical Science, Cornell University
fw83@cornell.edu

Abstract. We revisit compressed learning in the PAC learning framework. Specifically, we derive error bounds for learning halfspace concepts with compressed data. We propose the *regularity* assumption over a pair of concept and data distribution to greatly generalize former assumptions. For a *regular* concept we define a *robust factor* to characterize the margin distribution and show that such a factor tightly controls the generalization error of a learned classifier. Moreover, we extend our analysis to the more general linearly non-separable case. Empirical results on both toy and real world data validate our analysis.

1 Introduction

The recent years have witnessed a surge of interest in *compressed learning* [2], i.e., learning with randomly projected data (compressed data) instead of original data. Compressed learning is necessary in two aspects: efficiency and privacy [13]. On one hand, learning with compressed data saves considerable running time and storage since random projection can effectively reduce the dimension of data. On the other hand, compressed learning can also be served as an important alternative for protecting data privacy. For example, in health care, security and finance related applications [9], data often contain sensitive information. Private database owners are only permitted to provide analyst with factitiously perturbed data rather than the original data. Random projection is a commonly used method to mask the original appearance of data for such privacy concerns [9]. In these scenarios, learning and analysis is only permitted to carried out on those randomly projected data. Existing works dealing with compressed data cover a variety of topics in machine learning such as classification, regression and manifold learning [1, 2, 4, 7, 10, 12, 13]. In this paper, we concentrate on the classification case.

A key issue in classification with compressed data is the learnability, whether the concept in the original space can be accurately learnt using the randomly projected data. There are two representative works on this problem.

[*] Supported by NSFC (Grant No. 60975003) and State Key Science and Technology Project on Marine Carbonate Reservoir Characterization (2008ZX05004-006).

© Springer-Verlag Berlin Heidelberg 2010

One is the loss function based analysis presented in [2]. They analyze the hinge loss of the linear classifier learned by support vector machine on compressed data and show the hinge loss of the compressedly learned classifier would not deviate much from that of the classifier learned on original data. However, they cannot guarantee the two classifiers have similar generalization error rates.

The other directly addresses the generalization error of a learned classifier on compressed data in the PAC framework [1]. The key factor affecting the generalization error of a learned classifier is the *margin distribution* of a concept, where the margin is defined as the distance between a sample and the separating boundary [6]. Since random projection perturbs data, a sample will very likely be wrongly classified if its margin is tiny. Therefore the learnability can only be guaranteed with some restrictions over the pair of concept and data distribution such that original data with small margins are of a nonessential proportion. Arriaga and Vempala [1] introduce the ℓ-robustness assumption ($\ell > 0$) which requires the margin of every sample is at least ℓ. Since random projection approximately preserves the margin of a finite set of samples [1], compressed learning of an ℓ-robust concept is in fact the classical problem of learning with a margin. They further show that an ℓ-robust halfspace concept can be accurately learned simply by a perceptron. However, the ℓ-robustness assumption is so restrictive that every halfspace concept is not ℓ-robust under many commonly adopted assumptions on data distributions (e.g., normal distributions and uniform distributions). Moreover, in real world problems, compressed data cannot always be well separated with a margin.

In this paper, we propose the *regularity* assumption to relax the *ℓ-robust* assumption to a more general case, under which the learnability of compressed learning halfspace concepts is revisited. The regularity assumption also imposes restrictions over a pair of concept and data distribution, but only requires that "almost" every sample has a nonzero margin. For example, a halfspace concept is regular with any continuous distribution. Hence we can work with more data distributions. However, under this relaxed assumption, compressed data may be wrongly classified since margins in the original space can be arbitrarily small. Therefore, we further define the *robust factor* for each regular concept to characterize the margin distribution. Under the regularity assumption, we revisit the problem of learning halfspace concepts with compressed data. We use the voted-perceptron algorithm proposed by Freund and Schapire [5] to perform the learning task on compressed data. Generalization error bounds based on the robust factor are derived for a learned classifier. We show that under some reasonable conditions on the robust factor, a regular halfspace concept can be accurately learned. The error analysis is also extended to the linearly non-separable case. Numerical experiments validate our analysis.

2 Preliminaries

The training set is denoted as $S = \{(x_1, y_1), ..., (x_m, y_m)\}$. $x_i \in \mathbb{R}^n$ is sampled independently from an unknown distribution $\mathcal{D}$ in $\mathbb{R}^n$ and then normalized onto the unit ball. The label $y_i \in \{-1, +1\}$ is given by an unknown halfspace concept $w \in \mathbb{R}^n$, i.e., $y_i = \operatorname{sign}\left(w^T x_i\right)$. Assume $\|w\| = 1$, where $\|\cdot\|$ is the Euclidean norm. R is a matrix of size $k \times n$. $w' = Rw$ is the projection of w under R. We use $w \cdot x$ to denote the inner product between w and x. $v(x) = |w \cdot x|$ is the distance between x and the hyperplane w. This is the margin of x with respect to w as used in [6]. The generalization error of a classifier $f(x)$ under distribution $\mathcal{D}$ is defined as

$$\mathrm{err}_{\mathcal{D}}(f) = \mathscr{P}_{\mathcal{D}}\{x : f(x) \neq y\},$$

where $\mathscr{P}_{\mathcal{D}}$ is the probability measure of $\mathcal{D}$.

2.1 Random Projection

Random projection is the technique of projecting a set of samples into a lower dimensional space by a random matrix. One of the most important properties of random projection is that it approximately preserve distance, which is stated by the following Johnson-Lindenstrauss lemma [1, 3, 8].

Theorem 1 (Johnson-Lindenstrauss lemma [1, 3, 8]). *Let $u, v \in \mathbb{R}^n$. R is a $k \times n$ random matrix with entries chosen independently from $N(0, 1/k)$. $u' = Ru$, $v' = Rv$. For any ϵ with $0 < \epsilon < 1$,*

$$\mathscr{P}\left\{(1-\epsilon)\|u-v\|^2 \leq \|u'-v'\|^2 \leq (1+\epsilon)\|u-v\|^2\right\} \geq 1 - e^{-\frac{\epsilon^2 k}{8}}.$$

The statement of Theorem 1 is due to Arriaga and Vempala [1].

In Theorem 1, the probability $\mathscr{P}$ is taken over the randomness of the matrix R. Specifically, if denoting $\mathbb{P}$ as the set of all the possible random matrices as in Theorem 1, then $\mathscr{P}$ is the probability measure over $\mathbb{P}$ defined by the construction of R. The corresponding distribution is denoted as $\mathrm{RP}(k, n)$.

Theorem 1 has a direct corollary as follows, which states that random projection also approximately preserves inner product.

Corollary 2. *Let $u, v \in \mathbb{R}^n$ with $\|u\|, \|v\| \leq 1$. Let $R \sim RP(k, n)$ and u', v' be the projections of u, v under R. Then for any $\epsilon > 0$,*

$$\mathscr{P}\{u \cdot v - \epsilon \leq u' \cdot v' \leq u \cdot v + \epsilon\} \geq 1 - 2e^{-\frac{\epsilon^2 k}{8}}.$$

Theorem 1 and Corallary 2 are crucial to our analysis.

2.2 ℓ-Robust Concepts

Our definition of a *regular* concept is motivated by the ℓ-robust concept, which is firstly introduced in [1].

Definition 3 (ℓ-robust halfspace concepts). *A half-space concept $w \in \mathbb{R}^n$ in conjunction with a distribution $\mathcal{D}$ in $\mathbb{R}^n$ is said to be ℓ-robust ($\ell > 0$), if*

$$\mathscr{P}_{\mathcal{D}}\{x : |w \cdot x| \leq \ell\} = 0. \tag{1}$$

Note that ℓ-robustness is an assumption placed over the couple $(w, \mathcal{D})$. It requires almost surely the margin of a sample is greater than ℓ. This margin measures how much one can alter a sample value without affecting its label. Therefore, the margin distribution characterizes the robustness of a concept to noise in sample values.

3 Regular Concepts and Robust Factors

In this section, we define regular halfspace concepts to generalize their ℓ-robust counterparts. For a regular concept, a corresponding robust factor is defined to characterize its margin distribution.

3.1 Regular Concepts

The formal definition of a regular concept is as follows.

Definition 4 (regular halfspace concepts). *Let $(w, \mathcal{D})$ be a halfspace concept and data distribution pair in $\mathbb{R}^n$. $(w, \mathcal{D})$ is called regular, if*

$$\mathscr{P}_{\mathcal{D}}\{x : |w \cdot x| = 0\} = 0. \tag{2}$$

When $(w, \mathcal{D})$ is regular, we call w a regular concept with respect to $\mathcal{D}$ or simply a regular concept when there is no confusion about $\mathcal{D}$. Clearly, an ℓ-robust concept is also a regular concept. A regular concept w requires points on the separating hyperplane forming a set of measure zero under $\mathcal{D}$. Therefore, almost surely every sample has a nonzero margin. Since under any continuous distribution in $\mathbb{R}^n$ the volume of an $n-1$ dimensional simplex is zero, every hyperplane is regular if the data distribution is continuous. Moreover, $(w, \mathcal{D})$ is not regular if there exists an $\epsilon > 0$, such that $\mathscr{P}_{\mathcal{D}}\{x : |w \cdot x| = 0\} = \epsilon$. Therefore, w is not regular when distributions are those placing nonzero measures on the separating hyperplane. This type of concept is unstable to noise in sample values, since any slight perturbation of a sample lying on the separating boundary will make it wrongly classified. We will not consider them here.

There exists an important property for a regular concept, which leads to the definition of the robust factor. The proof is a simple use of the monotone property of probability measures.

Proposition 5. *$(w, \mathcal{D})$ is regular if and only if for any $\epsilon > 0$, there exists an $\ell > 0$, such that*

$$\mathscr{P}_{\mathcal{D}}\{x : |w \cdot x| \leq \ell\} \leq \epsilon. \tag{3}$$

3.2 Robust Factors

The real number ℓ in (3) plays a similar role as that in an ℓ-robust concept, both characterizing the margin distribution of a concept. With a given $\epsilon > 0$, it is reasonable to believe that a pair $(w, \mathcal{D})$ with a larger ℓ corresponding to (3) is more robust to noise. This inspires us to introduce the following definition of the robust factor for a regular concept.

Definition 6 (robust factor). *The halfspace concept w is regular in conjunction with $\mathcal{D}$ in $\mathbb{R}^n$. Denote*

$$L_\epsilon = \{\ell \in [0,1] : \ell \, \textit{satisfies} \, \mathscr{P}_\mathcal{D}\{|w \cdot x| \leq \ell\} \leq \epsilon\}$$

for a given $\epsilon > 0$. Define $\phi(\epsilon) = \sup_{\ell \in L_\epsilon} \ell$ as the robust factor with respect to $(w, \mathcal{D})$.

It can be easily shown that $\phi(\epsilon) \in L_\epsilon$. When the distribution is continuous, $\phi(\epsilon)$ is simply the largest real number ℓ such that $\mathscr{P}_\mathcal{D}\{|w \cdot x| \leq \ell\} = \epsilon$. See the following examples, where $\phi(\epsilon)$ can be explicitly expressed or bounded.

Example 7 (robust factor of uniform distribution on the unit sphere in $\mathbb{R}^3$). Let $w = [1, 0, 0]^T$ be the halfspace concept. $\mathcal{D}$ is the uniform distribution on the unit sphere in $\mathbb{R}^3$. Then $\phi(\epsilon) = \frac{\epsilon}{4\pi}$.

Generally, the robust factor of uniform distribution on the unit sphere in $\mathbb{R}^n$ has the following upper bounds.

Example 8. $w = [1, 0, \cdots, 0]^T \in \mathbb{R}^n \, (n > 3)$ is the halfspace concept. $\mathcal{D}$ is the uniform distribution on the unit sphere in $\mathbb{R}^n$. Then for any $\epsilon \in (0, 1)$, we have

$$\phi(\epsilon) \leq \begin{cases} \frac{\Gamma(\frac{n}{2})\epsilon}{8\pi^{(n-1)/2}} & n \, \text{is even}, \\ \sin\left(\frac{\Gamma(\frac{n}{2})\epsilon}{4\pi^{(n-1)/2}}\right) & n \, \text{is odd}. \end{cases}$$

Proof. Transforming to the spherical coordinate system, we can represent the n Cartesian coordinates as follows

$$\begin{aligned} x_1 &= \cos\theta_1, \\ x_2 &= \sin\theta_1 \cos\theta_2, \\ &\cdots, \\ x_{n-1} &= \sin\theta_1 \cdots \sin\theta_{n-2} \cos\theta_{n-1}, \\ x_n &= \sin\theta_1 \cdots \sin\theta_{n-2} \sin\theta_{n-1}, \end{aligned}$$

where $\theta_i \in [0, \pi]$ for $1 \leq i \leq n-2$ and $\theta_{n-1} \in [0, 2\pi]$. Furthermore, we have $|w \cdot x| = |\cos\theta_1|$. Let $\theta \in [0, \pi/2]$ such that $\cos\theta = \phi(\epsilon)$. Denote $A_{\phi(\epsilon)} = \mathscr{P}_\mathcal{D}\{|w \cdot x| \leq \phi(\epsilon)\}$. We have

$$\begin{aligned} A_{\phi(\epsilon)} =& \int_{\theta_1=\theta}^{\pi-\theta} \int_{\theta_2=0}^{\pi} \cdots \int_{\theta_{n-2}=0}^{\pi} \int_{\theta_{n-1}=0}^{2\pi} dS \\ =& \int_{\theta}^{\pi-\theta} d\theta_1 \sin^{n-2}\theta_1 \int_0^\pi d\theta_2 \sin^{n-3}\theta_2 \cdots \int_0^\pi d\theta_{n-2} \sin\theta_{n-2} \int_0^{2\pi} d\theta_{n-1} \\ =& \frac{2\pi^{(n-1)/2}}{\Gamma\left(\frac{n-1}{2}\right)} \int_\theta^{\pi-\theta} d\theta_1 \sin^{n-2}\theta_1, \end{aligned}$$

where dS is the area element of the n-sphere, i.e.,

$$dS = \sin^{n-2}\theta_1 \sin^{n-3}\theta_2 \cdots \sin\theta_{n-2} \, d\theta_1 \cdots d\theta_{n-1},$$

and we obtain the last equality by using the surface area formula of the n-sphere. By the technique of integration by parts, we have the following recurrence formula:

$$\int_{\theta}^{\pi-\theta} \sin^{n-2} t\, dt = \frac{2\sin^{n-3}\theta\cos\theta}{n-2} + \frac{n-3}{n-2}\int_{\theta}^{\pi-\theta} \sin^{n-4} t\, dt.$$

Since $\theta \in [0, \pi/2]$,

$$\int_{\theta}^{\pi-\theta} \sin^{n-2} t\, dt \geq \frac{n-3}{n-2}\int_{\theta}^{\pi-\theta} \sin^{n-4} t\, dt. \tag{4}$$

If n is even, by (4), we have

$$\int_{\theta}^{\pi-\theta} \sin^{n-2} t\, dt \geq \frac{(n-3)(n-5)\cdots 1}{(n-2)(n-4)\cdots 2}\int_{\theta}^{\pi-\theta} dt = 2\frac{\Gamma(\frac{n-1}{2})}{\Gamma(\frac{n}{2})}\left(\frac{\pi}{2}-\theta\right).$$

Therefore,

$$\epsilon = A_{\phi(\epsilon)} \geq \frac{4\pi^{(n-1)/2}}{\Gamma(\frac{n}{2})}\left(\frac{\pi}{2}-\theta\right).$$

We have $\frac{\pi}{2} - \theta \leq \frac{\Gamma(\frac{n}{2})\epsilon}{4\pi^{(n-1)/2}}$. And the robust factor can be bounded by

$$\phi(\epsilon) = \cos\theta \leq \sin\left(\frac{\Gamma(\frac{n}{2})\epsilon}{4\pi^{(n-1)/2}}\right).$$

If n is odd, we have

$$\int_{\theta}^{\pi-\theta} \sin^{n-2} t\, dt \geq \frac{(n-3)(n-5)\cdots 2}{(n-2)(n-4)\cdots 3}\int_{\theta}^{\pi-\theta} \sin t\, dt = 4\frac{\Gamma(\frac{n-1}{2})}{\Gamma(\frac{n}{2})}\cos\theta.$$

Therefore,

$$\epsilon = A_{\phi(\epsilon)} \geq \frac{8\pi^{(n-1)/2}}{\Gamma(\frac{n}{2})}\cos\theta.$$

And this directly gives the upper bound of the robust factor, i.e.,

$$\phi(\epsilon) \leq \frac{\Gamma(\frac{n}{2})\epsilon}{8\pi^{(n-1)/2}}.$$

Thus the result follows.

The robust factor of an ℓ-robust concept satisfies $\phi(0) = \ell > 0$. This is a very strong condition and does not generally hold for a regular concept. We only assume $\phi(0) = 0$ in the following discussion. Hence, as $\epsilon \to 0$, $\phi(\epsilon)$ also decreases to 0. The speed that $\phi(\epsilon)$ approaches zero is an important characteristic of the margin distribution which greatly affects the learning result. We prefer those decreasing slowly near $\epsilon = 0$. This is because the robust factor with a smaller decreasing rate will eventually take a larger value when ϵ is sufficiently close to zero. Only for those concepts with their robust factors not decreasing too fast near $\epsilon = 0$, can we learn a classifier accurate enough. Based on this idea, we introduce the following additional assumption to control the decreasing rate of a robust factor. This assumption can be viewed as a slight variant of Tsybakov's noise condition [11].

Assumption 9. *Suppose there exist constants $0 < \epsilon_0 < 1$, $C > 0$ and $0 \leq \alpha < 1/4$ such that for every $\epsilon \in [0, \epsilon_0)$, $\phi(\epsilon)$ satisfies*

$$\phi(\epsilon) \geq C\epsilon^{\alpha}. \tag{5}$$

If the assumption holds, there does not exist such a neighborhood of the origin that $\phi(\epsilon) \leq \bar{C}\epsilon^{\beta}$ holds for all ϵ in the neighborhood, where $\bar{C} > 0$ and $\beta > \alpha$. And the rate of $\phi(\epsilon)$ decreasing to zero is no faster than that of a polynomial with order less than 1/4. Not all robust factors satisfy this assumption, e.g., the above two examples. We will give experimental results on toy data to show that failing to satisfy it will greatly affect the accuracy of the compressedly learned classifier. It will also be shown in the following section that if the assumption holds, an sufficiently accurate classifier can be learned based on compressed data.

4 Learning Regular Halfspace Concepts

In this section, we present error bounds for learning regular halfspace concepts with compressed data. We first summarize our main results then give the proofs.

4.1 Main Results

We use the voted-perceptron algorithm proposed in [5] as the base algorithm to learn from compressed data. The outputs of the algorithm is a weighted ensemble of linear classifiers. There is a parameter T in the algorithm which represents the number of iterations. In the following analysis, we simply set $T = 1$ for convenience. Note that our main results are not particular for the base algorithm, since the analysis does not depend on the detailed structure of the algorithm. In fact, any algorithm with a comparable generalization guarantee can be used to obtain similar results.

Theorem 10. *Let $w \in \mathbb{R}^n$ be a halfspace concept with $\|w\| = 1$. $\mathcal{D}$ is a distribution over $\mathbb{R}^n$. Suppose $(w, \mathcal{D})$ is regular. Let $R \sim RP(k, n)$ be a random matrix. S is the training set of size m. S' is its projection under R. For any given $\epsilon > 0$ and $\delta > 4\epsilon$, with probability at least $1-\delta$, the generalization error of the classifier $f(x)$ output by the voted-perceptron algorithm based on S' satisfies*

$$err_{\mathcal{D}}\left(f\right) \leq \frac{1}{\phi^2}\left(\frac{16\left(1+\epsilon\right)^2 + 16\delta}{m+1} + 40\epsilon + 18\delta\right), \tag{6}$$

if $k = \max\left\{\frac{32}{\phi^2}\log\frac{2}{\epsilon(\delta/4-\epsilon)}, \frac{8}{\epsilon^2}\log\frac{2(m+2)}{\delta}\right\}$ *where* $\phi = \phi\left(\epsilon^2\right)$*, provided* $m+1 \geq \frac{1}{2\epsilon^2}\log(8/\delta)$.

The generalization error depends critically on the robust factor. If the robust factor takes a larger value, the learned classifier will be more accurate. This is consistent with our intuition that, if samples own larger margins, the learning problem should be easier.

Also, as the sample size m approaches infinity, the upper bound approaches $\frac{c(\epsilon+\delta)}{\phi^2}$, where c is some constant. And the convergence rate is $1/m$. Hence, when training sample is enough, the accuracy of the compressedly learned classifier is determined by the intrinsic hardness of the problem, i.e., the margin distribution.

As a simple corollary of Theorem 10, it can be shown that under Assumption 9, the learned classifier $f(x)$ can be arbitrarily accurate by suitably choosing the dimension k and sample size m. This guarantees we can learn a regular halfspace concept with satisfactory accuracy on compressed data.

Corollary 11. *We invoke Assumption 9. Let* α *and* ϵ_0 *be the constants from Assumption 9. For any* $\epsilon > 0$*, let* $\epsilon_1 = \min\left\{\epsilon_0, c_1(\epsilon/2)^{\frac{1}{1-4\alpha}}\right\}$*, where* c_1 *is some fixed constant. If* δ *satisfies* $4\epsilon_1 < \delta < c_2\epsilon_1$ *for some constant* c_2*, then with probability at least* $1-\delta$*, the learned classifier* $f(x)$ *in Theorem 10 satisfies*

$$err_{\mathcal{D}}(f) \leq \epsilon,$$

if $k = \max\left\{\frac{32}{\phi^2}\log\frac{2}{\epsilon_1(\delta/4-\epsilon_1)}, \frac{8}{\epsilon_1^2}\log\frac{2(m+2)}{\delta}\right\}$ *and* $m \geq \max\left\{160/\phi^2, \frac{\log(8/\delta)}{2\epsilon_1^2}\right\}$*, where* $\phi = \phi\left(\epsilon_1^2\right)$.

Specifically, when $(w, \mathcal{D})$ is ℓ-robust, the sample complexity in Corollary 11 is $\tilde{O}\left(1/\ell^2\right)$ in terms of ℓ, the same order as Arriaga and Vempala's [1], where the $\tilde{O}(\cdot)$ notation hides the logarithmic factors. However our lower bound on the projection dimension k is $k \geq \Omega(\log m)$, which improves the bound $k \geq \Omega(m)$ obtained in [1] .

We will give the detailed proof of Theorem 10 in the following part of this section. The proof of Corollary 11 is direct and mainly technical. We omit it here.

4.2 Proofs

For a given w, define the product set $E \subset \mathbb{P} \times \mathbb{R}^n$ as

$$E = \{(R, x) : \text{sign}(w \cdot x) \neq \text{sign}(w' \cdot x')\},$$

where $w' = Rw$, $x' = Rx$. For a point $x \in \mathbb{R}^n$, the x *cross section* of E is defined as $E_x = \{R : (R, x) \in E\} = \{R : \text{sign}(w \cdot x) \neq \text{sign}(w' \cdot x')\}$. Similarly, the R cross section of E is $E_R = \{x : (R, x) \in E\} = \{x : \text{sign}(w \cdot x) \neq \text{sign}(w' \cdot x')\}$. We first need to build some propositions and lemmas.

Lemma 12. *Given $w, x \in \mathbb{R}^n$ with $\|w\| = \|x\| = 1$, for any $\ell \in (0,1)$,*

$$\mathscr{P}\{E_x\} \leq 2e^{-\frac{\ell^2 k}{8}} + I\left(|w \cdot x| \leq \ell\right)$$

holds, where $I(\cdot)$ is the indicator function.

Proof. Decompose E_x according to whether $F = \{|w \cdot x| \leq \ell\}$ happens or not. Since $E_x \cap F \subseteq F$ and $E_x \cap F^c \subseteq \{|w \cdot x - w' \cdot x'| > \ell\}$, the lemma follows directly by Proposition 5 and Corollary 2.

Proposition 13. *Let $w \in \mathbb{R}^n$ be a halfspace concept. $\|w\| = 1$. $\mathcal{D}$ is a distribution over $\mathbb{R}^n$. Suppose $(w, \mathcal{D})$ is regular. $R \sim RP(k,n)$ and $w' \in \mathbb{R}^k$ is the projection of w under R. Then for any $0 < \epsilon < \delta$, we have*

$$\mathscr{P}\{err_{\mathcal{D}}(w') > \epsilon\} \leq \delta,$$

if $k \geq \frac{8}{\phi(\epsilon^2)^2} \log \frac{2}{\epsilon(\delta - \epsilon)}$.

Proof. Clearly, $|\mathrm{err}_{\mathcal{D}}(w) - \mathrm{err}_{\mathcal{D}}(w')| = \mathrm{err}_{\mathcal{D}}(w')$. We will first show that for a fixed $R \in \mathbb{P}$,

$$|\mathrm{err}_{\mathcal{D}}(w) - \mathrm{err}_{\mathcal{D}}(w')| \leq \mathscr{P}_{\mathcal{D}}\{E_R\}. \tag{7}$$

In fact by rewriting probability into an integral of the corresponding indicator function, we have

$$\begin{aligned}
\left|\mathrm{err}_{\mathcal{D}}(w) - \mathrm{err}_{\mathcal{D}}\left(w'\right)\right| &= \left|\int I\left(\mathrm{sign}\left(w \cdot x\right) \neq y\right)\, d\mathscr{P}_{\mathcal{D}} - \int I\left(\mathrm{sign}\left(w' \cdot x'\right) \neq y\right)\, d\mathscr{P}_{\mathcal{D}}\right| \\
&\leq \int \left|I\left(\mathrm{sign}\left(w \cdot x\right) \neq y\right) - I\left(\mathrm{sign}\left(w' \cdot x'\right) \neq y\right)\right|\, d\mathscr{P}_{\mathcal{D}} \\
&= \int I\left(E_R\right)\, d\mathscr{P}_{\mathcal{D}} = \mathscr{P}_{\mathcal{D}}\{E_R\}.
\end{aligned}$$

The expectation of $\mathrm{err}_{\mathcal{D}}(w')$ with respect to the randomness of the random matrix is

$$\begin{aligned}
\mathscr{E}\{\mathrm{err}_{\mathcal{D}}(w')\} &= \int \mathrm{err}_{\mathcal{D}}(w')\, d\mathscr{P} = \int |\mathrm{err}_{\mathcal{D}}(w) - \mathrm{err}_{\mathcal{D}}(w')|\, d\mathscr{P} \\
&\leq \int \mathscr{P}_{\mathcal{D}}\{E_R\}\, d\mathscr{P} = \int\int I(E)\, d\mathscr{P}_{\mathcal{D}}\, d\mathscr{P} \\
&= \int\int I(E)\, d\mathscr{P}\, d\mathscr{P}_{\mathcal{D}} = \int \mathscr{P}\{E_x\}\, d\mathscr{P}_{\mathcal{D}} \\
&\leq \int 2e^{-\frac{\phi\left(\epsilon^2\right)^2 k}{8}} + I\left(|w \cdot x| \leq \phi\left(\epsilon^2\right)\right)\, d\mathscr{P}_{\mathcal{D}} \\
&\leq\ 2e^{-\frac{\phi\left(\epsilon^2\right)^2 k}{8}} + \epsilon^2,
\end{aligned}$$

where the first equality of the third line is obtained by Fubini's Theorem and the last inequality but one is because of Lemma 12. Therefore, by Markov's Inequality, we have

$$\mathscr{P}\{\mathrm{err}_{\mathcal{D}}(w') > \epsilon\} \leq \frac{\mathscr{E}\{\mathrm{err}_{\mathcal{D}}(w')\}}{\epsilon} \leq \frac{2}{\epsilon}e^{-\frac{\phi\left(\epsilon^2\right)^2 k}{8}} + \epsilon.$$

By solving $\frac{2}{\epsilon}e^{-\frac{\phi\left(\epsilon^2\right)^2 k}{8}} + \epsilon \leq \delta$ for k, the proposition follows.

We also have the following result. The proof is very similar with that of Proposition 13. The key idea is using Fubini's Theorem to change the integration orders. We omit the details here.

Proposition 14. *Let $w \in \mathbb{R}^n$ be a halfspace concept. $\|w\| = 1$. $\mathcal{D}$ is a distribution over $\mathbb{R}^n$. Suppose $(w, \mathcal{D})$ is regular. x is a random sample with distribution $\mathcal{D}$. $R \sim RP(k,n)$. Let $w', x' \in \mathbb{R}^k$ be the projection of w, x under R, respectively. Then for any $0 < \epsilon < 1/2, \delta > \epsilon$, if $k \geq \frac{32}{\phi^2}\log\frac{2}{\epsilon(\delta-\epsilon)}$,*

$$\mathscr{P}\{\mathscr{P}_{\mathcal{D}}\{|w' \cdot x'| \leq \phi/2\} > \epsilon\} < \delta \tag{8}$$

holds where $\phi = \phi\left(\epsilon^2\right)$.

We also need the following error bound of the base learning algorithm due to Freund and Schapire [5].

Theorem 15. *Let S be a set of m samples with $\|x_i\| \leq r$. Let (x_{m+1}, y_{m+1}) be a test sample. For $h \in \mathbb{R}^n$, $\|h\| = 1$ and $\gamma > 0$, let*

$$D_{h,\gamma} = \sqrt{\sum_{i=1}^{m+1}\xi_i^2} = \sqrt{\sum_{i=1}^{m+1}\max\{0, \gamma - y_i(h \cdot x_i)\}^2}.$$

Then the probability (over the choice of all $m+1$ samples) that the voted-perceptron algorithm with $T = 1$ does not predict y_{m+1} on the test sample x_{m+1} is at most

$$\mathscr{E}_{\mathcal{D}}\left\{\frac{2}{m+1}\inf_{\|h\|=1;\gamma>0}\left(\frac{r + D_{h,\gamma}}{\gamma}\right)^2\right\},$$

where $\mathscr{E}_{\mathcal{D}}\{\cdot\}$ is the expectation over the $m+1$ samples.

Now we are in the position of presenting the complete proof of Theorem 10.
Proof of Theorem 10. Let (x'_{m+1}, y_{m+1}) be the projection of the test sample (x_{m+1}, y_{m+1}). Take $k = \max\left\{\frac{32}{\phi^2}\log\frac{2}{\epsilon(\delta/4-\epsilon)}, \frac{8}{\epsilon^2}\log\frac{2(m+2)}{\delta}\right\}$, where $\phi = \phi\left(\epsilon^2\right)$. By Proposition 13 and Proposition 14,

$$\mathscr{P}\{\mathrm{err}_{\mathcal{D}}(w') \geq \epsilon\} \leq \delta/4,$$
$$\mathscr{P}\{\mathscr{P}_{\mathcal{D}}\{|w' \cdot x'| \geq \phi/2\} \geq \epsilon\} \leq \delta/4$$

both holds. Moreover, by Theorem 1, $\|w\| = 1$, and $\|x_i\| = 1$, $1 \le i \le m+1$, we have $\mathscr{P}\{\|w'\| \ge 1+\epsilon\} \le \frac{\delta}{2(m+2)}$ and $\mathscr{P}\{\|x_i'\| \ge 1+\epsilon\} \le \frac{\delta}{2(m+2)}$, for $1 \le i \le m+1$. Define the "good" set of random matrix as:

$$G = \{R : \mathrm{err}_{\mathcal{D}}(w') \le \epsilon\} \cap \{R : \mathscr{P}_{\mathcal{D}}\{|w' \cdot x'| \le \phi/2\} \le \epsilon\}$$
$$\cap \{R : \|w'\| \le 1+\epsilon\} \cap \left\{R : \max_{1 \le i \le m+1} \|x_i'\| \le 1+\epsilon\right\}.$$

Hence $\mathscr{P}\{G\} \ge 1-\delta$. Fix a $R \in G$ in the following. S' is the projection of S under R. Denote $A' = S' \cup \{(x'_{m+1}, y_{m+1})\}$. The empirical error of w' on A' is

$$\mathrm{err}_{A'}(w') = \frac{1}{m+1} \sum_{i=1}^{m+1} I\left(\mathrm{sign}\left(w' \cdot x_i'\right) \ne y_i\right).$$

By the Chernoff bound, we have

$$\mathscr{P}_{\mathcal{D}}\{|\mathrm{err}_{A'}(w') - \mathrm{err}_{\mathcal{D}}(w')| \ge \epsilon\} \le 2e^{-2(m+1)\epsilon^2} < \frac{\delta}{4}.$$

Since $R \in G$, $\mathrm{err}_{\mathcal{D}}(w') \le \epsilon$. Therefore, we obtain

$$\mathscr{P}_{\mathcal{D}}\{(m+1)\mathrm{err}_{A'}(w') \le 2\epsilon(m+1)\} \ge 1 - \frac{\delta}{4},$$

provided $m+1 \ge \frac{\log(8/\delta)}{2\epsilon^2}$. Similarly, we can also bound the number of samples correctly classified by w' but with margin less than $\phi/2$, i.e.,

$$\mathscr{P}_{\mathcal{D}}\left\{\sum_{i=1}^{m+1} I\left(0 < y_i\left(w' \cdot x_i'\right) \le \phi/2\right) \le 2\epsilon(m+1)\right\}$$
$$\ge \mathscr{P}_{\mathcal{D}}\left\{\sum_{i=1}^{m+1} I\left(|w' \cdot x_i'| \le \phi/2\right) \le 2\epsilon(m+1)\right\} \ge 1 - \frac{\delta}{4}.$$

Setting $\gamma = \phi/2$, $D^2_{h,\gamma}$ can be upper bounded by $(m+1)(1+\epsilon+\phi/2)^2$. Hence

$$\inf_{\|h\|=1,\gamma} \left(\frac{r + D_{h,\gamma}}{\gamma}\right)^2 \le \frac{1+\epsilon+2(m+1)\left(1+\epsilon+\phi/2\right)^2}{\phi^2/4},$$

if $m \ge 4$. What's more, D has a tighter bound. With probability at least $1-\delta/2$ over the randomness of all $m+1$ samples, we have

$$D^2_{w',\phi/2} = \sum_{i=1}^{m+1} \max\left\{0, \phi/2 - y_i\left(w' \cdot x_i'\right)\right\}^2$$
$$\le (1+2\epsilon+\epsilon^2+\phi/2)^2 \sum_{i=1}^{m+1} I\left(\mathrm{sign}\left(w' \cdot x_i'\right) \ne y_i\right)$$
$$+ \phi^2/4 \sum_{i=1}^{m+1} I\left(0 < y_i\left(w' \cdot x_i'\right) \le \phi/2\right)$$
$$\le 2\epsilon(m+1)\left(4 + 2\phi + \phi^2/2\right).$$

Note that $D_{w'/\|w'\|,\phi/(2\|w'\|)} = D_{w',\phi/2}/\left\|w'\right\|$. Therefore, if $m \geq 2/\epsilon$, with probability at least $1-\delta/2$,

$$\inf_{\|h\|=1,\gamma}\left(\frac{r+D_{h,\gamma}}{\gamma}\right)^2 \leq \left(\frac{(1+\epsilon)^2+D_{w',\phi/2}}{\phi/2}\right)^2 \leq \frac{(1+\epsilon)^4+2D^2_{w',\phi/2}}{\phi^2/4}$$
$$\leq \frac{(1+\epsilon)^4+4\epsilon(m+1)(4+2\phi+\phi^2/2)}{\phi^2/4},$$

Hence, we can finally bound the error rate

$$\mathrm{err}_{\mathcal{D}}(f) \leq \mathscr{E}_{\mathcal{D}}\left\{\frac{2}{m+1}\inf_{\|h\|=1;\gamma>0}\left(\frac{r+D_{h,\gamma}}{\gamma}\right)^2\right\}$$
$$\leq \left(1-\frac{\delta}{2}\right)\frac{(1+\epsilon)^4+4\epsilon(m+1)(4+2\phi+\phi^2/2)}{(m+1)\phi^2/8}$$
$$+\frac{\delta}{2}\frac{1+\epsilon+2(m+1)\left(1+\epsilon+\phi/2\right)^2}{(m+1)\phi^2/8}$$
$$\leq \frac{16}{m+1}\frac{(1+\epsilon)^2+\delta}{\phi^2}+\frac{40\epsilon+18\delta}{\phi^2}.$$

5 Linearly Non-separable Case

Our analysis can be further extended to the case when data are linearly non-separable in the original space. We would like the compressedly learned classifier to be not much worse than the best linear classifier in the original space. We need to generalize the definition of regularity to a general linear classifier. Here, we only consider linear classifiers passing through the origin.

Definition 16. *A linear classifier h with a distribution $\mathcal{D}$ in $\mathbb{R}^n$ is regular, if*

$$\mathscr{P}_{\mathcal{D}}\left\{x : |h\cdot x| = 0\right\} = 0. \tag{9}$$

The definition is the same as that of a regular concept. However, here $\mathrm{err}_{\mathcal{D}}(h)$ is generally nonzero. Let $\hat{h} = \arg\min_{h\in\mathbb{R}^n}\mathrm{err}_{\mathcal{D}}(h)$ with η the minimal error rate. The following result bounds the generalization error of the classifier $f(x)$ learned on compressed data in terms of η.

Theorem 17. *$\mathcal{D}$ is a distribution in $\mathbb{R}^n$. Let $\hat{h}\in\mathbb{R}^n$ be the linear classifier with the minimal error rate η under $\mathcal{D}$. $\left\|\hat{h}\right\| = 1$. Suppose $(\hat{h},\mathcal{D})$ is regular. $R \sim RP(k,n)$. S is the training set of size m. Let S' be the projection of S under R. For any given $\epsilon > 0$ and $\delta > 8\epsilon$, with probability at least $1-\delta$, the generalization error of the classifier $f(x)$ output by the voted-perceptron algorithm based on S' satisfies,*

$$err_{\mathcal{D}}(f) \leq \frac{1}{\phi^2}\left(\frac{4(1+\epsilon)^2+2\delta}{m+1}+10(\epsilon+\eta)+8\delta\right),$$

if $k = \max\left\{ \frac{8}{\phi(\epsilon^2)^2} \log \frac{2}{\epsilon(\delta/8-\epsilon)}, \frac{32}{\phi^2} \log \frac{2}{\epsilon(\delta/8-\epsilon)}, \quad \frac{8}{\epsilon^2} \log \frac{2(m+1)}{\delta} \right\}$,
where $\phi = \phi\left((\epsilon+\eta)^2\right)$.

This result shows that if the margin distribution places little mass on small margins, one can learn a linear classifier with compressed data which would not be much too worse than the best linear classifier one can learned with original data. When m is sufficiently large, the upper bound approximates $\frac{c(\epsilon+\eta+\delta)}{\phi^2}$, and the convergence rate is $1/m$.

The proof is similar with that of Theorem 10. We omit it here and provide it in the supplementary material.

6 Experiments

In this section, we provide experimental results on synthesize data. The purpose of these experiments is to test how different distributions affect the accuracy of a classifier trained on compressed data and whether these effects are consistent with our analysis. We first introduce the experimental setups, and then provide the detailed experimental results.

6.1 Experimental Setup

For each data set used in our experiment, we randomly choose 90% of the samples as the training set, and the rest is used as the testing set. We then train two classifiers using the base algorithm on the original training set and compressed training set respectively, where the compressed training set is obtained by projecting the original training set to a k-dimensional space with a random matrix. Finally the classification accuracy is evaluated on the original and compressed testing set with the originally and compressedly trained classifier. To reduce the training bias, we repeat the above procedure for N rounds and report the averaged results. In this way, we totally trained $M \times N$ classifiers on compressed data, and the averaged accuracy of compressedly trained classifiers are also reported to compare with that of the classifiers trained on original data. We choose several different values of dimension k and plot the test accuracy curve as a function of the dimension. In the experiments, we set $M = 50$, $N = 5$ and $T = 10$ in the base algorithm.

6.2 Numerical Experiments

We synthesize data to test two aspects of our analysis: the effect of failing to satisfy Assumption 9 and the effect of different values of robust factors, on the compressedly learned classifiers.

First, we show that when Assumption 9 does not hold, the compressedly learned classifier could be significantly worse than the classifier learned on the original data. From Example 8, it can be seen that the robust factor of uniform distribution on the unit sphere in $\mathbb{R}^n$ fails to satisfy the assumption. We test this type of

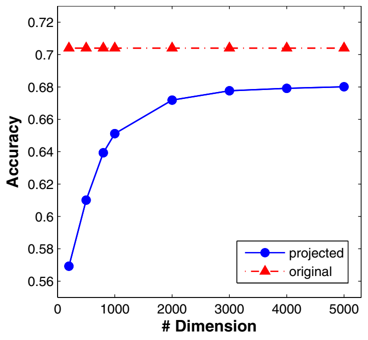


Fig. 1. Classification accuracy result on the uniform distribution over the unit n-sphere

distribution with $n = 1000$. The toy data set consists of 1000 i.i.d. samples generated from the distribution. $w = [1, 0, \cdots, 0]^T$ is the halfspace concept to learn. The averaged classification accuracy result is shown in Figure 1, which shows that the compressedly learned classifier is much worse than the classifier trained on original data, with a non-negligible error gap between the two classifiers even when the dimension k is greatly larger than the original dimension n.

Second, we test the effect of the robust factor on the generalization error of a learned classifier. Specifically, we want to show that if the robust factor of a distribution takes a larger value, the learned classifier will be more accurate. We choose $w = [1, 0, \cdots, 0]^T$ as the common halfspace concept and obtain a sequence of distributions with different robust factors near $\epsilon = 0$ by modifying a 1000-dimensional uniform distribution on the unit sphere as follows. When a sample x is generated from the uniform distribution, we compute the value $v(x) = |w \cdot x|$. If $v(x)$ is less than a threshold (0.01 in our experiments) we reject this sample with probability $1 - p$. Otherwise we simply accept it. The robust factor of the obtained distribution has a larger value near $\epsilon = 0$ than that of the

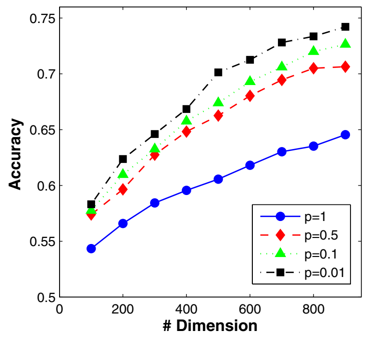


Fig. 2. Accuracy curves for four distributions with different robust factors. Small values of p give relatively large values of robust factor near $\epsilon = 0$.

uniform distribution. A smaller p gives a larger value of the robust factor near $\epsilon = 0$. We test four different values of p: 1, 0.5, 0.1, 0.01. From each distribution, we generate 1000 samples. The averaged classification accuracy result is shown in Figure 2. For a common k, a distribution with a larger robust factor near the origin results in a more accurate classifier. This results approve our analysis using robust factor to bound the error rate of a learned classifier.

7 Conclusion

In this paper, we study the problem of learning regular halfspace concepts with compressed data. We notice that the hardness of compressed learning is captured by the margin distribution. Therefore, we define the robust factor to characterize the margin distribution of a regular concept. We show that the generalization error of a compressedly learned classifier is tightly bounded in terms of the robust factor. Our analysis is also extended to the linearly non-separable case. Both theoretical and experimental results are provided to show that under certain conditions, learning halfspace concepts accurately with compressed data is possible.

References

[1] Arriaga, R.I., Vempala, S.: An algorithmic theory of learning: Robust concepts and random projection. In: FOCS 1999: Proc. of the 40th Foundations of Computer Science (1999)

[2] Calderbank, R., Jafarpour, S., Schapire, R.: Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain. Technical report, Princeton University (2009)

[3] Dasgupta, S., Gupta, A.: An elementary proof of a theorem of johnson and lindenstrauss. Random Structures and Algorithms 22(1), 60–65 (2003)

[4] Freund, Y., Dasgupta, S., Kabra, M., Verma, N.: Learning the structure of manifolds using random projections. In: Advances in Neural Information Processing Systems, vol. 20, pp. 473–480. MIT Press, Cambridge (2008)

[5] Freund, Y., Schapire, R.E.: Large margin classification using the perceptron algorithm. Mach. Learn. 37(3), 277–296 (1999)

[6] Garg, A., Har-Peled, S., Roth, D.: On generalization bounds, projection profile, and margin distribution. In: ICML 2002: Proceedings of the Nineteenth International Conference on Machine Learning, pp. 171–178 (2002)

[7] Hegde, C., Wakin, M., Baraniuk, R.: Random projections for manifold learning. In: Advances in Neural Information Processing Systems, vol. 20, pp. 641–648. MIT Press, Cambridge (2008)

[8] Johnson, W., Lindenstrauss, J.: Extensions of lipschitz maps into a hilbert space. Contemp. Math. 26, 189–206 (1984)

[9] Liu, K., Ryan, J.: Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. IEEE Trans. on Knowl. and Data Eng. 18(1), 92–106 (2006); Senior Member-Kargupta, Hillol

[10] Maillard, O., Munos, R.: Compressed least-squares regression. In: Advances in Neural Information Processing Systems, vol. 21. MIT Press, Cambridge (2009)

[11] Tsybakov, A.B.: Optimal aggregation of classifiers in statistical learning. Ann. Statist. 32, 135–166 (2004)
[12] Wang, F., Li, P.: Efficient non-negative matrix factorization with random projections. In: The 10th SIAM International Conference on Data Mining, pp. 281–292 (2010)
[13] Zhou, S., Lafferty, J., Wasserman, L.: Compressed regression. In: Advances in Neural Information Processing Systems, vol. 20, pp. 1713–1720. MIT Press, Cambridge (2008)

A Lower Bound for Learning Distributions Generated by Probabilistic Automata

Borja Balle, Jorge Castro, and Ricard Gavaldà

Universitat Politècnica de Catalunya, Barcelona
{bballe,castro,gavalda}@lsi.upc.edu

Abstract. Known algorithms for learning PDFA can only be shown to run in time polynomial in the so-called distinguishability μ of the target machine, besides the number of states and the usual accuracy and confidence parameters. We show that the dependence on μ is necessary for every algorithm whose structure resembles existing ones. As a technical tool, a new variant of Statistical Queries termed L_∞-queries is defined. We show how these queries can be simulated from samples and observe that known PAC algorithms for learning PDFA can be rewritten to access its target using L_∞-queries and standard Statistical Queries. Finally, we show a lower bound: every algorithm to learn PDFA using queries with a resonable tolerance needs a number of queries larger than $(1/\mu)^c$ for every $c < 1$.

1 Introduction

Probabilistic finite automata (PFA) are important as modeling formalisms as well as computation models. They are closely related to Hidden Markov Models (HMM's) in their ability to represent distributions on finite alphabets and also to POMDP's; see e.g. [8, 17, 18] for background.

One of the main associated problems is that of approximating the distribution generated by an unknown probabilistic automaton from samples. The problem is relatively simple if the structure of the automaton is somehow known and only transition probabilities have to be estimated, and much harder and poorly-solved in practice if the transition graph is unknown. Probabilistic Deterministic Finite Automata (PDFA) — in which the underlying automaton is deterministic but transitions still have probabilities — have been often considered as a restriction worth studying, even though they cannot generate all distributions generated by PFA [8].

The grammatical inference community has produced a substantial number of methods for learning (distributions generated by) PFA or PDFA, most of them using so-called "state split-merge" or "evidence-driven" strategies; see the references in [6, 17, 18, 7]. Many of these methods are only proved valid empirically, but some have proofs of learning in the limit.

The problem has also been intensely studied in variants of the PAC model adapted to distribution learning. Abe and Warmuth showed in [1] that hardness is not information-theoretic: one can learn (distributions generated by) PFA

© Springer-Verlag Berlin Heidelberg 2010

with samples of size polynomial in alphabet size, number of states in the target machine, and inverses of the accuracy and confidence parameters (ϵ and δ); but also that the problem is computationally intractable for large alphabet sizes, unless $\mathsf{RP} = \mathsf{NP}$. Kearns et al. [13] showed that learning PDFA even over 2-letter alphabets is computationally as hard as solving the *noisy parity learning problem*, of interest in coding theory and for which only super-polynomial time algorithms are known.

It was later observed that polynomial-time learnability is feasible if one allows polynomiality not only in the number of states but also in other measures of the target automaton complexity. Specifically, Ron et al. [16] showed that acyclic PDFA can be learned w.r.t the Kullback-Leibler divergence in time polynomial in alphabet size, $1/\epsilon$, $1/\delta$, number of target states, and $1/\mu$, where μ denotes the *distinguishability* of the target automaton, to be defined in Sect. 2. Clark and Thollard extended the result to general PDFA by considering also as a parameter the expected length of the strings generated by the automata [6]. Their algorithm, a state merge-split method, was in turn extended or refined in subsequent work [10, 15, 9, 4]. Furthermore, in [11] a PAC algorithm for learning PFA was given, similar in spirit to [7], whose running time is polynomial in the inverse of a condition parameter, intuitively an analog of μ for PFA.

Here we consider the dependence on the distinguishability parameter μ of known algorithms. We know that the sample complexity and running time of the Clark-Thollard and related algorithms is polynomially bounded on $1/\mu$ (as well as other parameters), but it is conceivable that one could also prove a polynomial bound in another parameter, much smaller but yet unidentified. We rule out this possibility for a large class of learning algorithms, intuitively those that proceed by applying statistical tests to subsets of the sample to distinguish distributions generated at different states of the target automaton. To this end, we define a variant of Kearns' statistical queries [12], called L_∞-queries. We observe that known algorithms for learning PDFA, such as Clark-Thollard and our variant [4], can be rewritten accessing the target distribution only through L_∞-queries (to infer the structure) plus standard statistical queries (to approximate transition probabilities). We then show that any algorithm that learns the class of PDFA with a given distinguishability μ from L_∞-queries and statistical queries with reasonably bounded tolerance will require more than $(1/\mu)^c$ queries for every $c < 1$. Our result thus indicates that, if PDFA learning algorithms of complexity substantially smaller than $1/\mu$ do exist, they must use their input sample quite differently from known algorithms.

While we introduce our L_∞-queries as a technical concept to formulate a lower bound, we believe they may deserve further study. Interestingly, the hard targets that force our lower bound are essentially the noiseless parity functions, which are learnable in time polynomial in the number of variables, but by our result not from L_∞-queries. Recalling that noisy parity functions seem computationally hard to learn, this suggests a connection to investigate between our L_∞-queries and noisy distribution learning, as there is one between SQ and noisy concept learning. Additionally, we give several indications (not rigorous proofs) that L_∞-queries cannot be efficiently simulated by standard SQ.

2 Preliminaries

We consider several measures of divergence between distributions. Let D_1 and D_2 be probability distributions on a discrete set X. The Kullback–Leibler (KL) divergence is defined as

$$\mathsf{KL}(D_1 \| D_2) = \sum_{x \in X} D_1(x) \log \frac{D_1(x)}{D_2(x)}, \tag{1}$$

where the logarithm is taken to base 2. The KL is sometimes called relative entropy. The supremum distance is $\mathrm{L}_\infty(D_1, D_2) = \max_{x \in X} |D_1(x) - D_2(x)|$, and the total variation distance is $L_1(D_1, D_2) = \sum_{x \in X} |D_1(x) - D_2(x)|$.

An algorithm learns a class of distributions $\mathcal{D}$ over some set X if for any $D \in \mathcal{D}$ and $\epsilon > 0$ it is given access to D through some oracle and outputs a hypothesis $\hat{D}$ that is ϵ-close to D w.r.t. the KL divergence, that is, $\mathsf{KL}(D \| \hat{D}) < \epsilon$.

A PDFA A is a tuple $\langle Q, \Sigma, \tau, \gamma, \xi, q_0 \rangle$ where Q is a finite set of states, Σ is the alphabet, $\tau : Q \times \Sigma \longrightarrow Q$ is the transition function, $\gamma : Q \times (\Sigma \cup \{\xi\}) \longrightarrow [0,1]$ defines the probability of emitting each symbol from each state ($\gamma(q, \sigma) = 0$ when $\sigma \in \Sigma$ and $\tau(q, \sigma)$ is not defined), ξ is a special symbol not in Σ reserved to mark the end of a string, and $q_0 \in Q$ is the initial state. It is required that $\sum_{\sigma \in \Sigma \cup \{\xi\}} \gamma(q, \sigma) = 1$ for every state q. Transition function τ is extended to $Q \times \Sigma^*$ in the usual way. Also, the probability of generating a given string $x\xi$ from state q can be calculated recursively as follows: if x is the empty word λ the probability is $\gamma(q, \xi)$, otherwise x is a string $\sigma_0 \sigma_1 \ldots \sigma_k$ with $k \geq 0$ and $\gamma(q, \sigma_0 \sigma_1 \ldots \sigma_k \xi) = \gamma(q, \sigma_0) \gamma(\tau(q, \sigma_0), \sigma_1 \ldots \sigma_k \xi)$. Assuming every state of A has non-zero probability of generating some string, one can define for each state q a probability distribution D_q on Σ^*: for each x, probability $D_q(x)$ is $\gamma(q, x\xi)$. The one corresponding to the initial state D_{q_0} is called the distribution defined by A.

Definition 1. *We say distributions D_1 and D_2 are μ-*distinguishable *when $\mu \leq \mathrm{L}_\infty(D_1, D_2)$. A PDFA A is μ-distinguishable when for each pair of states q_1 and q_2 their corresponding distributions D_{q_1} and D_{q_2} are μ-distinguishable.*

Given a multiset S of strings from Σ^* we denote by $S(x)$ the multiplicity of x in S, write $|S| = \sum_{x \in \Sigma^*} S(x)$. To each multiset S corresponds an empirical distribution $\hat{S}$ defined in the usual way, $\hat{S}(x) = S(x)/|S|$.

A parity on n variables is a function $h : \{0,1\}^n \to \{0,1\}$ of the form $h(x_1, \ldots, x_n) = \sum_i a_i x_i \mod 2$, for some $(a_1, \ldots, a_n) \in \{0,1\}^n$.

The following is a simple consequence of Chebyshev-Cantelli inequality that will be used when proving the lower bound.

Lemma 1. *Let X be a random variable with expectation μ and variance σ^2. If $t > 2|\mu|$ then:*

$$\mathbb{P}\left[|X| \geq t\right] \leq 2\frac{\sigma^2}{t(t-2|\mu|)}. \tag{2}$$

3 L_∞-Queries

In this section we present a new kind of query, the L_∞-query, which we describe as a call to an oracle DIFF_∞. Roughly speaking, these queries can be used whenever the learning task is to approximate a probability distribution whose support is contained in a free monoid Σ^*. This query is an abstraction of a pattern of access to distributions appearing in algorithms that learn (distributions generated by) PDFA [3, 16, 6, 10, 15, 9, 4]. At some point, all algorithms described in these papers use samples from suffix distributions to test for state-distinctness w.r.t. the supremum distance.

Let D be a distribution over Σ^*, where Σ is a finite alphabet. If $A \subseteq \Sigma^*$ is prefix-free, we denote by D^A the conditional distribution under D of having a prefix in A. That is, for every $y \in \Sigma^*$ we have

$$D^A(y) = \frac{D(Ay)}{D(A\Sigma^*)} = \frac{\sum_{x\in A} D(xy)}{\sum_{x\in A} D(x\Sigma^*)}, \tag{3}$$

where $D(x\Sigma^*)$ is the probability under D of having x as a prefix. The oracle $\mathrm{DIFF}_\infty(D)$ answers queries of the form (A, B, α, β), where $A, B \subseteq \Sigma^*$ are (encodings of) disjoint and prefix-free sets, and $\alpha, \beta \in (0,1)$ are real numbers. Let μ denote the supremum distance between distributions D^A and D^B; that is, $\mu = \mathrm{L}_\infty(D^A, D^B)$. Then oracle $\mathrm{DIFF}_\infty(D)$ must answer a query (A, B, α, β) according to the following rules:

1. If either $D(A\Sigma^*) < \beta$ or $D(B\Sigma^*) < \beta$, it answers "?".
2. If both $D(A\Sigma^*) > 3\beta$ and $D(B\Sigma^*) > 3\beta$, it answers some number $\hat{\mu}$ such that $|\mu - \hat{\mu}| < \alpha$.
3. Otherwise, the oracle may either answer "?" or give an α-good approximation $\hat{\mu}$ of μ, arbitrarily.

To be precise, the algorithm asking a query will provide A and B in the form of oracles deciding the membership problems for $A\Sigma^*$ and $B\Sigma^*$.

Similarly to oracles answering statistical queries [12], the price an algorithm has to pay for a call to $\mathrm{DIFF}_\infty(D)$ depends on the parameters of the query. As will be seen in the next section, a call to $\mathrm{DIFF}_\infty(D)$ with a query (A, B, α, β) can be simulated with $\tilde{O}(\alpha^{-2}\beta^{-2})$ samples from D. Accordingly, we make the following definition.

Definition 2. *An algorithm for learning a class of distributions $\mathcal{D}$ over Σ^* using L_∞-queries will be called* sample efficient *if there exists polynomials p, q, r such that for each $D \in \mathcal{D}$ the algorithm makes at most $r(1/\epsilon, |D|)$ queries with $\alpha > 1/p(1/\epsilon, |D|)$ and $\beta > 1/q(1/\epsilon, |D|)$ for each query, where $|D|$ is some measure of complexity, and it outputs a hypothesis $\hat{D}$ which is ϵ-close to D.*

Remark 1 (The role of β). An algorithm asking an L_∞-query does not know a priori the probability under D of having a prefix in A. It could happen that the region $A\Sigma^*$ had very low probability, and this might indicate that a good approximation of D in this region is not necessary in order to obtain a good estimate of D. Furthermore, getting this approximation would require a large number of examples. Thus, β allows a query to fail when at least one of the regions being compared has low probability. This prevents a learner from being penalized for asking queries whose answer might be irrelevant after all.

Remark 2 (Representation of A and B). From now on we will concentrate on the information-theoretic aspects of L_∞-queries. Hence, only the number of samples needed to simulate queries and the number of such queries needed to learn a specific class of distributions will be taken into account. We are not concerned with how A and B are encoded or how membership to them is tested from the code: the representation could be a finite automaton, a Turing machine, a hash table, a logical formula, etc.

3.1 Relation with Statistical Queries

Although L_∞-queries compute a value which is statistical in nature, it is not clear whether they can be simulated by statistical queries (or the other way round). Indeed, we provide some evidence suggesting that they cannot, at least efficiently.

To begin with, one has to say what would be the equivalent of statistical queries when the target of the learning process is a distribution instead of a concept. Recall that in the usual statistical query model one asks queries of the form (χ, α) where $\chi : X \times \{0,1\} \to \{0,1\}$ is a predicate and $0 < \alpha < 1$ is some tolerance. If D is a distribution over X and $f : X \to \{0,1\}$ is a concept, a query (χ, α) to the oracle $\mathrm{SQ}(f, D)$ answers with an α-good approximation $\hat{p}_\chi$ of $p_\chi = \mathbb{P}_x[\chi(x, f(x)) = 1]$, where x is drawn according to D. Kearns interprets this oracle as a proxy to the usual PAC example oracle $\mathrm{EX}(f, D)$ abstracting the fact that learners usually use samples only to obtain statistical properties about concept f under distribution D. Note that oracle $\mathrm{SQ}(f, D)$ can be simulated online using $\mathrm{EX}(f, D)$: seeing one example $(x, f(x))$ at a time, check whether $\chi(x, f(x)) = 1$ and discard it, only keeping track of the number of examples seen so far and how many of them satisfied the predicate. An obvious adaptation of statistical queries for learning distributions over X is to do the same forgetting about labels. Then $\chi : X \to \{0,1\}$ is again a predicate, and the oracle $\mathrm{SQ}(D)$ returns an α-good approximation of $\mathbb{P}_x[\chi(x) = 1]$. Since χ is the characteristic function of some subset of X, learners can ask the oracle for an approximation to the probability of any event. We assume that this is the natural translation of statistical queries for distribution learning.

As in the case of concept learning, statistical queries for distributions can be simulated online with essentially constant memory: just count elements in the sample satisfying the predicate. Now, this does not seem possible for L_∞-queries, where in order to compute the supremum distance between two empirical distributions one needs to collect sets of examples, estimate the probabilities of elements in the sets and compare these probabilities to see which one defines the maximum difference. This indicates that a single statistical query can not be used to simulate a L_∞-query. However, this does not preclude the possibility that L_∞-queries can be simulated with a larger number of statistical queries.

An obvious such simulation is: given access to oracles $\mathrm{SQ}(D^A)$ and $\mathrm{SQ}(D^B)$, obtain approximations of $D^A(x)$ and $D^B(x)$ for each x in the support and then return the largest difference $|D^A(x) - D^B(x)|$. This is not feasible when the support is infinite, although for most reasonable classes of distributions with infinite support the string defining the supremum distance cannot be very long. But even for statistical queries that return exact probabilities, this approach amounts to finding a string where the supremum distance between two distributions is attained. A problem that was shown to be NP-hard for the case of distributions generated by probabilistic automata in [14]. On the other hand, when one is not asking for particular probabilities, but samples from the distributions are available instead, the empirical supremum distance is usually a good approximation of the actual distance provided enough examples are available. This is the topic of next section.

We currently have a candidate class of distributions which we believe can rule out the possibility of simulating L_∞-queries using a polynomial number of statistical queries.

3.2 Simulation

In this section we show how to simulate calls to $\mathrm{DIFF}_\infty(D)$ using examples from D provided by the classical PAC example oracle $\mathrm{EX}(D)$. Our fist lemma says that the supremum distance between two arbitrary distributions over Σ^* can be approximated with a moderate number of examples provided a similar number of examples from both distributions is available.

Let D^A and D^B be two distributions over Σ^*. Let S^A be a sample of size n_A from D^A and S^B a sample of size n_B from D^B. Define $\mu = \mathrm{L}_\infty(D^A, D^B)$ and its empirical estimation $\hat{\mu} = \mathrm{L}_\infty(\hat{S}^A, \hat{S}^B)$. Fix some error probability $0 < \delta < 1$, an approximation factor $0 < \alpha < 1$, and an arbitrary constant $0 < c < 1$. Now define

$$N_1 = \frac{6}{\alpha^2 c} \ln \frac{24}{\alpha^2 c \delta}. \tag{4}$$

Lemma 2. *If $n_A, n_B \in [cN, N]$ for some $N > N_1$, then $|\hat{\mu} - \mu| \leq \alpha$ with probability at least $1 - \delta/2$.*

The proof is based on Chernoff bounds and is omitted.

Now we describe a simulation of L_∞-queries using the usual $\mathrm{EX}(D)$ oracle from the PAC model. For any distribution D, each call to $\mathrm{EX}(D)$ takes unit time and returns an example drawn according to D. As it is usual in the PAC model, the simulation will have some error probability to account, among other things, for the fact that with low probability examples provided by $\mathrm{EX}(D)$ can be unrepresentative of D.

Let D be a distribution over Σ^*. Fix some L_∞-query (A, B, α, β) and some error probability δ. Now D^A and D^B will be suffix distributions of D; that is, conditional distributions obtained when words have a prefix in A or B. Let $p_A = D(A\Sigma^*)$ (respectively, $p_B = D(B\Sigma^*)$) denote the probability that a word drawn according to D has a prefix in A (respectively, in B). As before, μ will be the supremum distance between D^A and D^B.

Given a sample S from D, a sample S^A from D^A is obtained as follows. For each word $x \in S$, check whether $x = yz$ with $y \in A$. If this is the case, add z to S^A. The multiset obtained,

$$S^A = \{z : yz \in S \text{ and } y \in A\}, \tag{5}$$

is a sample from D^A. Note that since A is prefix-free, each word in S contributes at most one word to S^A, and thus all examples in S^A are mutually independent. Similarly, a sample S^B from D^B is obtained. Let n_A and n_B denote the respective sizes of S^A and S^B.

In order to simulate a call to $\mathrm{DIFF}_\infty(D)$ with query (A, B, α, β), draw a sample S of size N from D using $\mathrm{EX}(D)$. Then, build samples S^A and S^B from S and obtain approximations $\hat{p}_A = n_A/N$ and $\hat{p}_B = n_B/N$ of p_A and p_B, respectively. If either $\hat{p}_A < 2\beta$ or $\hat{p}_B < 2\beta$, return "?". Otherwise, return $\hat{\mu} = \mathrm{L}_\infty(\hat{S}^A, \hat{S}^B)$.

The following theorem shows that $\tilde{O}(\alpha^{-2}\beta^{-2})$ samples are enough for the simulation to succeed with high probability.

Theorem 1. *For any distribution D over Σ^*, a L_∞-query (A, B, α, β) to the oracle* $\mathrm{DIFF}_\infty(D)$ *can be simulated with error probability smaller than δ using $N > N_0$ calls to the oracle* $\mathrm{EX}(D)$*, where*

$$N_0 = \max\left\{\frac{3}{\alpha^2\beta}\ln\frac{12}{\alpha^2\beta\delta}, \frac{1}{2\beta^2}\ln\frac{8}{\delta}\right\}. \tag{6}$$

Proof. It follows from Chernoff bounds that $\hat{p}_A$ and $\hat{p}_B$ will both be β-good approximations with probability at least $1 - \delta/2$ if $N > (1/2\beta^2)\ln(8/\delta)$. Thus, the simulation will answer "?" correctly with high probability. On the other side, if both $\hat{p}_A \geq 2\beta$ and $\hat{p}_B \geq 2\beta$, then by Lemma 2 with $c = 2\beta$ the estimate $\hat{\mu}$ will be α-good with probability at least $1 - \delta/2$. □

Remark 3 (Running time of the simulation). Although the number of examples required by the simulation bounds its running time from below, this number does not completely determine how long the simulation will take. In fact, the time required to check if $x \in \Sigma^*$ belongs to $A\Sigma^*$ or $B\Sigma^*$ affects the total running time. Furthermore, depending on the representation of A and B, checking whether x has a prefix in one of them may depend on its length $|x|$. Thus, if $T_A(m)$ and $T_B(m)$ represent the time needed to check if a string of length m has a prefix in A and B, respectively, the *expected* running time of the simulation using N examples is $O(N\,\mathbb{E}_x(\max\{T_A(|x|), T_B(|x|)\}))$. Note that if A and B are represented by automata, then $T_A(m), T_B(m) \leq cm$ for some constant c. In this case, the expected running time of the simulation is $O(NL)$, where $L = \mathbb{E}_x[|x|]$ is the expected length of D. This justifies the appearance of L in running time bounds for algorithms learning PDFA in the PAC model.

4 Lower Bound

In this section we prove that no sample efficient L_∞-query algorithm satisfying some restrictions can learn a certain class of distributions $\mathcal{D}_n$. Since this class is a subclass of all PDFA with $\Theta(n)$ states, it will follow that the class of distributions generated by PDFA is not learnable sample efficiently from L_∞-queries.

Let $\mathcal{P}_n$ be the set of parities on n variables. Consider the class of distributions $\mathcal{D}_n$ over $\{0,1\}^{n+1}$ where there is a distribution D_h for each parity $h \in \mathcal{P}_n$ which for any $x \in \{0,1\}^n$ and $y \in \{0,1\}$ satisfies $D_h(xy) = 2^{-n}$ if $h(x) = y$ and $D_h(xy) = 0$ otherwise. The class $\mathcal{D}_n$ contains 2^n distributions. Note that each one of these distributions can be represented by a PDFA with at most $2n+2$ states.

We will show that for n large enough, the class $\mathcal{D}_n$ can not be learned with a sample efficient L_∞-query algorithm. To do so, an adversary answering the queries asked by a learning algorithm is provided. Then it is shown that very little information about the underlying distribution can be gained with a sub-exponential number of such queries when answers are provided by the adversary. The argument is similar in nature to that used in [12] to prove that parities can not be learned in the statistical query model. Basically, we show that for each answer the number of distributions in $\mathcal{D}_n$ that are inconsistent with it is at most a sub-exponential number. Since there are an exponential number of distributions in $\mathcal{D}_n$, after a sub-exponential number of queries only a small fraction of the whole set of distributions has been ruled out. Thus, the adversary can always find a distribution which is consistent with every answer given to the algorithm but still has large error with respect to the hypothesis provided by the learner.

We present our lower bound for algorithms using L_∞-queries only. The argument for dealing with standard SQ queries, in case the algorithm uses both types, is exactly as in the lower bound proved by Kearns for concept learning parities, and we omit it for brevity. Let L be a sample efficient algorithm for learning $\mathcal{D}_n$ using L_∞-queries only. Fix ϵ to be some constant smaller than $1/9$. Now, let $p(n)$ and $q(n)$ be two functions such that for each query (A, B, α, β) asked by L the following holds: 1) $\alpha > 1/p(n)$, 2) $\beta > 1/q(n)$, 3) $p(n)$ and $q(n)$ are $2^{o(n)}$, and 4) there exist positive k_A and k_B such that $A \subseteq \{0,1\}^{k_A}$ and $B \subseteq \{0,1\}^{k_B}$. A query (A, B, α, β) satisfying 4 will be called *strict*. Restricting to strict queries is a technical condition which we believe can be removed in a more careful analysis. Nonetheless, this condition holds for the PDFA learning algorithms we are aware of when restricted to target PDFA representing parities. That is because a non-strict query in this setting means the algorithm is considering states generating words of different lengths, and this in turn means hypotheses having infinite KL with any $D \in \mathcal{D}_n$.

The following theorem states our lower bound formally. Its qualitative corollary is immediate.

Theorem 2. *Let functions $p(n)$ and $q(n)$ be $2^{o(n)}$. If $\epsilon \leq 1/9$ and n is large enough, an algorithm using strict* L_∞*-queries where $\alpha > 1/p(n)$ and $\beta > 1/q(n)$ for any query (A, B, α, β) cannot learn $\mathcal{D}_n$ with $o(2^n/\max\{p(n)^2 q(n), q(n)^2\})$ queries.*

Corollary 1. *For $\epsilon \leq 1/9$ and n large enough, the class $\mathcal{D}_n$ cannot be learned sample efficiently with* L_∞*-queries.*

Proof (of Theorem 2). Let (A, B, α, β) be a strict L_∞-query asked by L. Without loss of generality we assume that $k_A \geq k_B$. If $k_A \leq n$, for any $a \in \{0,1\}$, we define the quantity $\theta_{A,a}$ as $(-1)^a/2$ if the all zero string belongs to A and as 0 otherwise. If $k_A = n+1$, the quantity θ'_A is defined as $(-1)^a/2$ if $0 \cdots 0a \in A$ and $0 \cdots 0\bar{a} \notin A$, where $a \in \{0,1\}$ and $\bar{a}$ means negation; we let $\theta'_A = 0$ otherwise. Quantities $\theta_{B,b}$ and θ'_B are defined similarly.

The adversary is defined and analyzed in two parts. In the first part we consider the cases where it answers "?", while the situations where some $\hat{\mu}$ is answered are considered in the second part. Our analysis begins by considering the following three cases, where the adversary answers the query with "?":

1. If either $k_A, k_B > n + 1$.
2. If either $k_A \leq n$ with $|A| < 2^{k_A}\beta$ or $k_B \leq n$ with $|B| < 2^{k_B}\beta$.
3. If either $k_A = n + 1$ with $|A| < 2^{n+2}\beta - 2\theta'_A$ or $k_B = n + 1$ with $|B| < 2^{n+2}\beta - 2\theta'_B$.

Recall that an oracle answering L_∞-queries may answer "?" whenever the probability of the words with a prefix in A or B is smaller than 3β. We will only reason about A; by symmetry, the same arguments work for B. In case 1, it is obvious that $D_h(A\{0,1\}^*) = 0$ for any parity $h \in \mathcal{P}_n$ and therefore the answer "?" is consistent with all distributions in $\mathcal{D}_n$. Now, in case 2, if $k_A \leq n$ then $D_h(A\{0,1\}^*) = 2^{-k_A}|A|$ independently of h. Thus, the answer "?" is consistent with all parities if $|A| < 2^{k_A}\beta$. Lastly, for case 3 assume that $k_A = n + 1$. Now $D_h(A\{0,1\}^*) = D_h(A)$, and this probability does depend on h since it equals 2^{-n} times the number of words $xy \in A$ such that $h(x) = y$. Hence, it is not possible for the answer "?" to be consistent with all distributions, although we show that it is consistent with most of them. If parity h is chosen uniformly at random, by a routine calculation one shows that

$$\mathbb{E}_h[D_h(A)] = 2^{-n}\left(\frac{|A|}{2} + \theta'_A\right). \tag{7}$$

So, our adversary answers "?" whenever $\mathbb{E}_h[D_h(A)] < 2\beta$. The number of distributions in $\mathcal{D}_n$ inconsistent with this answer can be upper bounded using a probabilistic argument. By Chebyshev's inequality,

$$\mathbb{P}_h[D_h(A) > 3\beta] \leq \mathbb{P}_h[|D_h(A) - \mathbb{E}_h[D_h(A)]| > \beta] \leq \frac{\mathbb{V}_h[D_h(A)]}{\beta^2}. \tag{8}$$

The leftmost probability in this equation is the number of inconsistent distributions times 2^{-n}. Now, write A as the disjoint union $A = A_{01} \cup A'$, where for any $x \in \{0,1\}^n$ the words $x0$ and $x1$ belong to A_{01} if and only if $x0, x1 \in A$. This partition implies that for any parity h exactly a half of the words $xy \in A_{01}$ satisfy $h(x) = y$. It follows then that a part of $D_h(A)$ does not depend on h: $D_h(A) = 2^{-n-1}|A_{01}| + D_h(A')$. Thus only the part A' contributes to the variance of $D_h(A)$. Taking this into account, a computation with indicator variables and a standard linear algebra argument show that

$$\mathbb{V}_h[D_h(A)] = 2^{-2n}\left(\frac{|A'|}{4} - {\theta'_A}^2\right). \tag{9}$$

Applying the bounds $1/q(n) < \beta < 1$, the definition of θ'_A and recalling the assumption $|A| < 2^{n+2}\beta - 2\theta'_A$, we see that (8) and (9) imply that the number of distributions inconsistent with the answer "?" is, in this case, smaller than $q(n)^2$.

So far, we have shown that whenever the adversary answers "?", at most $q(n)^2$ distributions in $\mathcal{D}_n$ are inconsistent with this answer. Now we move ahead to the second part of the analysis. In the rest of the cases the adversary answers with some $\hat{\mu}$. In particular:

1. If $k_B < k_A < n + 1$ then $\hat{\mu} = 2^{k_A - n - 1}$.
2. If $k_B < k_A = n + 1$ then $\hat{\mu} = 1$.
3. If $k_B = k_A$ then $\hat{\mu} = 0$.

In what follows we show that, if n is large enough, the number of distributions inconsistent with the answer is, in each case, bounded by $\max\{p(n)^2 q(n), q(n)^2\}$.

Before proceeding, observe that in all these cases $k_A \leq n+1$ and for any parity h the conditional distribution D_h^A has support $\{0,1\}^{n+1-k_A}$ with the convention that $\{0,1\}^0 = \{\lambda\}$, the set with the empty string. Furthermore, if $k_A \leq n$ we can write any parity $h \in \mathcal{P}_n$ as $h = f + g$ where $f \in \mathcal{P}_{k_A}$ and $g \in \mathcal{P}_{n-k_A}$, with the convention that $\mathcal{P}_0$ contains only the constant 0. Then, for any $x = yz$ with $y \in \{0,1\}^{k_A}$ and $z \in \{0,1\}^{n-k_A}$ we have $h(x) = f(y) + g(z)$. Everything holds equally when replacing A by B.

We start now with case 1. Like before, we have $D_h(A\{0,1\}^*) = 2^{-k_A}|A| = p_A$ and $D_h(B\{0,1\}^*) = 2^{-k_B}|B| = p_B$ for any parity h. Now, given $y \in \{0,1\}^{n-k_A}$ and $z \in \{0,1\}$, by definition we can write

$$D_h^A(yz) = \frac{\sum_{x \in A} D_h(xyz)}{p_A}. \tag{10}$$

Writing $h = f + g$, define $A_f^a = \{x \in A : f(x) = a\}$ for $a \in \{0,1\}$. This yields the partition $A = A_f^0 \cup A_f^1$. The numerator in (10) can then be written as

$$\sum_{x \in A} D_h(xyz) = \sum_{x \in A_f^0} D_h(xyz) + \sum_{x \in A_f^1} D_h(xyz). \tag{11}$$

Recall that $D_h(xyz) = 2^{-n}$ if and only if $h(xy) = f(x) + g(y) = z$. Hence, if $g(y) = z$ then $D_h(xyz) = 2^{-n}$ for all $x \in A_f^0$. Similarly, $D_h(xyz) = 2^{-n}$ for all

$x \in A_f^1$ if and only if $g(y) \neq z$. Thus, the following expression for the conditional distribution D_h^A holds:

$$D_h^A(yz) = \frac{2^{-n}}{p_A} \cdot \begin{cases} |A_f^0| & \text{if } g(y) = z \ , \\ |A_f^1| & \text{if } g(y) \neq z \ . \end{cases} \tag{12}$$

Note that for any parity h both values can be attained for some choice of y and z. With the obvious modifications, these expressions hold for B too.

Now we compute the supremum distance between D_h^A and D_h^B for any $h \in \mathcal{P}_n$. Write $h = f + g = f' + g'$ where $f \in \mathcal{P}_{k_A}$, $f' \in \mathcal{P}_{k_B}$, $g \in \mathcal{P}_{n-k_A}$ and $g' \in \mathcal{P}_{n-k_B}$. Then $\mathrm{L}_\infty(D_h^A, D_h^B)$ equals

$$\max\left\{ \frac{2^{k_A-n}}{|A|} \max\left\{|A_f^0|, |A_f^1|\right\}, \frac{2^{k_B-n}}{|B|} \max\left\{|B_{f'}^0|, |B_{f'}^1|\right\} \right\} \tag{13}$$

because D_h^A and D_h^B are distributions over suffixes of different lengths. Since $\max\{|A_f^0|, |A_f^1|\} \geq |A|/2$ and $\max\{|B_{f'}^0|, |B_{f'}^1|\} \leq |B|$, we see that

$$\mathrm{L}_\infty(D_h^A, D_h^B) = \frac{2^{k_A-n}}{|A|} \max\left\{|A_f^0|, |A_f^1|\right\} . \tag{14}$$

Note this distance only depends on the first k_A bits of the parity h.

In order to count how many distributions in $\mathcal{D}_n$ are inconsistent with the answer $\hat{\mu} = 2^{k_A-n}/2$ given by the adversary we use another probabilistic argument. Assume that a parity $h \in \mathcal{P}_n$ is chosen uniformly at random and let $f \in \mathcal{P}_{k_A}$ be the parity obtained from the first k_A bits of h. Then it is easy to verify that for $a \in \{0, 1\}$ we have

$$\mathbb{E}_h[|A_f^a|] = \frac{|A|}{2} + \theta_{A,a}, \text{ and } \mathbb{V}_h[|A_f^a|] = \frac{|A|}{4} + \frac{\theta_{A,a}}{2}. \tag{15}$$

Using these computations and recalling that $\alpha \geq 1/p(n)$ and $p_A = |A|/2^{k_A} > \beta \geq 1/q(n)$, we apply Lemma 1 and get, after some calculations,

$$\mathbb{P}_h\left[\left|\frac{|A_f^a|}{|A|} - \frac{1}{2}\right| > \alpha 2^{n-k_A}\right] \leq \frac{p(n)^2 q(n)\left(2^{k_A} + 2q(n)\theta_{A,a}\right)}{2^{n+1}\left(2^n - 2p(n)q(n)|\theta_{A,a}|\right)}. \tag{16}$$

Since $k_A \leq n$, $|\theta_{A,a}| \leq 1/2$ and $\theta_{A,0} + \theta_{A,1} = 0$, a union bound yields

$$\mathbb{P}_h\left[\left|\mathrm{L}_\infty(D_h^A, D_h^B) - \frac{2^{k_A-n}}{2}\right| > \alpha\right] \leq \frac{p(n)^2 q(n)}{2^n - p(n)q(n)}. \tag{17}$$

Therefore, the number of distributions in $\mathcal{D}_n$ inconsistent with the answer given by our adversary in this case is asymptotically bounded from above by $p(n)^2 q(n)$.

Case 2 is next. Because the adversary has not answered "?" we know that $|A| \geq 2^{n+2}\beta - 2\theta'_A$ and $|B| \geq 2^{k_B}\beta$. Since $k_A = n + 1$ it follows that $D_h^A(\lambda) = 1$

if $D_h(A\{0,1\}^*) \neq 0$, otherwise we define $D_h^A(\lambda) = 0$. Hence, for any parity h the supremum distance between D_h^A and D_h^B can be written as

$$\mathrm{L}_\infty(D_h^A, D_h^B) = \max\left\{D_h^A(\lambda), \frac{2^{k_B-n}}{|B|}\max\{|B_f^0|, |B_f^1|\}\right\}, \tag{18}$$

where f corresponds to the first k_B bits of h. Note that $\mathrm{L}_\infty(D_h^A, D_h^B) \neq 1$ implies that $D_h(A\{0,1\}^*) = 0$. Now there are two possibilities. If $A_{01} \neq \varnothing$ then for any parity h we have $D_h(A\{0,1\}^*) \neq 0$ and therefore the answer $\hat{\mu} = 1$ is consistent with every parity. On the other hand, $A_{01} = \varnothing$ implies that $A = A'$ and $|A| \leq 2^n$ because for each prefix $x \in \{0,1\}^n$ at most one of $x0$ and $x1$ belongs to A. In the latter situation we have $\mathbb{P}_h[|\mathrm{L}_\infty(D_h^A, D_h^B) - 1| > \alpha] \leq \mathbb{P}_h[\mathrm{L}_\infty(D_h^A, D_h^B) \neq 1] = \mathbb{P}_h[D_h(A\{0,1\}^*) = 0]$. This last probability is bounded by

$$\mathbb{P}_h\left[|D_h(A\{0,1\}^*) - \mathbb{E}_h\left[D_h(A\{0,1\}^*)\right]| \geq \mathbb{E}_h\left[D_h(A\{0,1\}^*)\right]\right], \tag{19}$$

which in turn can be bounded using Chebyshev's inequality by

$$\frac{\mathbb{V}_h\left[D_f(A\{0,1\}^*)\right]}{\mathbb{E}_h\left[D_h(A\{0,1\}^*)\right]^2}. \tag{20}$$

Therefore, by (7) and (9) and the bounds on $|A|$, θ'_A and β, we see that the at most $q(n)^2/16$ distributions in $\mathcal{D}_n$ are inconsistent with the answer $\hat{\mu} = 1$.

Now we consider case number 3, where $k = k_A = k_B$ and the adversary responds $\hat{\mu} = 0$. Two distinct situations need to be considered: $k = n+1$ and $k \leq n$. Assume first that $k = n+1$. An argument already used in case 2 shows that if both $A_{01} \neq \varnothing$ and $B_{01} \neq \varnothing$, then for each parity h it holds that $D_h^A(\lambda) = D_h^B(\lambda) = 1$ and therefore $\mathrm{L}_\infty(D_h^A, D_h^B) = 0$ irrespective of h. In this case the answer is consistent with every distribution. If exactly one of $A_{01} = \varnothing$ and $B_{01} = \varnothing$ holds, suppose it is $A_{01} = \varnothing$ without loss of generality, then $\mathrm{L}_\infty(D_h^A, D_h^B) \neq 0$ whenever $D_h(A\{0,1\}^*) = 0$, which, by case 2, happens for at most $q(n)^2/16$ distributions in $\mathcal{D}_n$. Now, if both $A_{01} = \varnothing$ and $B_{01} = \varnothing$, it is easy to see using a union bound that $\hat{\mu} = 0$ is inconsistent with at most $q(n)^2/8$ distributions.

Assume now that $k \leq n$. Then, from the fact that $|A| = |A_f^0| + |A_f^1|$ and $|B| = |B_f^0| + |B_f^1|$, the following expression for the L_∞ distance between D_h^A and D_h^B can be deduced:

$$\mathrm{L}_\infty(D_h^A, D_h^B) = 2^{k-n}\max_{a\in\{0,1\}}\left\{\left|\frac{|A_f^a|}{|A|} - \frac{|B_f^a|}{|B|}\right|\right\} = 2^{k-n}\left|\frac{|A_f^0|}{|A|} - \frac{|B_f^0|}{|B|}\right|, \tag{21}$$

where $f \in \mathcal{P}_k$ is formed with the first k bits of h. We will show that in this case $\hat{\mu} = 0$ is a response consistent with most of the distributions in $\mathcal{D}_n$. Write $X_f = |A_f^0|/|A| - |B_f^0|/|B|$ and note that by (15) we have $\mathbb{E}_h[X_f] = \theta_A/|A| - \theta_B/|B|$, where, for simplicity, we write θ_A and θ_B for $\theta_{A,0}$ and $\theta_{B,0}$ respectively. Performing further computations one sees that

$$\mathbb{E}_h\left[X_f^2\right] = \frac{1}{4|A|} + \frac{1}{4|B|} + \frac{\theta_A}{|A|^2} + \frac{\theta_B}{|B|^2}. \tag{22}$$

Combining the last two expressions and observing that $\theta_A \theta_B = 0$, the following formula for the variance of X_f is obtained:

$$\mathbb{V}_h[X_f] = \frac{1}{4|A|} + \frac{1}{4|B|} + \frac{\theta_A}{2|A|^2} + \frac{\theta_B}{2|B|^2}. \tag{23}$$

Since $\beta > 1/q(n)$ implies $|A|, |B| > 2^k/q(n)$, plugging these bounds in previous formulas yields:

$$|\mathbb{E}_h[X_f]| \leq \frac{q(n)}{2^{k+1}}, \text{ and } \mathbb{V}_h[X_f] \leq \frac{q(n)}{2^{k+1}} + \frac{q(n)^2}{2^{2k+1}}. \tag{24}$$

Lemma 1 then yields the bound

$$\mathbb{P}_h[\mathrm{L}_\infty(D_h^A, D_h^B) > \alpha] = \mathbb{P}_h[|X_f| > \alpha 2^{n-k}] \leq \frac{p(n)^2 q(n)(1 + q(n)/2^n)}{2^{n+1} - 2p(n)q(n)}, \tag{25}$$

where we have used that $\alpha > 1/p(n)$ and $k \leq n$. From this bound, the number of distributions for which the answer is inconsistent is asymptotically $p(n)^2 q(n)/2$.

So far we have seen that, if n is large enough, for any strict L_∞-query issued by L, the answer given by the adversary is inconsistent with at most $\max\{p(n)^2 q(n), q(n)^2\}$ distributions in $\mathcal{D}_n$. Since there are 2^n distributions for any given n, after sub-exponentially many queries there will be still many different distributions in $\mathcal{D}_n$ consistent with all the answers provided to the learner.

Now, note that the relative entropy between any two distributions in $\mathcal{D}_n$ is infinite because they have different supports. Thus, for n big enough, if L outputs a hypothesis in $\mathcal{D}_n$, it will have infinite error with high probability with respect to the random choice of a target distribution in $\mathcal{D}_n$. Recalling that for each pair of distributions in $\mathcal{D}_n$ we have $\mathrm{L}_1(D_f, D_g) = 1$, we also get a lower bound for learning $\mathcal{D}_n$ using the variation distance as error measure. Now assume L outputs some distribution $\hat{D}$, not necessarily in $\mathcal{D}_n$, such that $\mathsf{KL}(D_f \| \hat{D}) \leq \epsilon$ for some $D_f \in \mathcal{D}_n$. Then it follows from Pinsker's inequality [5] that $\mathsf{KL}(D_g \| \hat{D}) \geq (1/2 \ln 2)(1 - \sqrt{2 \ln 2 \epsilon})^2$ for any other distribution D_g different from D_f. Since $\epsilon \leq 1/9$, we then have $\mathsf{KL}(D_g \| \hat{D}) > 2/9$. Therefore, if a target distribution in $\mathcal{D}_n$ is chosen at random, then L will have large error with high probability. □

4.1 A Lower Bound in Terms of Distinguishability

A lower bound on the complexity of learning the class of PDFA with a given distinguishability now follows easily using a padding argument. We ignore the dependence on ϵ in the statement.

An L_∞-query algorithm is (p,q)-bounded if, for every query (A, B, α, β) it asks, $\alpha > 1/p$ and $\beta > 1/q$, where p and q may depend on inputs of the algorithm and the complexity of the target distribution.

Corollary 2. *Let p and q be functions in $n^{O(1)} \cdot (1/\mu)^{o(1)}$. For every $c < 1$, there is no (p,q)-bounded L_∞-query algorithm that, for every n and μ, learns the class of distributions generated by PDFA with n states and distinguishability μ with $(1/\mu)^c$ queries.*

Proof. Recall the class of distributions $\mathcal{D}_k$ from the proof of Theorem 2. For every m and k, define the class of distributions $\mathcal{C}_{m,k}$ as follows: for every distribution D in $\mathcal{D}_k$, there is a distribution in $\mathcal{C}_{m,k}$ that gives probability $D(x)$ to each string of the form $0^m x$, and 0 to strings not of this form. Every distribution in $\mathcal{D}_k$ is generated by a PDFA with $2k$ states and distinguishability 2^{-k}. It follows that every distribution $\mathcal{C}_{m,k}$ is generated by a PDFA with $m + 2k$ states and distinguishability also 2^{-k}.

Now let $m = m(k)$ grow as $2^{o(k)}$. Assume for contradiction the existence of an algorithm as in the statement of the theorem. This algorithm is (p, q)-bounded with p and q that grow like $(m + 2k)^{O(1)} \cdot (1/2^{-k})^{o(1)} = 2^{o(k)}$. By an immediate reduction, the algorithm can be used to learn the classes of distributions D_k with 2^{kc} queries for some $c < 1$. But since 2^{kc} is in $o(2^{k-o(k)})$, this contradicts Theorem 2. $\square$

5 Conclusion

Let us remark that the lower bound in the previous section, as other lower bounds for learning from statistical queries, is strangely both information-theoretic and complexity-theoretic. We know, by the results in [1], that the barrier for learning PDFA is complexity-theoretic, not information-theoretic. Yet, our result says that, for algorithms that can only see their target through the lens of statistical and L_∞-queries, the problem becomes information-theoretic.

As open problems on which we are working, we shall mention possible relations between L_∞-queries and other variants of SQ proposed in the literature, and in particular those by Ben-David et al. [2] for distribution learning. Another problem is narrowing the gap between lower and upper bound: our lower bound plus the simulation we describe does not forbid the existence of algorithms that learn from $O(1/\mu)$ samples. Yet, the best bounds we can prove now for the Clark-Thollard algorithm and its variants are larger, namely $\Theta(1/\mu^2)$ at best.

Acknowledgements. This work is partially supported by the Spanish Ministry of Science and Technology contracts TIN-2008-06582-C03-01 (SESAAME) and TIN-2007-66523 (FORMALISM), by the Generalitat de Catalunya 2009-SGR-1428 (LARCA), and by the EU PASCAL2 Network of Excellence (FP7-ICT-216886). B. Balle is supported by an FPU fellowship (AP2008-02064) of the Spanish Ministry of Education.

References

[1] Abe, N., Warmuth, M.K.: On the computational complexity of approximating distributions by probabilistic automata. Mach. Learn. 9(2-3), 205–260 (1992)

[2] Ben-David, S., Lindenbaum, M.: Learning distributions by their density levels: A paradigm for learning without a teacher. J. Comput. Syst. Sci. 55(1), 171–182 (1997)

[3] Carrasco, R.C., Oncina, J.: Learning deterministic regular grammars from stochastic samples in polynomial time. RAIRO (Theoretical Informatics and Applications) 33(1), 1–20 (1999)
[4] Castro, J., Gavaldà, R.: Towards feasible PAC-learning of probabilistic deterministic finite automata. In: Clark, A., Coste, F., Miclet, L. (eds.) ICGI 2008. LNCS (LNAI), vol. 5278, pp. 163–174. Springer, Heidelberg (2008)
[5] Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, New York (2006)
[6] Clark, A., Thollard, F.: PAC-learnability of probabilistic deterministic finite state automata. Journal of Machine Learning Research (2004)
[7] Denis, F., Esposito, Y., Habrard, A.: Learning rational stochastic languages. In: Lugosi, G., Simon, H.U. (eds.) COLT 2006. LNCS (LNAI), vol. 4005, pp. 274–288. Springer, Heidelberg (2006)
[8] Dupont, P., Denis, F., Esposito, Y.: Links between probabilistic automata and hidden markov models: probability distributions, learning models and induction algorithms. Pattern Recognition 38(9), 1349–1371 (2005)
[9] Gavaldà, R., Keller, P.W., Pineau, J., Precup, D.: PAC-learning of markov models with hidden state. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 150–161. Springer, Heidelberg (2006)
[10] Guttman, O., Vishwanathan, S.V.N., Williamson, R.C.: Learnability of probabilistic automata via oracles. In: Jain, S., Simon, H.U., Tomita, E. (eds.) ALT 2005. LNCS (LNAI), vol. 3734, pp. 171–182. Springer, Heidelberg (2005)
[11] Hsu, D., Kakade, S.M., Zhang, T.: A spectral algorithm for learning hidden markov models. CoRR abs/0811.4413 (2008)
[12] Kearns, M.: Efficient noise-tolerant learning from statistical queries. J. ACM 45(6), 983–1006 (1998)
[13] Kearns, M.J., Mansour, Y., Ron, D., Rubinfeld, R., Schapire, R.E., Sellie, L.: On the learnability of discrete distributions. In: STOC, pp. 273–282 (1994)
[14] Lyngsø, R.B., Pedersen, C.N.S.: The consensus string problem and the complexity of comparing hidden markov models. J. Comput. Syst. Sci. 65(3), 545–569 (2002)
[15] Palmer, N., Goldberg, P.W.: PAC-learnability of probabilistic deterministic finite state automata in terms of variation distance. Theor. Comput. Sci. 387(1), 18–31 (2007)
[16] Ron, D., Singer, Y., Tishby, N.: On the learnability and usage of acyclic probabilistic finite automata. J. Comput. Syst. Sci. 56(2), 133–152 (1998)
[17] Vidal, E., Thollard, F., de la Higuera, C., Casacuberta, F., Carrasco, R.C.: Probabilistic finite-state machines - part I. IEEE Trans. Pattern Anal. Mach. Intell. 27(7), 1013–1025 (2005)
[18] Vidal, E., Thollard, F., de la Higuera, C., Casacuberta, F., Carrasco, R.C.: Probabilistic finite-state machines - part II. IEEE Trans. Pattern Anal. Mach. Intell. 27(7), 1026–1039 (2005)

Lower Bounds on Learning Random Structures with Statistical Queries

Dana Angluin[1,⋆], David Eisenstat[2], Leonid (Aryeh) Kontorovich[3], and Lev Reyzin[4,⋆⋆]

[1] Yale University, New Haven CT USA
dana.angluin@yale.edu
[2] Brown University, Providence RI USA
eisenstatdavid@gmail.com
[3] Ben Gurion University of the Negev, Israel
karyeh@cs.bgu.ac.il
[4] Yahoo! Research, New York NY, USA
lreyzin@yahoo-inc.com

Abstract. We show that random DNF formulas, random log-depth decision trees and random deterministic finite acceptors cannot be weakly learned with a polynomial number of statistical queries with respect to an arbitrary distribution on examples.

1 Introduction

Polynomial time learning algorithms have been given for random log-depth decision trees by Jackson and Servedio [6], random monotone DNF formulas by Jackson et al. [5] and Sellie [12] and random general DNF formulas by Sellie [12], with respect to the uniform distribution. These algorithms are based on statistical estimates of various parameters and can be implemented using statistical queries as defined by Kearns [8].

Blum et al. [2] give upper and lower bounds on the number of statistical queries required to learn concepts from a given class in terms of a distribution-dependent statistical query dimension of the class. A corollary of their characterization is that parity functions with $\log n$ relevant variables cannot be weakly learned with respect to the uniform distribution using a polynomial number of statistical queries. This implies that arbitrary log-depth decision trees and DNF formulas with $\Theta(n)$ terms are not weakly learnable with respect to the uniform distribution using a polynomial number of statistical queries, because they can represent parity functions with $\log n$ relevant variables.

The key difference between these negative results and the positive results cited above is that the choice of a structure (DNF formula or decision tree) to be learned is random. In particular, "bad structures" (capable of representing a parity problem with $\log n$ relevant variables with respect to the uniform distribution) occur with vanishing probability as n increases.

⋆ Research supported by the National Science Foundation under Grant CCF-0916389.

⋆⋆ This material is based upon work supported by the National Science Foundation under a National Science Foundation Graduate Research Fellowship and under Grant # 0937060 to the Computing Research Association for the Computing Innovation Fellowship program.

© Springer-Verlag Berlin Heidelberg 2010

1.1 Learning DFAs

A deterministic finite acceptor M over the alphabet $\{0, 1\}$ can be used to represent the set $L(M) \cap \{0, 1\}^n$ of accepted binary strings of length n. Gold [4] gave one of the earliest hardness results for learning DFAs, that finding a smallest DFA consistent with given positive and negative data is NP-hard. The PAC-reduction given by Pitt and Warmuth [10] of learning Boolean formulas to learning DFAs, combined with the negative results of Kearns and Valiant [7] for PAC-learning Boolean formulas, gives cryptographic evidence for the hardness of PAC-learning arbitrary DFAs.

Because a DFA of $2n + 1$ states can compute the parity of an arbitrary subset of a string of n bits, the results of Blum et al. [2] imply that there is no algorithm to learn arbitrary DFAs using a polynomial number of statistical queries with respect to the uniform distribution. In light of the positive results described above for random DNF formulas and random decision trees, it is natural to ask how hard it is to learn random DFAs of $O(n^c)$ states with respect to the uniform distribution over strings of length n.

Trakhtenbrot and Barzdin [13] consider the problem of learning arbitrary finite automata using experiments – in each experiment the learning algorithm selects an input string to query and receives the sequence of output symbols generated by the target machine on reading that input. For a DFA, this is equivalent to a sequence of membership queries on all prefixes of the experiment string. Trakhtenbrot and Barzdin also consider the problem of learning almost all automata under a natural distribution on DFAs.

Building on these results, Freund et al. [3] consider a model in which the target is a DFA with an arbitrary transition graph and states independently labeled as accepting or rejecting with probability $1/2$. The learner sees the outputs of a random walk in the transition graph, and must at each step predict the next output. Freund et al. show that there is an algorithm that makes a number of mistakes bounded by a polynomial in n for uniformly almost all automata of n states. In an empirical study of the effectiveness of a learning algorithm based on Trakhtenbrot and Barzdin's contraction algorithm, Lang [9] studied what fraction of all $16n^2 - 1$ binary strings of length at most $2 \log n + 3$ is needed to achieve high levels of generalization for randomly generated machines of about n states.

These results do not directly shed light on the question of how difficult it is to learn a set of binary strings of length n accepted by a random DFA of $O(n^c)$ states with respect to the uniform distribution. We are interested in this problem, which is open to the best of our knowledge.

1.2 Lower Bounds

In this paper we show that no algorithm using a polynomial number of statistical queries can learn random DNF formulas, random decision trees, or random DFAs with respect to an arbitrary distribution. Thus the random choice of a structure is not sufficient for the positive results cited above for DNF formulas and decision trees. Even if the structure is randomly chosen, it may be possible to find a "bad input distribution" that allows a hard parity problem to be embedded in that random structure. This situation is a natural one to consider in the case of a boosting algorithm attempting to learn a random structure, because successive rounds of boosting may modify an initially simple input distribution in complex ways that depend on the target concept.

Specifically, we consider the problem of learning the behavior of a random structure using statistical queries, where the distribution on examples is adversarial, and therefore may depend on the random structure to be learned. For the cases when the random structure is a DNF formula, a log-depth decision tree, or a DFA we show that with at least a constant probability, there is a distribution on the inputs that embeds a nontrivial parity computation in the random structure. In general the "bad" distribution constrains some variables to constant values and some variables to copy other variables or their negations, and is uniform on the rest. These results provide some support for the distributional assumptions made in the positive results of Jackson and Servedio [6], Jackson et al. [5] and Sellie [11, 12].

2 Preliminaries

We consider concept classes over binary strings. Let $\Sigma = \{0, 1\}$. Σ^* is the set of all binary strings and ε is the empty string. The set of all binary strings of length n, length at most n and length less than n are denoted Σ^n, $\Sigma^{\leq n}$ and $\Sigma^{<n}$. A concept class $\mathcal{C}$ is a collection $\{\mathcal{C}_n\}$ indexed by the positive integer n, where each $\mathcal{C}_n$ is a set of concepts over Σ^n (or $\Sigma^{\leq n}$) that is, a set of mappings f from Σ^n (or $\Sigma^{\leq n}$) to Σ. Concept classes are generally specified by some representation, for example, log depth decision trees over n variables. A concept class has polynomially bounded representations if there is a fixed polynomial p such that every concept f in $\mathcal{C}_n$ has a representation of length bounded by $p(n)$.

2.1 Learning with Statistical Queries

Let $X = \Sigma^n$ or $\Sigma^{\leq n}$, let D be a probability distribution over X and f a mapping from X to $\{0, 1\}$. The goal of learning is to be able to predict $f(x)$ well when x is drawn according to D. Statistical queries provide a particular way of gathering information about the function f. They were introduced by Kearns to characterize a wide class of noise tolerant learning algorithms [8].

The statistics oracle $\mathrm{STAT}(f, D)$ answers statistical queries, each of which specifies two arguments: a predicate χ mapping $X \times \{0, 1\}$ to $\{0, 1\}$, and a tolerance $\tau \in [0, 1]$. The answer returned by the oracle is any number v such that

$$|\mathrm{E}_D[\chi(x, f(x))] - v| \leq \tau.$$

That is, the oracle may return any number v within an additive error τ of the expected value of χ on a labeled example $(x, f(x))$ where x is drawn according to D and classified using f. A statistical query abstracts the process of drawing a sample of labeled examples $(x, f(x))$ according to D to estimate the probability that they satisfy the predicate χ. For example, a statistical query might be used to estimate the probability that the conjunction of two literals is equal to the label of an example.

A concept class $\mathcal{C}$ with polynomially bounded representations is (strongly) learnable in polynomial time using statistical queries if there exists a polynomial p and a learning algorithm A such that for every positive integer n, for every $f \in \mathcal{C}_n$, for every probability distribution D on Σ^n and for every $\epsilon > 0$, A with inputs n and ϵ and access to the statistics oracle $\mathrm{STAT}(f, D)$ satisfies the following properties.

1. For every statistical query (χ, τ) made by A, the predicate χ can be evaluated in time bounded by $p(1/\epsilon, n)$ and $1/\tau < p(1/\epsilon, n)$.
2. A runs in time bounded by $p(1/\epsilon, n)$.
3. A outputs a hypothesis h such that $h(x)$ can be evaluated in time bounded by $p(1/\epsilon, n)$ and when x is drawn from D, the probability that $h(x) \neq f(x)$ is at most ϵ.

For a concept class that does not have polynomially bounded representations, the polynomial p has an additional parameter $\text{size}(f)$ bounding the length of the representation of the target concept f. For a randomized learning algorithm A we require that A return an ϵ-good hypothesis h with at least an inverse polynomial probability. A single statistical query suffices to estimate the accuracy of a candidate hypothesis, allowing the probability of success to be boosted easily using repeated runs.

To define weak learnability, we omit ϵ and ask that the probability that $h(x) \neq f(x)$ be bounded by $1/2 - 1/q(n)$ for some polynomial q. That is, the error rate in this case need only be better than $1/2$ by an inverse polynomial. Polynomial time weak learnability using statistical queries has been shown to imply polynomial time strong learnability using statistical queries by Aslam and Decatur [1].

To define learnability using a polynomial number of statistical queries, we replace the requirement that A run in polynomial time with the requirement that A make at most a polynomial number of statistical queries. We still require that each statistical query have a polynomial time evaluatable predicate and an inverse polynomial tolerance.

To define learnability of random target concepts, we assume that there is a probability distribution on the elements of $\mathcal{C}_n$ that determines the choice of the target concept f. We require that the probability that A fails to satisfy the required conditions be $o(1)$ as a function of n. Thus, there may be a subset of the concepts in $\mathcal{C}_n$ on which A always fails, but the probability of that set must tend to zero as n increases.

2.2 The Parity Learning Problem

A parity function over n variables with ℓ relevant variables is a mapping from Σ^n to $\{0, 1\}$ that is equal to the sum modulo 2 of a fixed subset of ℓ of its arguments. For the problem of learning a parity function ψ over n variables with $\ell(n)$ relevant variables with respect to the uniform distribution on examples, a learning algorithm is given n, $\ell(n)$ and access to the statistics oracle $\text{STAT}(\psi, U)$, where U is the uniform distribution over Σ^n. When $\ell(n) = \Theta(\log n)$, no learning algorithm, even a randomized one, can achieve weak learning for this problem using polynomially many statistical queries [2].

3 Random DNF Formulas

In this section we prove the lower bound for random DNF formulas. The embedding is quite straightforward in this case, and highlights the general framework of the reduction. The framework is similar for random log depth decision trees and random deterministic finite acceptors, but the embeddings are somewhat more complex.

3.1 Model

Let n be positive integer and $V = \{v_1, \ldots, v_n\}$. We adopt the model used by Sellie of random DNF formulas over the variables V. Each term is a conjunction of $c \log(n)$ literals created by selecting a random subset of $c \log(n)$ variables and negating each variable independently with probability $1/2$. The target random DNF formula is a disjunction of independently selected terms. Sellie gives a polynomial time algorithm to learn a random DNF with at most $n^c \log\log(n)$ terms under the uniform distribution on inputs [12].

For ease of description we first consider random monotone DNF formulas, in which the step of negating variables is omitted; the general case is described later. Given a positive integer ℓ, let $n = 2^{3\ell}$; then we have $\ell = \frac{1}{3}\log(n)$ and $2^\ell = n^{1/3}$. Let ϕ denote a random monotone DNF formula of $t = 2^{\ell-1}$ terms, where each term contains ℓ variables.

We first show that with probability $1 - o(1)$, no variable occurs more than once in ϕ, that is, ϕ is a read once formula. We can think of the process of choosing terms as successive random choices of a set of ℓ variables. If each set chosen avoids the variables chosen by the previous sets, then the formula is read once. Consider the last set chosen; it must avoid a collection of at most $s = (t-1)\ell$ variables, which it does with probability at least

$$\binom{(n-s)}{\ell}\binom{n}{\ell}^{-1} \geq 1 - \frac{s\ell}{n-s+1},$$

provided n is sufficiently large. Thus the probability of failure for the last term is $O((\log^2 n)/n^{2/3})$, and the probability that any of the $O(n^{1/3})$ terms fails is at most $O((\log^2 n)/n^{1/3})$, which bounds the probability that ϕ is not read once.

In what follows we assume that ϕ is read once. As an example, for $\ell = 3$ we have $n = 512$ and $t = 4$ and a possible value of ϕ is the following.

$$\phi = v_{14}v_{133}v_{170} \vee v_{22}v_{101}v_{337} \vee v_{55}v_{266}v_{413} \vee v_{10}v_{332}v_{507}$$

3.2 Embedding Parity

We consider the problem of learning a parity function with ℓ relevant variables from a total of $m = n/t$ variables $Y = \{y_1, y_2, \ldots, y_m\}$ with respect to the uniform distribution on assignments to Y. Because $\ell = \Theta(\log(n))$ and $m = \Theta(n^{2/3})$, such a function cannot be weakly learned using a polynomial number of statistical queries with respect to the uniform distribution [2].

Let L denote the set of literals over Y. A mapping from V to L is an *equi-grouping* if exactly $n/(2m) = t/2$ variables in V are mapped to each literal in L.

With respect to an arbitrary mapping f from V to L, an assignment a to the variables Y induces an assignment a_f to the variables V by $a_f(v_i) = a(f(v_i))$, that is, the value assigned to variable v_i is the value assigned by a to the literal $f(v_i) \in L$. More generally, a distribution D over assignments a to variables in Y induces a distribution D_f over assignments b to variables in V, where $D_f(b)$ is the sum of $D(a)$ such that $a_f = b$.

Fix an arbitrary parity function with ℓ relevant variables from Y. It can be represented by a DNF formula ψ of $t = 2^{\ell-1}$ terms, where each term has exactly one literal for each relevant variable, and the number of positive literals in each term is odd. For example, if the relevant variables are $\{y_{33}, y_{57}, y_{108}\}$, we have

$$\psi = y_{33}y_{57}y_{108} \vee y_{33}y'_{57}y'_{108} \vee y'_{33}y_{57}y'_{108} \vee y'_{33}y'_{57}y_{108}.$$

Note that the parity formula ψ and the random DNF ϕ each contain t terms of ℓ literals each. We describe a straightforward embedding of ψ into ϕ.

Choose a random bijection between the terms of ϕ and the terms of ψ, and for each term, a random bijection between the variables in the term of ϕ and the literals in the corresponding term of ψ. If v_i is a variable in ϕ, let $f(v_i)$ be the corresponding literal in ψ. Because the variables in ϕ are all distinct, f maps exactly $t/2$ distinct variables of ϕ to each literal of ψ.

Extend f arbitrarily to an equi-grouping by randomly dividing the unused variables in V into groups of size $t/2$ and mapping each group to a random distinct one of the unused literals in L. For every assignment a to the variables Y, $\psi(a) = \phi(a_f)$, so this construction embeds the parity function ψ into the random monotone DNF formula ϕ.

Continuing the example of ϕ and ψ, we could choose f to map v_{14} and v_{22} to y_{33}, v_{55} and v_{10} to y'_{33}, also v_{133} and v_{266} to y_{57}, v_{101} and v_{332} to y'_{57} and finally, v_{170} and v_{507} to y_{108}, v_{337} and v_{413} to y'_{108}, though different bijections are permitted. The unused 500 variables v_i are divided arbitrarily into groups of two and each group of two is mapped to one of the 250 unused literals y_j or y'_j. An assignment to the variables y_j induces an assignment to the variables v_i in which the variables in each group of two take on the value of the literal they are mapped to.

Note that the uniform distribution U on assignments to Y induces the distribution U_f on assignments to V, in which groups of $t/2$ variables are assigned the same value, the groups corresponding to a literal and its complement receive complementary values, and groups corresponding to different variables are independent.

3.3 Reduction

We now describe a reduction showing that a learning algorithm A that weakly learns a random monotone DNF formula ϕ (over n variables with t terms and ℓ variables per term) with respect to an arbitrary distribution D using a polynomial number of statistical queries to oracle $\mathrm{STAT}(\phi, D)$ could be used to weakly learn an arbitrary parity function ψ (over m variables with ℓ relevant variables) with respect to the uniform distribution using the same number of statistical queries to oracle $\mathrm{STAT}(\psi, U)$.

For the reduction, we randomly choose an equi-grouping g mapping V to L. We then run A with variables V, simulating access to a statistical query oracle $\mathrm{STAT}(\phi, U_g)$, where ϕ is a DNF formula that embeds ψ with respect to g. (That is, $\psi(a) = \phi(a_g)$ for all assignments a to the variables Y.)

A statistical query (χ, τ) made by A is transformed to (χ', τ), where

$$\chi'(a, b) = \chi(a_g, b).$$

That is, χ' transforms the assignment a to Y into the assignment a_g to V, keeps the label b, and applies χ. The query (χ', τ) is asked of the statistical query oracle $\mathrm{STAT}(\psi, U)$

for the parity problem, and the answer is returned as the answer to A's query (χ, τ). Even though we do not know a correct function ϕ embedding ψ, this transformation allows us to answer the statistical queries of A correctly for some such ϕ.

When A halts and returns a hypothesis h, the reduction halts and outputs a hypothesis $h'(a) = h(a_g)$. That is, the hypothesis h' transforms an assignment a to Y to the assignment a_g to V and applies h.

3.4 Correctness

Suppose that instead of choosing a random equi-grouping g, we could generate a random monotone DNF formula ϕ, rejecting if it is not read once, and otherwise choosing an embedding f of ψ into ϕ as described in the embedding subsection. The resulting distribution over equi-groupings f would still be uniform. Because ϕ is read once with probability $1 - o(1)$, this means that if A succeeds in weakly learning random monotone DNF formulas, the reduction gives a randomized algorithm to learn with probability $1 - o(1)$ an arbitrary parity function over m variables with ℓ relevant variables to the same level of accuracy using the same number of statistical queries with the same tolerances as A.

The extension to general (non-monotone) DNF formulas is straightforward; a general DNF formula with n variables, t terms and ℓ variables per term is read once with probability $1 - o(1)$, and embedding a parity function ψ into a general read once formula just requires mapping literals (rather than variables) over V to literals over Y and modifying the definition of the induced assignment a_g appropriately. Thus we conclude the following.

Theorem 1. *No algorithm can weakly learn random monotone (or general) DNF formulas with n variables, $n^{1/3}$ terms and $\log n$ variables per term with respect to an arbitrary distribution using a polynomial number of statistical queries.*

3.5 Extensions

This technique can also be used to show lower bounds for DNF formulas with more or fewer terms. If $(t\ell)^2$ is $o(n)$, then a random DNF with n variables, t terms and ℓ variables per term will be read once with probability $1 - o(1)$. If the number of terms is larger than $2^{\ell-1}$, it can be trimmed by choosing one literal per excess term to fix to the value 0 so that the term is eliminated under the constructed distribution. If the number of terms is smaller than $2^{\ell-1}$, we can choose a number of literals per term to fix to the value 1 so that the term effectively becomes smaller under the constructed distribution. For example, we could use the same logic to embed parity functions with $\Theta(\log\log(n))$ relevant variables (which are still not weakly learnable with a polynomial number of statistical queries) by using $\Theta(\log(n))$ terms.

To handle these cases, instead of just choosing an equi-grouping g from V to L, the reduction first randomly chooses an appropriate number of variables from V to set randomly and independently to 0 or 1, and then chooses an equi-grouping on the rest. The resulting induced distribution on assignments to V is constant on some variables, and behaves as before on the rest.

4 Random Decision Trees

The reduction for decision trees is slightly more complex than that for DNF formulas: with high probability the depth k top of a random decision tree of depth $3k$ will query all distinct variables, which can be used to embed an arbitrary depth k decision tree by choosing random paths of length $2k$ from the boundary nodes to leaf nodes.

4.1 Model

We consider binary decision trees over n variables $V = \{v_1, \ldots, v_n\}$ of uniform depth $k \geq 0$, where $n = 2^k$. The nodes of the tree are indexed by binary strings of length at most k, where the empty string is the root and the two children of x are $x0$ and $x1$. The string indexing a node gives the sequence of query answers to reach that node.

A *decision tree* of depth k is specified by two maps, $\alpha : \Sigma^{<k} \to V$ and $\beta : \Sigma^k \to \Sigma$, where α determines the variable queried at each internal node and β determines the binary label of each leaf. We require that the variables queried along any path in the tree be distinct. The value of a decision tree on an assignment a mapping V to $\{0, 1\}$ is determined by querying the values of the variables indicated by α to reach a leaf of the tree, whose β value is the desired output.

For a random decision tree of depth $k \geq 1$, the values of α are chosen uniformly at random and the values of β are chosen such that for all $x \in \Sigma^{k-1}$, one of $\beta(x0)$ and $\beta(x1)$ is 0 and the other is 1, where both outcomes have equal probability and the choices for different x's are independent. This is one of the models of random decision trees considered by Jackson and Servedio [6]; results for their other models should be similar.

4.2 Embedding

We show that an arbitrary decision tree of depth k can be embedded with probability $1 - o(1)$ in a random decision tree of depth $3k$ (or, with a bit more work, depth $(2+\epsilon)k$). Fix an arbitrary decision tree $T = (\alpha, \beta)$ of depth k over the variables $V = \{v_1, \ldots, v_n\}$. Let $T' = (\alpha', \beta')$ be a random decision tree of depth $k' = 3k$ with $n' = 2^{k'} = n^3$ variables $W = \{w_1, \ldots, w_{n'}\}$.

For each $x \in \Sigma^k$, let $y(x) \in \Sigma^{k'-1-k}$ be chosen uniformly at random. That is, for each internal node x at depth k in T', we choose a random extension $y(x)$ of its path that reaches the parent of a pair of leaves in T'. Let H' be the set of prefixes of strings $xy(x)$; these are the nodes along any of the n chosen paths.

Define T' to be *favorable* if the map α' is one-to-one on the domain H', that is, all the variables queried along the chosen paths in T' are distinct. T' is favorable with probability $1 - o(1)$ by a union bound, because the set H' has $O(n \log n)$ elements. We assume that T' is favorable in what follows. The embedding is illustrated in Figures 1 and 2.

We now define a map g from W to $V \cup \{0, 1\}$. If $g(w_i) = v_j$ we say w_i copies v_j, and if $g(w_i) = 0$ or $g(w_i) = 1$ we say that w_i is fixed to the corresponding constant value. For $x \in \Sigma^{<k}$, we define $g(\alpha'(x)) = \alpha(x)$, that is, the variable $\alpha'(x)$ copies the variable $\alpha(x)$. For $x \in \Sigma^k$ and z a proper prefix of $y(x)$, we define $g(\alpha'(xz)) = b$

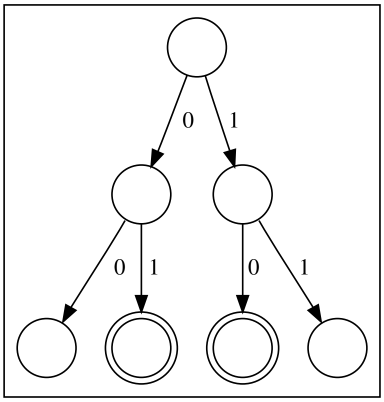


Fig. 1. The arbitrary decision tree T. Leaves with output 1 are indicated by double circles.

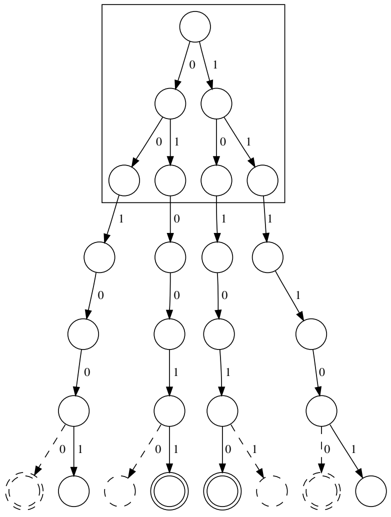


Fig. 2. The subgraph of the embedding tree T' containing a copy of T, the chosen random paths, and the correctly chosen outputs. T' is favorable if all the internal nodes of this subgraph have distinct variables.

where $b \in \Sigma$ is the unique value such that $xzb \in H'$, that is, the variable $\alpha'(xz)$ is fixed to the value necessary to follow the path chosen below x. For $x \in \Sigma^k$, we define $g(\alpha'(xy(x))) = b$ where $b \in \Sigma$ is the unique value such that $\beta'(xy(x)b) = \beta(x)$. That is, the variable $\alpha'(xy(x))$ is fixed to the value necessary to arrive at the leaf of T' with the same output as T at x. This is possible because β' takes on 0 and 1 in some order for the two children of $xy(x)$. At this point, each variable v_i in V has been assigned a distinct copy in W for each occurrence of v_i in T, so each variable in V has at most $n-1$ copies in W. We now choose unused variables from W without replacement to bring each variable in V up to exactly $n-1$ copies. Each remaining unused variable in W is fixed to 0 or 1 randomly and independently.

The mapping g induces a mapping from an assignment a to V to an assignment a_g to W by $a_g(w_j) = a(g(w_j))$ for all $j = 1, \ldots, n'$. For every assignment a to V, the output of T on a is the same as the output of T' on a_g. Thus, for any distribution D on assignments a, statistical queries to $\mathrm{STAT}(T', D_g)$ can be answered by making statistical queries to $\mathrm{STAT}(T, D)$.

4.3 Reduction

Suppose A is an algorithm that weakly learns random log depth decision trees using statistical queries with respect to an arbitrary distribution. Faced with the problem of learning an arbitrary depth k decision tree T over the $n = 2^k$ variables V with respect to a distribution D using statistical queries to $\mathrm{STAT}(T, D)$, we let $k' = 3k$ and $n' = n^3$

and $W = \{w_1, \ldots, w_{n'}\}$. Define the set of *suitable reduction mappings* to be all g mapping g from W to $V \cup \{0, 1\}$ subject to the condition that $|g^{-1}(v_i)| = n - 1$ for $i = 1, \ldots, n$, that is, exactly $n - 1$ variables in W are mapped to each variable in V and the remaining variables in W are mapped to 0 or 1. Choose a random suitable reduction mapping g.

We run A with the n' variables W and depth parameter k', simulating statistical queries to $\mathrm{STAT}(T', D_g)$ for some decision tree T' of depth k' over the variables W. When A makes a statistical query (χ, τ), we define

$$\chi'(a) = \chi(a_g),$$

make a statistical query to $\mathrm{STAT}(T, D)$ with (χ', τ) and return the answer to A. When A halts with output h, we output h', where $h'(a) = h(a_g)$.

4.4 Correctness

The distribution over suitable reduction mappings g would be uniform if we could first generate a random depth k' decision tree T' and a random choice of paths $y(x)$ from its nodes at depth k (rejecting if T' is not favorable) and then proceed to choose g to embed T in T' as described in the embedding subsection. Because the probability that we reject T' as not favorable is $o(1)$, with probability $1 - o(1)$, A weakly learns a tree T' embedding T, which means that the reduction must weakly learn T. Because log depth decision trees with n variables can express all parity functions over n variables with $\log n$ relevant variables, our reduction proves the following.

Theorem 2. *No algorithm can weakly learn random decision trees with n variables and depth $\log n$ with respect to an arbitrary distribution using a polynomial number of statistical queries.*

5 Random Deterministic Finite Acceptors

The embedding for random DFAs is somewhat more complex. By the results of Trakhtenbrot and Barzdin, we know that with high probability, if one state of a random DFA is reachable from another, it is reachable by a path of length $O(\log n)$. In order to represent parity, we embed two trees of $O(\log n)$ depth in the machine, but we must also find paths that return from the leaves of the trees to both their roots, in order to test the parities of a sequence of variables.

5.1 Model of Random DFAs

Let n be a positive integer and let Q be a finite set of n states with start state $q_0 \in Q$. We consider a standard model of random deterministic finite acceptors, in which the entries of the transition function $\delta : Q \times \Sigma \to Q$ and the set of accepting states $F \subseteq Q$ are chosen uniformly at random.

5.2 Representing Parity

We name variables for the parity problem using strings from a prefix-free set V constructed as follows. Let $\sigma : \{1, 2, \ldots\} \to \Sigma^*$ be the bijection defined by

$$\begin{aligned}\sigma(1) &= \epsilon\\ \sigma(2m) &= \sigma(m)0\\ \sigma(2m+1) &= \sigma(m)1.\end{aligned}$$

Thus σ maps m to its binary representation after the first 1. Let $\ell = \lceil \log^2 n \rceil$ and let $V = \{\sigma(m) : m \in \{\ell, \ldots, 2\ell - 1\}\}$ be the set of *variables*. Note that $|V| = \ell = \Theta(\log^2 n)$.

Our goal is to make a random machine compute a parity function whose relevant variables are any subset $U \subseteq V$. To give some intuition, we first describe how to construct a finite state machine that accepts a sequence of variables when the sequence has an odd number of occurrences of variables from a given set U. As an example, assume that $\ell = 6$, which gives the set of variables

$$V = \{10, 11, 000, 001, 010, 011\}.$$

Assume that the set of relevant variables is the following.

$$U = \{10, 000, 001\}.$$

The representation we choose for an assignment $a : V \to \{0, 1\}$ is to list (in some order) the variables $v \in V$ such that $a(v) = 1$. For example, if a assigns 1 to the variables 10, 11, 000, and 010, one representation of a would be 1011000010.

Given this representation, we construct a finite automaton to accept the strings representing assignments with odd parity on the variables in U as follows. Let V' be the set of proper prefixes of strings from V. For each string $v' \in V'$ there are states $[q_0, v']$ and $[q_1, v']$. If for some $b \in \Sigma$ both $v' \in V'$ and $v'b \in V'$ then let

$$\delta([q_0, v'], b) = [q_0, v'b] \text{ and } \delta([q_1, v'], b) = [q_1, v'b].$$

Suppose for some $b \in \Sigma$, $v' \in V'$ and $vb \in V$. If $v'b \in U$, that is, $v'b$ is a relevant variable, we exchange even and odd as follows:

$$\delta([q_0, v'], b) = [q_1, \varepsilon] \text{ and } \delta([q_1, v'], b) = [q_0, \varepsilon].$$

If $v'b \notin U$, we do not exchange even and odd:

$$\delta([q_0, v'], b) = [q_0, \varepsilon] \text{ and } \delta([q_1, v'], b) = [q_1, \varepsilon].$$

The start state is $[q_0, \varepsilon]$ and the only accepting state is $[q_1, \varepsilon]$. After reading in a sequence of distinct variables, the machine is in state $[q_0, \varepsilon]$ if an even number of the variables read are from U, and in state $[q_1, \varepsilon]$ if an odd number of them are from U.

The machine constructed by this process for the example values of U and V is illustrated in Figure 3. Note that on the input string 1011000010 the machine reaches state

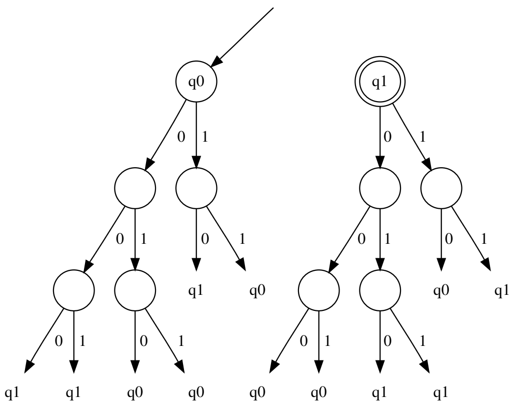


Fig. 3. One idea for a finite state machine to compute the parity of variables $U = \{10, 000, 001\}$ from $V = \{10, 11, 000, 001, 010, 011\}$. The in-degrees of q_0 and q_1 are in general too large.

q_0 and rejects, which is correct because in this assignment exactly two of the variables in U (namely 10 and 000) are set to 1.

An attempt to embed this kind of machine for computing parity into a random finite state acceptor encounters the problem that the in-degrees of the states $[q_0, \varepsilon]$ and $[q_1, \varepsilon]$ are in general too large. We avoid this problem by changing the input representation to allow each variable to be followed by a string that prepares the machine to accept the next variable. Even though each variable may have a different string, the contents and assignments of these strings do not reveal any information about the relevant variables.

5.3 Embedding and Reduction

We show how to embed a parity computation on the variables V into a random finite state acceptor M with n states with at least constant probability of success. The process is illustrated in Figures 4, 5 and 6.

We first choose any state q_1 different from the start state q_0. We generate a random finite acceptor M with n states. We think of the structure of M as being revealed to us in stages. With probability $1/4$, we have $q_0 \notin F$ and $q_1 \in F$.

Let $V' = \{\sigma(m) : m \in \{1, \ldots, \ell - 1\}\}$. These are all the proper prefixes of the variables V. Because $|V'| = O(\log^2 n)$, with probability $1 - o(1)$, there are two non-overlapping trees rooted at q_0 and q_1, that is, the set of states $R = \{\delta(q, v') : q \in \{q_0, q_1\},\ v' \in V'\}$ has cardinality $2|V'|$. In this case, the values of $\delta(q, v)$ for $q \in \{q_0, q_1\}$ and $v \in V$ are all independent random choices.

For each variable $v \in V$ we would ideally like to find a string x_v such that x_v takes $\delta(q_0, v)$ and $\delta(q_1, v)$ back to the states q_0 and q_1 (in some order), that is,

$$\{\delta(q_0, vx_v), \delta(q_1, vx_v)\} = \{q_0, q_1\}.$$

Depending on the order, the variable v is or is not relevant. Though we cannot achieve this *proper functioning* for all variables, we can do so for a constant fraction of the variables.

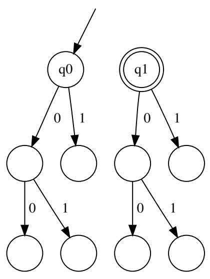


Fig. 4. The even state is q_0 and the odd state is q_1. With probability $1-o(1)$, there are two non-overlapping trees with $\ell - 1$ nodes rooted at q_0 and q_1. We don't yet commit to the outgoing arcs of the leaves.

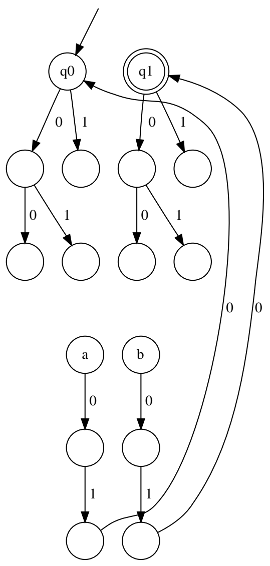


Fig. 5. With constant probability, a pair $(a, b) \in Q^2$ chosen uniformly at random can reach (q_0, q_1) via the same string while avoiding the trees

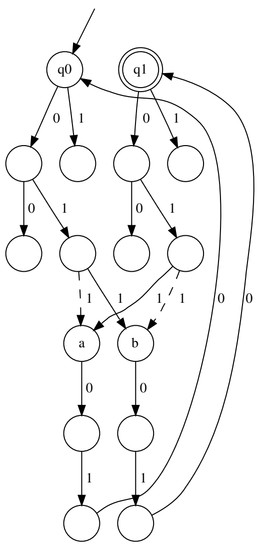


Fig. 6. Now we choose the outgoing arcs corresponding to variable 011. With constant probability, there is a path back to (q_0, q_1). The solid arcs signify a relevant variable; the dashed ones, an irrelevant variable. These cases are equally likely and independent of the string to prepare for another variable.

We first show that with constant probability, for two random states $(a, b) \in Q^2$, there is a short string x such that $\{\delta(a, x), \delta(b, x)\} = \{q_0, q_1\}$ and neither path touches the states in R. The latter stipulation is important because it enables a bijection that swaps the values of $\delta(q_0, v)$ and $\delta(q_1, v)$, which allows us to conclude that the return strings x_v don't give away the relevant variables.

The proof of existence of the short string is as follows. Let $(a, b) \in Q^2$ be chosen uniformly at random, assume n is even, and let $X = \{\sigma(m) : m \in \{n^2/4, \ldots, n^2/2 - 1\}\}$. We show that if $x \in X$ is chosen uniformly at random, then with probability $(2 - o(1))/n^2$, we have $\{\delta(a, x), \delta(b, x)\} = \{q_0, q_1\}$, that is, x is *good*. Also, if $y \in X$ is chosen independently, then the probability that both x and y are good is $(4-o(1))/n^4$. By inclusion-exclusion, the probability that exactly one string in X is good is at least

$$\begin{aligned}|X|(2-o(1))/n^2-2\binom{|X|}{2}(4-o(1))/n^4 &= (1 - o(1))/2 - 2\binom{n^2/4}{2}(4 - o(1))/n^4\\ &= (1 - o(1))/2 - (1 - o(1))/4\\ &= (1 - o(1))/4.\end{aligned}$$

The key observation is that the success event of x and the success event of y are very close to being independent Bernoulli trials with success probability $2/n^2$. The strings under consideration have length about $2 \log n$. If just before the final letter of each string we have reached from a and b two different states whose outward transitions have not been committed in any way, the success probabilities *will* be independent and exactly $2/n^2$. Of course, something may go wrong before then: we may touch a state in R or have the paths loop or touch one another. With only $O(\log^2 n)$ states to avoid however, this event is probability $o(1)$.

Finally, we observe that with constant probability, there will be $\log^2 n/8$ variables that function properly. We can embed any parity function over these variables, which is enough to make $n^{\Theta(\log n)}$ parity functions, ensuring that a superpolynomial number of statistical queries are needed in expectation [2].

For the reduction, an assignment a to $\log^2 n/8$ variables is transformed to a string containing the list of strings vx_v for those variables v on which a takes the value 1. Note that these strings are all of length $O(\log^3 n)$. If desired, the strings could be padded by repeating the variables they contain an odd number of times.

Theorem 3. *No algorithm can weakly learn random deterministic finite acceptors with n states with respect to an arbitrary distribution on strings of length at most $\Theta(\log^3 n)$ using a polynomial number of statistical queries.*

6 Discussion

As described in Section 1, there are polynomial time algorithms to learn certain classes of random decision trees and random DNF formulas with respect to the uniform distribution, and these algorithms can be implemented with statistical queries. However, it is open whether random deterministic finite acceptors of n^c states can be learned in polynomial time with respect to the uniform distribution on strings of length n.

Acknowledgements

During parts of this work Aryeh Kontorovich was in the Department of Mathematics of the Weizmann Institute and Lev Reyzin was in the Department of Computer Science of Yale University.

References

[1] Aslam, J.A., Decatur, S.E.: General bounds on statistical query learning and PAC learning with noise via hypothesis boosting. Information and Computation 141(2), 85–118 (1998)

[2] Blum, A., Furst, M., Jackson, J., Kearns, M., Mansour, Y., Rudich, S.: Weakly learning DNF and characterizing statistical query learning using Fourier analysis. In: STOC 1994: Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, pp. 253–262. ACM, New York (1994)

[3] Freund, Y., Kearns, M.J., Ron, D., Rubinfeld, R., Schapire, R.E., Sellie, L.: Efficient learning of typical finite automata from random walks. Information and Computation 138(1), 23–48 (1997)

[4] Mark Gold, E.: Complexity of automaton identification from given data. Information and Control 37(3), 302–320 (1978)

[5] Jackson, J., Lee, H., Servedio, R., Wan, A.: Learning random monotone DNF. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX and RANDOM 2008. LNCS, vol. 5171, pp. 483–497. Springer, Heidelberg (2008)

[6] Jackson, J.C., Servedio, R.A.: Learning random log-depth decision trees under the uniform distribution. SIAM J. Comput. 34(5), 1107–1128 (2005)

[7] Kearns, M., Valiant, L.G.: Crytographic limitations on learning boolean formulae and finite automata. In: STOC 1989: Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, pp. 433–444. ACM, New York (1989)

[8] Kearns, M.: Efficient noise-tolerant learning from statistical queries. J. ACM 45(6), 983–1006 (1998)

[9] Lang, K.J.: Random DFA's can be approximately learned from sparse uniform examples. In: COLT 1992: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 45–52. ACM, New York (1992)

[10] Pitt, L., Warmuth, M.K.: Prediction-preserving reducibility. J. Comput. Syst. Sci. 41(3), 430–467 (1990)

[11] Sellie, L.: Learning random monotone DNF under the uniform distribution. In: COLT, pp. 181–192 (2008)

[12] Sellie, L.: Exact learning of random DNF over the uniform distribution. In: STOC, pp. 45–54 (2009)

[13] Trakhtenbrot, B.A., Barzdin, Y.M.: Finite Automata: Behavior and Synthesis. North Holland, Amsterdam (1973)

Recursive Teaching Dimension, Learning Complexity, and Maximum Classes[⋆]

Thorsten Doliwa[1], Hans Ulrich Simon[1], and Sandra Zilles[2]

[1] Fakultät für Mathematik, Ruhr-Universität Bochum
D-44780 Bochum, Germany
{thorsten.doliwa,hans.simon}@rub.de
[2] Department of Computer Science, University of Regina
Regina, SK, Canada S4S 0A2
zilles@cs.uregina.ca

Abstract. This paper is concerned with the combinatorial structure of concept classes that can be learned from a small number of examples. We show that the recently introduced notion of recursive teaching dimension (RTD, reflecting the complexity of teaching a concept class) is a relevant parameter in this context. Comparing the RTD to self-directed learning, we establish new lower bounds on the query complexity for a variety of query learning models and thus *connect teaching to query learning.*

For many general cases, the RTD is upper-bounded by the VC-dimension, e.g., classes of VC-dimension 1, (nested differences of) intersection-closed classes, "standard" boolean function classes, and finite maximum classes. The RTD thus is the first model to *connect teaching to the VC-dimension.*

The combinatorial structure defined by the RTD has a remarkable resemblance to the structure exploited by sample compression schemes and hence *connects teaching to sample compression.* Sequences of teaching sets defining the RTD coincide with unlabeled compression schemes both (i) resulting from Rubinstein and Rubinstein's corner-peeling and (ii) resulting from Kuzmin and Warmuth's Tail Matching algorithm.

1 Introduction

The complexity of the problem of learning a concept C in a given concept class $\mathcal{C}$ can be measured in different ways. If A is a learning algorithm of a particular type, one measures for instance how much information A must process, how many prediction errors A will make on single attributes of C, or how expensive the computation executed by A is, when identifying C. The worst-case behavior of A is given by the highest such amount measured over all concepts $\mathcal{C}$. The complexity value assigned to $\mathcal{C}$ with respect to the underlying learning model is then defined as the best possible worst-case behavior of any learning algorithm.

While run-time and memory complexity are important aspects of machine learning problems, the aspect of "information complexity" (e.g., how many labeled data points are needed for learning) has at least equally important status.

[⋆] This work was supported by the Deutsche Forschungsgemeinschaft Grant SI 498/8-1 and by the Natural Sciences and Engineering Research Council of Canada (NSERC).

© Springer-Verlag Berlin Heidelberg 2010

From an application point of view, the cost of a machine learning process is often dominated by the amount of data needed. From a theoretical point of view, the study of information complexity yields formal guarantees concerning the amount of data that needs to be processed to solve a learning problem. Moreover, analyzing information complexity often helps to understand the structure of a given class of target concepts. In addition, the theoretical study of information complexity helps to identify parallels between various formal models of learning.

The reason for these parallels is that algorithms used in a number of different formal learning models reflect principles related to sample compression schemes, i.e., schemes for "encoding" a set of examples in a small subset of examples. For instance, from the set of examples they process, learning algorithms often extract a subset of particularly "significant" examples in order to represent their hypotheses. This way sample bounds for PAC-learning of a class $\mathcal{C}$ can be obtained from the size of a smallest sample compression scheme for $\mathcal{C}$, see [14, 5]. Here the size of a scheme is the size of the largest subset resulting from compression of any sample consistent with some concept in $\mathcal{C}$. Similarly teachers, which provide examples to the learner in models of co-operative learning, compress concepts to subsets of particularly "helpful" examples, cf. [6, 19, 10, 2].

In the past two decades, several learning models were defined with the aim of achieving low information complexity in a non-trivial way. One such model is learning from partial equivalence queries [15], which subsumes all types of queries for which negative answers are witnessed by counterexamples, e.g., membership, equivalence, subset, superset, and disjointness queries [1]. As lower bounds on the information complexity in this model (here called query complexity) hold for numerous learning models, they are particularly interesting objects of study.

In the query model of self-directed learning [7], a query is a prediction of a label for an instance of the learner's choice and the learner "pays" only for wrong predictions. Self-directed learners are very powerful; they yield a query complexity lower-bounding the one obtained from partial equivalence queries [8]. Even though the self-directed learning complexity can exceed the VC-dimension, existing results show some connection between these two complexity measures.

A recent model of teaching with low information complexity is *recursive teaching*, where a teacher chooses a sample based on a sequence of nested subclasses of $\mathcal{C}$, see [22]. The nesting is defined by (i) choosing all concepts in $\mathcal{C}$ that are easiest to teach, i.e., that have the smallest sets of examples distinguishing them from all other concepts in $\mathcal{C}$ and (ii) recursively repeating this process with the remaining concepts. The largest number of examples required at any stage is the *recursive teaching dimension (RTD)* of $\mathcal{C}$. The RTD significantly improves on bounds for previous teaching models. It lower-bounds not only the complexity of the "classical" teaching model [6, 19] but also the complexity of iterated optimal teaching [2], which is often significantly below the classical teaching dimension.

Using the RTD, this paper is the first one to establish a relation between teaching complexity and complexity of query learning, between teaching complexity and the VC-dimension, as well as between teaching complexity and sample compression, in particular revealing a surprisingly strong connection to unlabeled sample compression, cf. [4, 12]. No such relations are exhibited by the classical teaching models. Our main contributions are the following.

(i) We show that the RTD is never higher (and often considerably lower) than the complexity of *self-directed learning*. Hence all lower bounds on RTD will hold for self-directed learning, for learning from partial equivalence queries, and for a variety of other query learning models.

(ii) We establish a connection between the RTD and the *VC-dimension*. Though there are classes for which the RTD exceeds the VC-dimension, we present a number of quite general and natural cases in which the RTD is upper-bounded by the VC-dimension. These include classes of VC-dimension 1, intersection-closed classes and nested differences thereof, a variety of naturally structured boolean function classes, and finite maximum classes in general (i.e., classes of maximum possible cardinality for a given VC-dimension and domain size). It remains open whether every class of VC-dimension d has an RTD linear in d.

(iii) We establish a connection between the RTD and *unlabeled compression schemes*. To prove that the RTD of a finite maximum class equals its VC-dimension, we use a recent result from [17]. Rubinstein and Rubinstein show that, for all maximum classes, a technique called corner-peeling defines unlabeled compression schemes whose size equals the VC-dimension. Corner-peeling is a particular way of recursively removing concepts from the given concept class, while representing every such peeling step by a small subset of the underlying instance space, i.e., an unlabeled sample. Firstly, the recursive nesting of concept classes is common to both peeling and RTD. Secondly, and more importantly, we observe that every maximum class allows corner-peeling with an additional property, which ensures that the resulting unlabeled samples contain exactly those instances a teacher following the RTD model would use. A closer look reveals the following two facts for any finite maximum class $\mathcal{C}$ of VC-dimension d:

• Both Rubinstein and Rubinstein's corner-peeling *and* Kuzmin and Warmuth's Tail Matching [12] construct unlabeled compression schemes for $\mathcal{C}$ that map to samples exactly coinciding with those used in the RTD model for $\mathcal{C}$. All samples are of size at most d.

• The RTD model allows for a nesting of $\mathcal{C}$ that uses samples of size at most d whose unlabeled versions form an unlabeled compression scheme of size d.

The correspondence between RTD and compression schemes is quite remarkable, because these models arose in different branches of Learning Theory and, for that reason, differ in several respects:

• The RTD-model has comparatively restrictive rules for producing teaching sets (which is a kind of compression).

• It does not explicitly address the issue of *sample* compression (but rather compresses the concept as a function on the whole domain).

Despite these differences, sample compression schemes lead to RTD-nestings for a wide variety of classes (including linear arrangements and halfspaces). Consequently, the question of whether or not the RTD is linear in the VC-dimension appears to be related to the long-standing open question of whether or not the sample compression complexity is linear in the VC-dimension, cf. [14]. We believe that studying the RTD will continue to provide new insights into the combinatorial structure of concept classes that possess small compression schemes.

2 Definitions, Notations and Facts

Throughout this paper, X denotes a finite set and $\mathcal{C}$ denotes a concept class over the domain X. For $X' \subseteq X$, we define $\mathcal{C}_{|X'} := \{C \cap X' |\ C \in \mathcal{C}\}$. We treat concepts interchangeably as subsets of X and as $0,1$-valued functions on X. A labeled example is a pair (x, l) with $x \in X$ and $l \in \{0, 1\}$. If S is a set of labeled examples, we define $X(S) = \{x \in X \mid (x, 0) \in S \text{ or } (x, 1) \in S\}$. For brevity, $[n] := \{1, \ldots, n\}$. $\mathrm{VCD}(\mathcal{C})$ denotes the VC-dimension of a concept class $\mathcal{C}$.

Definition 1. *Let K be a function that assigns a "complexity" $K(\mathcal{C}) \in \mathbb{N}$ to each concept class $\mathcal{C}$. We say that K is* monotonic *if $\mathcal{C}' \subseteq \mathcal{C}$ implies that $K(\mathcal{C}') \leq K(\mathcal{C})$. We say that K is* twofold monotonic *if K is monotonic and, for every concept class $\mathcal{C}$ over X and every $X' \subseteq X$, it holds that $K(\mathcal{C}_{|X'}) \leq K(\mathcal{C})$.*

2.1 Learning Complexity

A *partial equivalence query* [15] of a learner is given by a function $h : X \to \{0, 1, *\}$ that is passed to an oracle. The latter returns "YES" if the target concept C^* coincides with h on all $x \in X$ for which $h(x) \in \{0, 1\}$; it returns a "witness of inequivalence" (i.e., an $x \in X$ such that $C^*(x) \neq h(x) \in \{0, 1\}$) otherwise. LC-PARTIAL($\mathcal{C}$) denotes the smallest number q such that there is some learning algorithm that exactly identifies each concept $C^* \in \mathcal{C}$ with up to q partial equivalence queries (regardless of the oracle's answering strategy).

A query in the model of *self-directed learning* [7, 8] consists of an instance $x \in X$ and a label $b \in \{0, 1\}$, passed to an oracle. The latter returns the true label $C^*(x)$ assigned to x by the target concept C^*. We say the learner *made a mistake* if $C^*(x) \neq b$. The *self-directed learning complexity* of $\mathcal{C}$, denoted $\mathrm{SDC}(\mathcal{C})$, is defined as the minimum worst-case number of mistakes that a learning algorithm A can achieve on $\mathcal{C}$, if A exactly identifies every $C^* \in \mathcal{C}$.

The *mistake bound* [13] of a particular learning algorithm A for concept class $\mathcal{C}$, denoted $M_A(\mathcal{C})$, is the worst-case number of 0,1-prediction mistakes made by A on any given sequence of instances labeled consistently according to some target concept from $\mathcal{C}$. The *optimal mistake bound* for a concept class $\mathcal{C}$, denoted $M_{opt}(\mathcal{C})$, is the minimum of $M_A(\mathcal{C})$ over all learning algorithms A.

Clearly, LC-PARTIAL and SDC are monotonic, and M_{opt} is twofold monotonic. The following chain of inequalities is well-known [8, 15]:

$$\mathrm{SDC}(\mathcal{C}) \leq \text{LC-PARTIAL}(\mathcal{C}) \leq M_{opt}(\mathcal{C}) \tag{1}$$

2.2 Teaching Complexity

A *teaching set* for a concept $C \in \mathcal{C}$ is a set S of labeled examples such that C, but no other concept in $\mathcal{C}$, is consistent with S. Let $\mathcal{TS}(C, \mathcal{C})$ denote the family of teaching sets for $C \in \mathcal{C}$, let $\mathrm{TS}(C;\mathcal{C})$ denote the size of the smallest teaching set for $C \in \mathcal{C}$, and let

$$\mathrm{TS}_{min}(\mathcal{C}) := \min_{C \in \mathcal{C}} \mathrm{TS}(C;\mathcal{C})\,,\ \ \mathrm{TS}_{max}(\mathcal{C}) := \max_{C \in \mathcal{C}} \mathrm{TS}(C;\mathcal{C})\,.$$

The quantity $\mathrm{TD}(\mathcal{C}) := \mathrm{TS}_{max}(\mathcal{C})$ is called the *teaching dimension* of $\mathcal{C}$ [6]. Note that TD is monotonic. A concept class $\mathcal{C}$ consisting of exactly one concept C has teaching dimension 0 because $\emptyset \in \mathcal{TS}(C, \{C\})$.

Definition 2 (see [22]). *A* teaching plan for $\mathcal{C}$ *is a sequence*

$$P = ((C_1, S_1), \ldots, (C_N, S_N)) \tag{2}$$

with the following properties:

- *$N = |\mathcal{C}|$ and $\mathcal{C} = \{C_1, \ldots, C_N\}$.*
- *For all $t = 1, \ldots, N$, $S_t \in \mathcal{TS}(C_t, \{C_t, \ldots, C_N\})$.*

The quantity $\mathrm{ord}(P) := \max_{t=1,\ldots,N-1} |S_t|$ *is called the* order of the teaching plan P. *Finally, we define*

$$\mathrm{RTD}(\mathcal{C}) := \min\{\mathrm{ord}(P) \mid P \textit{ is a teaching plan for } \mathcal{C}\}\,,$$
$$\mathrm{RTD}^*(\mathcal{C}) := \max_{X' \subseteq X} \mathrm{RTD}(\mathcal{C}_{|X'})\,.$$

The quantity $\mathrm{RTD}(\mathcal{C})$ *is called the* recursive teaching dimension of $\mathcal{C}$.

A teaching plan (2) is said to be *repetition-free* if the sets $X(S_1), \ldots, X(S_N)$ are pairwise distinct. (Clearly, the corresponding labeled sets, $S_1, \ldots, S_N$, are always pairwise distinct.) As observed in [22], the following holds:

- RTD is monotonic.
- The recursive teaching dimension coincides with the order of any teaching plan that is *in canonical form*, i.e., a teaching plan $((C_1, S_1), \ldots, (C_N, S_N))$ such that $|S_t| = \mathrm{TS}_{min}(\{C_t, \ldots, C_N\})$ holds for all $t \in \{1, \ldots, N-1\}$.

Intuitively, a canonical teaching plan is a sequence that is recursively built by always picking an easiest-to-teach concept C_t in the class $\mathcal{C} \setminus \{C_1, \ldots, C_{t-1}\}$ together with an appropriate teaching set S_t.

The definition of teaching plans immediately yields the following result:

Lemma 3. *1. If K is monotonic and $\mathrm{TS}_{min}(\mathcal{C}) \leq K(\mathcal{C})$ for every concept class $\mathcal{C}$, then $\mathrm{RTD}(\mathcal{C}) \leq K(\mathcal{C})$ for every concept class $\mathcal{C}$.*
2. If K is twofold monotonic and $\mathrm{TS}_{min}(\mathcal{C}) \leq K(\mathcal{C})$ for every concept class $\mathcal{C}$, then $\mathrm{RTD}^*(\mathcal{C}) \leq K(\mathcal{C})$ for every concept class $\mathcal{C}$.

RTD and TS_{min} are related as follows:

Lemma 4. $\mathrm{RTD}(\mathcal{C}) = \max_{\mathcal{C}' \subseteq \mathcal{C}} \mathrm{TS}_{min}(\mathcal{C}')$.

Proof. Let C_1 be the first concept in a canonical teaching plan P for $\mathcal{C}$ so that $\mathrm{TS}(C_1;\mathcal{C}) = \mathrm{TS}_{min}(\mathcal{C})$ and the order of P equals $\mathrm{RTD}(\mathcal{C})$. It follows that $\mathrm{RTD}(\mathcal{C}) = \max\{\mathrm{TS}(C_1;\mathcal{C}), \mathrm{RTD}(\mathcal{C} \setminus \{C_1\})\} = \max\{\mathrm{TS}_{min}(\mathcal{C}), \mathrm{RTD}(\mathcal{C} \setminus \{C_1\})\}$, and $\mathrm{RTD}(\mathcal{C}) \leq \max_{\mathcal{C}' \subseteq \mathcal{C}} \mathrm{TS}_{min}(\mathcal{C}')$ follows inductively. As for the reverse direction, let $\mathcal{C}_0' \subseteq \mathcal{C}$ be a maximizer of TS_{min}. Since RTD is monotonic, we get $\mathrm{RTD}(\mathcal{C}) \geq \mathrm{RTD}(\mathcal{C}_0') \geq \mathrm{TS}_{min}(\mathcal{C}_0') = \max_{\mathcal{C}' \subseteq \mathcal{C}} \mathrm{TS}_{min}(\mathcal{C}')$. $\square$

2.3 Intersection-Closed Classes and Nested Differences

A concept class $\mathcal{C}$ is called *intersection-closed* if $C \cap C' \in \mathcal{C}$ for all $C, C' \in \mathcal{C}$. Among the standard examples for intersection-closed classes are the d-dimensional boxes over domain $[n]^d$:

$$\mathrm{BOX}_n^d := \{[a_1 : b_1] \times \cdots \times [a_d : b_d] \mid \ \forall i = 1, \ldots, d : \ 1 \le a_i, b_i \le n\}$$

Here, $[a : b]$ is an abbreviation for $\{a, a+1, \ldots, b\}$, where $[a : b]$ is the empty set if $a > b$. For the remainder of this section, $\mathcal{C}$ is assumed to be intersection-closed.

For $T \subseteq X$, we define $\langle T \rangle_{\mathcal{C}}$ as the smallest concept in $\mathcal{C}$ containing T, i.e.,

$$\langle T \rangle_{\mathcal{C}} := \bigcap_{T \subseteq C \in \mathcal{C}} C\,.$$

A *spanning set* for $T \subseteq X$ w.r.t. $\mathcal{C}$ is a set $S \subseteq T$ such that $\langle S \rangle_{\mathcal{C}} = \langle T \rangle_{\mathcal{C}}$. S is called a *minimal spanning set* w.r.t. $\mathcal{C}$ if, for every proper subset S' of S, $\langle S' \rangle_{\mathcal{C}} \neq \langle S \rangle_{\mathcal{C}}$. $I(\mathcal{C})$ denotes the size of the largest minimal spanning set w.r.t. $\mathcal{C}$. It is well-known [16, 9] that every minimal spanning set w.r.t. $\mathcal{C}$ is shattered by $\mathcal{C}$. Thus, $I(\mathcal{C}) \le \mathrm{VCD}(\mathcal{C})$. Note that, for every $C^\circ \in \mathcal{C}$, $I(\mathcal{C}_{|C^\circ}) \le I(\mathcal{C})$, because each spanning set for a set $T \subseteq C^\circ$ w.r.t. $\mathcal{C}$ is also a spanning set for T w.r.t. $\mathcal{C}_{|C^\circ}$.

The class of *nested differences* of depth d (at most d) with concepts from $\mathcal{C}$, denoted $\mathrm{DIFF}^d(\mathcal{C})$ ($\mathrm{DIFF}^{\le d}(\mathcal{C})$, resp.), is defined inductively as follows:

$$\begin{aligned} \mathrm{DIFF}^1(\mathcal{C}) &:= \mathcal{C}\,, \\ \mathrm{DIFF}^d(\mathcal{C}) &:= \{C \setminus D \mid C \in \mathcal{C}, D \in \mathrm{DIFF}^{d-1}(\mathcal{C})\}\,, \\ \mathrm{DIFF}^{\le d}(\mathcal{C}) &:= \bigcup_{i=1}^{d} \mathrm{DIFF}^i(\mathcal{C})\,. \end{aligned}$$

Expanding the recursive definition of $\mathrm{DIFF}^d(\mathcal{C})$ shows that, e.g., a set in $\mathrm{DIFF}^4(\mathcal{C})$ has the form $C_1 \setminus (C_2 \setminus (C_3 \setminus C_4))$ where $C_1, C_2, C_3, C_4 \in \mathcal{C}$. We may assume without loss of generality that $C_1 \supseteq C_2 \supseteq \cdots$ because $\mathcal{C}$ is intersection-closed.

Nested differences of intersection-closed classes were examined thoroughly at an early stage of research on computational learning theory [9].

2.4 Maximum Classes and Unlabeled Compression Schemes

Let $\Phi_d(n) := \sum_{i=0}^{d} \binom{n}{i}$. For $d = \mathrm{VCD}(\mathcal{C})$ and for any subset X' of X, we have $\left|\mathcal{C}_{|X'}\right| \le \Phi_d(|X'|)$, according to Sauer's Lemma [20, 18]. The concept class $\mathcal{C}$ is called a *maximum class* if Sauer's inequality holds with equality for every subset X' of X. It is well-known [21, 5] that a class over a domain X is maximum iff Sauer's inequality holds with equality for $X' = X$.

The following definition is from [12]:

Definition 5. *An* unlabeled compression scheme *for a maximum class of VC-dimension d is given by an injective mapping r that assigns to every concept C a set $r(C) \subseteq X$ of size at most d such that the following condition is satisfied:*

$$\forall C, C' \in \mathcal{C} \ (C \neq C'), \exists x \in r(C) \cup r(C') : \ C(x) \neq C'(x)\,. \tag{3}$$

(3) is referred to as the *non-clashing property*. In order to ease notation, we add the following technical definitions. A *representation mapping of order k for a (not necessarily maximum) class $\mathcal{C}$* is any injective mapping r that assigns to every concept C a set $r(C) \subseteq X$ of size at most k such that (3) holds. A representation-mapping r is said to have the *acyclic non-clashing property* if there is an ordering $C_1, \ldots, C_N$ of the concepts in $\mathcal{C}$ such that

$$\forall 1 \leq i < j \leq N, \exists x \in r(C_i) : \; C_i(x) \neq C_j(x) \,. \tag{4}$$

Considering maximum classes, it was shown [12] that a representation mapping with the non-clashing property guarantees that, for every sample S labeled according to a concept from $\mathcal{C}$, there is exactly one concept $C \in \mathcal{C}$ that is consistent with S and satisfies $r(C) \subseteq X(S)$. This allows to encode (compress) a labeled sample S by $r(C)$ and, since r is injective, to decode (decompress) $r(C)$ by C (so that the labels in S can be reconstructed). This coined the term "unlabeled compression scheme".

A concept class $\mathcal{C}$ over a domain X of size n is identified with a subset of $\{0,1\}^n$. The *one-inclusion-graph* $\mathcal{G}(\mathcal{C})$ associated with $\mathcal{C}$ is defined as follows:

– The nodes are the concepts from $\mathcal{C}$.
– Two concepts are connected by an edge if and only if they differ in exactly one coordinate (when viewed as nodes in the Boolean cube).

A *cube* $\mathcal{C}'$ in $\mathcal{C}$ is a subcube of $\{0,1\}^n$ such that every node in $\mathcal{C}'$ represents a concept from $\mathcal{C}$. In the context of the one-inclusion graph, the instances (corresponding to the dimensions in the Boolean cube) are usually called "colors" (and an edge along dimension i is viewed as having color i).

The following definitions are from [17] (although, stylistically, we are stressing here the similarities to teaching plans):

Definition 6. *A* corner-peeling plan for $\mathcal{C}$ *is a sequence*

$$P = ((C_1, \mathcal{C}'_1), \ldots, (C_N, \mathcal{C}'_N)) \tag{5}$$

with the following properties:

1. *$N = |\mathcal{C}|$ and $\mathcal{C} = \{C_1, \ldots, C_N\}$.*
2. *For all $t = 1, \ldots, N$, $\mathcal{C}'_t$ is a cube in $\{C_t, \ldots, C_N\}$ which contains C_t and all its neighbors in $\mathcal{G}(\{C_t, \ldots, C_N\})$. (Note that this uniquely specifies $\mathcal{C}'_t$.)*

The nodes C_t are called the corners of the cubes *$\mathcal{C}'_t$, respectively. The dimension of the largest cube among $\mathcal{C}'_1, \ldots, \mathcal{C}'_N$ is called the* order of the corner-peeling plan *P. $\mathcal{C}$* can be *d*-corner-peeled *if there exists a corner-peeling plan of order d.*

$\mathcal{C}$ is called *shortest-path closed* if, for every pair of distinct concepts $C, C' \in \mathcal{C}$, $\mathcal{G}(\mathcal{C})$ contains a path of length $H(C, C')$ that connects C and C', where H denotes the Hamming distance. [17] showed the following:

1. If a maximum class $\mathcal{C}$ has a corner-peeling plan (5) of order $\mathrm{VCD}(\mathcal{C})$, then an unlabeled compression scheme for $\mathcal{C}$ is obtained by setting $r(C_t)$ equal to the set of colors in cube $\mathcal{C}'_t$ for $t = 1, \ldots, N$.

2. Every maximum class $\mathcal{C}$ can be VCD($\mathcal{C}$)-corner-peeled.

Although it was known before [12] that any maximum class has an unlabeled compression scheme, the scheme resulting from corner-peeling has some very special and nice features. We will see an application in Section 5, where we show that $\mathrm{RTD}(\mathcal{C}) = \mathrm{VCD}(\mathcal{C})$ for every maximum class $\mathcal{C}$.

3 Recursive Teaching and Query Learning

Kuhlmann proved the following result:

Lemma 7 (see [11]). *For every concept class* $\mathcal{C}$*:* $\mathrm{TS}_{min}(\mathcal{C}) \leq \mathrm{SDC}(\mathcal{C})$.

In view of (1), the monotonicity of LC-PARTIAL and SDC, the twofold monotonicity of M_{opt}, and in view of Lemma 3, we obtain:

Corollary 8. *For every concept class* $\mathcal{C}$*, the following holds:*

1. $\mathrm{RTD}(\mathcal{C}) \leq \mathrm{SDC}(\mathcal{C}) \leq \textit{LC-PARTIAL}(\mathcal{C}) \leq M_{opt}(\mathcal{C})$.
2. $\mathrm{RTD}^*(\mathcal{C}) \leq M_{opt}(\mathcal{C})$.

As demonstrated in [8], the model of self-directed learning is extremely powerful. According to Corollary 8, recursive teaching is an even more powerful model so that upper bounds on SDC apply to RTD as well, and lower bounds on RTD apply to SDC and LC-PARTIAL as well. The following result, which is partially known from [8, 22], illustrates this:

Corollary 9. 1. *If* $\mathrm{VCD}(\mathcal{C}) = 1$*, then* $\mathrm{RTD}(\mathcal{C}) = \mathrm{SDC}(\mathcal{C}) = 1$.
2. $\mathrm{RTD}(\textit{Monotone Monomials}) = \mathrm{SDC}(\textit{Monotone Monomials}) = 1$.
3. $\mathrm{RTD}(\textit{Monomials}) = \mathrm{SDC}(\textit{Monomials}) = 2$.
4. $\mathrm{RTD}(\mathrm{BOX}_n^d) = \mathrm{SDC}(\mathrm{BOX}_n^d) = 2$.
5. $\mathrm{RTD}(m\textit{-Term Monotone DNF}) \leq \mathrm{SDC}(m\textit{-Term Monotone DNF}) \leq m$.
6. $\mathrm{SDC}(m\textit{-Term Monotone DNF}) \geq \mathrm{RTD}(m\textit{-Term Monotone DNF}) \geq m$ *provided that the number of Boolean variables is at least* $m^2 + 1$.

Proof. All upper bounds on SDC are from [8] and, as mentioned above, they apply to RTD as well. Lower bound 1 on RTD (for concept classes with at most two distinct concepts) is trivial. RTD(Monomials) = 2 is shown in [22]. As a lower bound, this carries over to BOX_n^d which contains Monomials as a subclass. Thus the first five assertions are obvious from known results in combination with Corollary 8. As for the last assertion, we have to show that RTD(m-Term Monotone DNF) $\geq m$. To this end assume that there are $n \geq m^2 + 1$ Boolean variables. According to Lemma 4, it suffices to find a subclass $\mathcal{C}'$ of m-Term Monotone DNF such that $\mathrm{TS}_{min}(\mathcal{C}') \geq m$. Let $\mathcal{C}'$ be the class of all DNF formulas that contain precisely m pairwise variable-disjoint terms of length m each. Let F be an arbitrary but fixed formula in $\mathcal{C}'$. Without loss of generality, the teacher always picks either a minimal positive example (such that flipping any 1-bit to 0 turns it negative) or a maximal negative example (such that flipping any 0-bit to 1 turns it positive). By construction of $\mathcal{C}'$, the former example has precisely m ones (and reveals one of the m terms in F) and the latter example has precisely m zeros (and reveals one variable in each term). We may assume that the teacher consistently uses a numbering of the m terms from 1 to m and augments any 0-component (component i say) of a negative example by the number of the term that contains the corresponding Boolean variable (the term containing variable x_i). Since adding information is to the advantage of the learner, this will not corrupt the lower-bound argument. We can measure the knowledge that is still missing after having seen a collection of labeled instances by the following parameters:

– m', the number of still unknown terms
– $l_1, \dots, l_m$, where l_k is the number of still unknown variables in term k

The effect of a teaching set on these parameters is as follows: a positive example decrements m', and a negative example decrements some of $l_1, \dots, l_m$. Note that n was chosen sufficiently large[1] so that the formula F is not uniquely specified as long as none of the parameters has reached level 0. Since all parameters are initially of value m, the size of any teaching set for F must be at least m. □

In powerful learning models, techniques for proving lower bounds become an issue. One technique for proving a lower bound on RTD was applied already in the proof of Corollary 9: select a subclass $\mathcal{C}' \subseteq \mathcal{C}$ and derive a lower bound on $\mathrm{TS}_{min}(\mathcal{C}')$. We now turn to the question whether known lower bounds for LC-PARTIAL or SDC remain valid for RTD. [15] showed that LC-PARTIAL is lower-bounded by the logarithm of the length of a longest inclusion chain in $\mathcal{C}$. This bound does not even apply to SDC, which follows from an inspection of the class of half-intervals over domain $[n]$. The longest inclusion chain in this class, $\emptyset \subset \{1\} \subset \{1,2\} \subset \cdots \subset \{1,2,\dots,n\}$, has length $n+1$, but its self-directed learning complexity is 1. Theorem 8 in [3] implies that SDC is lower-bounded by $\log|\mathcal{C}|/\log|X|$ if $\mathrm{SDC}(\mathcal{C}) \geq 2$. A similar bound applies to RTD:

Lemma 10. *Suppose* $\mathrm{RTD}(\mathcal{C}) \geq 2$. *Then,* $\mathrm{RTD}(\mathcal{C}) \geq \frac{\log|\mathcal{C}|}{1+\log|X|}$ *and repetition-free teaching plans for* $\mathcal{C}$ *are of order at least* $\frac{\log|\mathcal{C}|}{\log|X|}$.

Proof. Let $k := \mathrm{RTD}(\mathcal{C})$, and let P be a teaching plan of order k for $\mathcal{C}$. Clearly, P contains $|\mathcal{C}|$ pairwise different teaching sets, and every teaching set is a labeled subset of X of size at most k. Thus,

$$|\mathcal{C}| \leq \sum_{i=1}^{k} \binom{|X|}{i} 2^i \leq 2^k \Phi_k(|X|) \leq (2|X|)^k . \tag{6}$$

Solving for k yields the desired lower bound on $\mathrm{RTD}(\mathcal{C})$. In a similar calculation for repetition-free teaching plans, a factor 2^i (and later 2^k) is missing in (6). □

[1] A slightly refined argument shows that requiring $n \geq (m-1)^2+1$ would be sufficient. But we made no serious attempt to make this assumption as weak as possible.

A subset $X' \subseteq X$ is called *$\mathcal{C}$-distinguishing* if, for each pair of distinct concepts $C, C' \in \mathcal{C}$, there is some $x \in X'$ such that $C(x) \neq C'(x)$. The matrix associated with a concept class $\mathcal{C}$ over domain X is given by $M(x, C) = C(x) \in \{0, 1\}$. We call two concept classes $\mathcal{C}, \mathcal{C}'$ equivalent if their matrices are equal up to permutation of rows or columns, and up to flipping all bits of a subset of the rows.[2] The following result characterizes the classes of recursive teaching dimension 1:

Theorem 11. *The following statements are equivalent:*

1. $\mathrm{SDC}(\mathcal{C}) = 1$.
2. $\mathrm{RTD}(\mathcal{C}) = 1$.
3. *There exists a $\mathcal{C}$-distinguishing set $X' \subseteq X$ such that $\mathcal{C}_{|X'}$ is equivalent to a concept class whose matrix M is of the form $M = [M'|\mathbf{0}]$ where M' is a lower-triangular square-matrix with ones on the main-diagonal and $\mathbf{0}$ denotes the all-zeros vector.*

Proof. 1 implies 2. If $\mathrm{SDC}(\mathcal{C}) = 1$, $\mathcal{C}$ contains at least two distinct concepts. Thus, $\mathrm{RTD}(\mathcal{C}) \geq 1$. According to Corollary 8, $\mathrm{RTD}(\mathcal{C}) \leq \mathrm{SDC}(\mathcal{C}) = 1$.

2 implies 3. Let P be a teaching plan of order 1 for $\mathcal{C}$, and let X' be the set of instances occurring in P (which clearly is $\mathcal{C}$-distinguishing). Let $(C_1, \{(x_1, b_1)\})$ be the first item of P. Let M be the matrix associated with $\mathcal{C}$ (up to equivalence). We make C_1 the first column and x_1 the first row of M. We may assume that $b_1 = 1$. (Otherwise flip all bits in row 1.) Since $\{(x_1, 1)\}$ is a teaching set for C_1, the first row of M is of the form $(1, 0, \ldots, 0)$. We may repeat this argument for every item in P so that the resulting matrix M is of the desired form. (The last zero-column represents the final concept in P with the empty teaching set.)

3 implies 1. Since X' is $\mathcal{C}$-distinguishing, exact identification of a concept $C \in \mathcal{C}$ is the same as exact identification of C restricted to X'. Let $x_1, \ldots, x_{N-1}$ denote the instances corresponding to the rows of M. Let $C_1, \ldots, C_N$ denote the concepts corresponding to the columns of M. A self-directed learner passes $(x_1, 0), (x_2, 0), \ldots$ to the oracle until it makes the first mistake (if any). If the first mistake (if any) happens for $(x_k, 0)$, the target concept must be C_k (because of the form of M). If no mistake has occurred on items $(x_1, 0), \ldots, (x_{N-1}, 0)$, there is only one possible target concept left, namely C_N. Thus the self-directed learner exactly identifies the target concept at the expense of at most one mistake. □

Note that concept classes of recursive teaching dimension 1 can have arbitrarily large VC-dimension. However, [11] presents a family $(\mathcal{C}_m)_{m \geq 1}$ of concept classes such that $\mathrm{VCD}(\mathcal{C}_m) = 2m$ but $\mathrm{RTD}(\mathcal{C}_m) \geq \mathrm{TD}_{min}(\mathcal{C}_m) = 3m$. This shows that RTD cannot generally be upper-bounded by the VC-dimension (but leaves open the possibility of an upper bound of the form $O(\mathrm{VCD}(\mathcal{C}))$).

As we have seen in this section, the gap between $\mathrm{SDC}(\mathcal{C})$ and LC-PARTIAL$(\mathcal{C})$ can be arbitrarily large (e.g., the class of half-intervals over domain $[n]$). We will see below, that a similar statement applies to $\mathrm{RTD}(\mathcal{C})$ and $\mathrm{SDC}(\mathcal{C})$ (despite of the fact that both measures assign value 1 to the same family of concept classes).

[2] Reasonable complexity measures (including RTD, SDC, VCD) are invariant under these operations.

4 Recursive Teaching and Intersection-Closed Classes

As shown by Kuhlmann [11], $\mathrm{TS}_{min}(\mathcal{C}) \le I(\mathcal{C})$ holds for every intersection-closed concept class $\mathcal{C}$. Kuhlmann's central argument (which occurred first in a proof of a related result in [8]) can be applied recursively so that the following is obtained:

Lemma 12. *For every intersection-closed class $\mathcal{C}$, $\mathrm{RTD}(\mathcal{C}) \le I(\mathcal{C})$.*

Proof. Let $k := I(\mathcal{C})$. We present a teaching plan for $\mathcal{C}$ of order at most k. Let $C_1, \ldots, C_N$ be the concepts in $\mathcal{C}$ in topological order such that $C_i \supset C_j$ implies $i < j$. It follows that, for every $i \in [N]$, C_i is an inclusion-maximal concept in $\mathcal{C}_i := \{C_i, \ldots, C_N\}$. Let S_i denote a minimal spanning set for C_i w.r.t. $\mathcal{C}$. Then:

– $|S_i| \le k$ and C_i is the unique minimal concept in $\mathcal{C}$ that contains S_i.
– As C_i is inclusion-maximal in $\mathcal{C}_i$, C_i is the only concept in $\mathcal{C}_i$ that contains S_i.

Thus $\{(x, 1) \mid x \in S_i\}$ is a teaching set of size at most k for C_i in $\mathcal{C}_i$. □

Since $I(\mathcal{C}) \le \mathrm{VCD}(\mathcal{C})$, we get

Corollary 13. *For every intersection-closed class $\mathcal{C}$, $\mathrm{RTD}(\mathcal{C}) \le \mathrm{VCD}(\mathcal{C})$.*

This implies $\mathrm{RTD}^*(\mathcal{C}) \le \mathrm{VCD}(\mathcal{C})$ for every intersection-closed class $\mathcal{C}$, since intersection-closedness is preserved when reducing a class $\mathcal{C}$ to $\mathcal{C}_{|X'}$ for $X' \subseteq X$.

For every fixed constant d (e.g., $d = 2$), [11] presents a family $(\mathcal{C}_m)_{m \ge 1}$ of intersection-closed concept classes such that the following holds:[3]

$$\forall m \ge 1 : \ \mathrm{VCD}(\mathcal{C}_m) = d \text{ and } \mathrm{SDC}(\mathcal{C}_m) \ge m\,. \tag{7}$$

This shows that $\mathrm{SDC}(\mathcal{C})$ can in general not be upper-bounded by $I(\mathcal{C})$ or $\mathrm{VCD}(\mathcal{C})$. It shows furthermore that the gap between $\mathrm{RTD}(\mathcal{C})$ and $\mathrm{SDC}(\mathcal{C})$ can be arbitrarily large (even for intersection-closed classes).

Lemma 12 generalizes to nested differences:

Theorem 14. *If $\mathcal{C}$ is intersection-closed then* $\mathrm{RTD}(\mathrm{DIFF}^{\le d}(\mathcal{C})) \le d \cdot I(\mathcal{C})$.

Proof. Any concept $C \in \mathrm{DIFF}^{\le d}(\mathcal{C})$ can be written in the form

$$C = C_1 \setminus \overbrace{(C_2 \setminus (\cdots (C_{d-1} \setminus C_d) \cdots))}^{=:D_1} \tag{8}$$

such that, for every j, $C_j \in \mathcal{C} \cup \{\emptyset\}$, $C_j \supseteq C_{j+1}$, and this inclusion is proper unless $C_j = \emptyset$. Let $D_j = C_{j+1} \setminus (C_{j+2} \setminus (\cdots (C_{d-1} \setminus C_d) \cdots))$. We may obviously assume that the representation (8) of C is *minimal* in the following sense:

$$\forall j = 1, \ldots, d : C_j = \langle C_j \setminus D_j \rangle_{\mathcal{C}} \tag{9}$$

We define a *lexicographic ordering*, $\sqsupset$, on concepts from $\mathrm{DIFF}^{\le d}(\mathcal{C})$ as follows. Let C be a concept with a minimal representation of the form (8), and let the

[3] A family satisfying (7) but *not* being intersection-closed was presented previously [3].

minimal representation of C' be given similarly in terms of C'_j, D'_j. Then, by definition, $C \sqsupset C'$ if $C_1 \supset C'_1$ or $C_1 = C'_1 \wedge D_1 \sqsupset D'_1$.

Let $k := I(\mathcal{C})$. We present a teaching plan of order at most dk for $\mathrm{DIFF}^{\le d}(\mathcal{C})$. Therein, the concepts are in lexicographic order so that, when teaching concept C with minimal representation (8), the concepts preceding C w.r.t. $\sqsupset$ are discarded already. A teaching set T for C is then obtained as follows:

– For every $j = 1, \ldots, d$, include in T a minimal spanning set for $C_j \setminus D_j$ w.r.t. $\mathcal{C}$. Augment its instances by label 1 if j is odd, and by label 0 otherwise.

By construction, C as given by (8) and (9) is the lexicographically smallest concept in $\mathrm{DIFF}^{\le d}(\mathcal{C})$ that is consistent with T. Since concepts being lexicographically larger than C are discarded already, T is a teaching set for C. □

Corollary 15. *Let $\mathcal{C}_1, \ldots, \mathcal{C}_r$ be intersection-closed classes over the domain X. Assume that the "universal concept" X belongs to each of these classes.*[4] *Then,*

$$\mathrm{RTD}\left(\mathrm{DIFF}^{\le d}(\mathcal{C}_1 \cup \cdots \cup \mathcal{C}_r)\right) \le d \cdot \sum_{i=1}^{r} I(\mathcal{C}_i)\,.$$

Proof. Consider the concept class $\mathcal{C} := \mathcal{C}_1 \wedge \cdots \wedge \mathcal{C}_r := \{C_1 \cap \cdots \cap C_r \mid C_i \in \mathcal{C}_i \text{ for } i = 1, \ldots, r\}$. According to [9], we have:

1. $\mathcal{C}_1 \cup \cdots \cup \mathcal{C}_r$ is a subclass of $\mathcal{C}$.
2. $\mathcal{C}$ is intersection-closed.
3. Let $C = C_1 \cap \cdots \cap C_r \in \mathcal{C}$. For all i, let S_i be a spanning set for C w.r.t. $\mathcal{C}_i$, i.e., $S_i \subseteq C$ and $\langle S_i \rangle_{\mathcal{C}_i} = \langle C \rangle_{\mathcal{C}_i}$. Then $S_1 \cup \cdots \cup S_r$ is a spanning set for C w.r.t. $\mathcal{C}$.

Thus $I(\mathcal{C}) \le I(\mathcal{C}_1) + \cdots + I(\mathcal{C}_r)$. The corollary follows from Theorem 14. □

5 Recursive Teaching Dimension and Maximum Classes

In this section, we show that the recursive teaching dimension coincides with the VC-dimension on the family of maximum classes. In a maximum class $\mathcal{C}$, every set of $k \le \mathrm{VCD}(\mathcal{C})$ instances is shattered, which implies $\mathrm{RTD}(\mathcal{C}) \ge \mathrm{TS}_{min}(\mathcal{C}) \ge \mathrm{VCD}(\mathcal{C})$. Thus, we can focus on the reverse direction and pursue the question whether $\mathrm{RTD}(\mathcal{C}) \le \mathrm{VCD}(\mathcal{C})$. We shall answer this question to the affirmative by establishing a connection between "teaching plans" and "corner-peeling plans".

We say that a corner-peeling plan (5) is *strong* if Condition 2 in Definition 6 is replaced as follows:

2'. For all $t = 1, \ldots, N$, $\mathcal{C}'_t$ is a cube in $\{C_t, \ldots, C_N\}$ which contains C_t and whose colors (augmented by their labels according to C_t) form a teaching set for $C_t \in \{C_t, \ldots, C_N\}$.

[4] This assumption is not restrictive: adding the universal concept to an intersection-closed class does not destroy the intersection-closedness.

We denote the set of colors of $\mathcal{C}'_t$ as X_t and its augmentation by labels according to C_t as S_t in what follows. The following result is obvious:

Lemma 16. *A strong corner-peeling plan of the form (5) induces a teaching plan of the form (2) of the same order.*

The following result justifies the attribute "strong" of corner-peeling plans:

Lemma 17. *Every strong corner-peeling plan is a corner-peeling plan.*

Proof. Assume that Condition 2 is violated. Then there is a color $x \in X \setminus X_t$ and a concept $C \in \{C_{t+1}, \ldots, C_N\}$ such that C coincides with C_t on all instances except x. But then C is consistent with set S_t so that S_t is *not* a teaching set for $C_t \in \{C_t, \ldots, C_N\}$, and Condition 2' is violated as well. □

Lemma 18. *Let $\mathcal{C}$ be a shortest-path closed concept class. Then, every corner-peeling plan for $\mathcal{C}$ is strong.*

Proof. Assume that Condition 2' is violated. Then some $C \in \{C_{t+1}, \ldots, C_N\}$ is consistent with S_t. Thus, the shortest path between C and C_t in $\mathcal{G}(\{C_t, \ldots, C_N\})$ does not enter the cube $\mathcal{C}'_t$. Hence there is a concept $C' \in \{C_{t+1}, \ldots, C_N\} \setminus \mathcal{C}'_t$ that is a neighbor of C_t in $\mathcal{G}(\{C_t, \ldots, C_N\})$, and Condition 2 is violated. □

As maximum classes are shortest-path closed [12], we obtain:

Corollary 19. *Every corner-peeling plan for a maximum class is strong, and therefore induces a teaching plan of the same order.*

Since [17] showed that every maximum class $\mathcal{C}$ can be $\mathrm{VCD}(\mathcal{C})$-corner-peeled, we may conclude that $\mathrm{RTD}(\mathcal{C}) \le \mathrm{VCD}(\mathcal{C})$. As mentioned above, $\mathrm{RTD}(\mathcal{C}) \ge \mathrm{VCD}(\mathcal{C})$ for every maximum class $\mathcal{C}$, which implies

Corollary 20. *For every maximum class $\mathcal{C}$,* $\mathrm{RTD}(\mathcal{C}) = \mathrm{VCD}(\mathcal{C})$.

The fact that, for every maximum class $\mathcal{C}$ and every $X' \subseteq X$, the class $\mathcal{C}_{|X'}$ is still maximum implies that $\mathrm{RTD}^*(\mathcal{C}) = \mathrm{VCD}(\mathcal{C})$ for every maximum class $\mathcal{C}$.

We close this section by establishing a connection between repetition-free teaching plans and representations having the acyclic non-clashing property:

Lemma 21. *Let $\mathcal{C}$ be an arbitrary concept class. Then the following holds:*

1. *Every repetition-free teaching plan (2) of order d for $\mathcal{C}$ induces a representation mapping r of order d for $\mathcal{C}$ given by $r(C_t) = X(S_t)$ for $t = 1, \ldots, N$. Moreover, r has the acyclic non-clashing property.*
2. *Every representation mapping r of order d for $\mathcal{C}$ that has the acyclic non-clashing property (4) induces a teaching plan (2) given by $S_t = \{(x, C_t(x)) \mid x \in r(C_t)\}$ for $t = 1, \ldots, N$. Moreover, this plan is repetition-free.*

Proof. 1. A clash between C_t and $C_{t'}$, $t < t'$, on $X(S_t)$ would contradict the fact that S_t is a teaching set for $C_t \in \{C_t, \ldots, C_N\}$.

2. Conversely, if $S_t = \{(x, C_t(x)) \mid x \in r(C_t)\}$ is not a teaching set for $C_t \in \{C_t, \ldots, C_N\}$, then there must be a clash on $X(S_t)$ between C_t and a concept from $\{C_{t+1}, \ldots, C_N\}$. Repetition-freeness is obvious since r is injective. $\square$

Corollary 22. *Let $\mathcal{C}$ be maximum of VC-dimension d. Then, there is a one-one mapping between repetition-free teaching plans of order d for $\mathcal{C}$ and unlabeled compression schemes with the acyclic non-clashing property.*

An inspection of [17] reveals that corner-peeling leads to an unlabeled compression scheme with the acyclic non-clashing property (again implying that $\mathrm{RTD}(\mathcal{C}) \le \mathrm{VCD}(\mathcal{C})$ for maximum classes $\mathcal{C}$). An inspection of [12] reveals that the unlabeled compression scheme obtained by the Tail Matching Algorithm has the acyclic non-clashing property too. Thus, this algorithm too can be used to generate a recursive teaching plan of order $\mathrm{VCD}(\mathcal{C})$ for any maximum class $\mathcal{C}$.

6 Conclusions

This paper relates the RTD, a recent teaching complexity notion, to classical learning complexity parameters. One of these parameters is SDC, the complexity of self-directed learning—the most information-efficient query model known to date. Our result lower-bounding the SDC by the RTD has implications for the analysis of information complexity in teaching and learning. In particular, every upper bound on SDC holds for RTD; every lower bound on RTD holds for SDC.

Another important parameter in our comparison is the VC-dimension. Although the VC-dimension can be arbitrarily large for classes of recursive teaching dimension 1 (see Theorem 11 and the remark thereafter) and arbitrarily smaller than SDC [3, 11], it does not generally lie between the two. However, while the SDC cannot be upper-bounded by any linear function of the VC-dimension, it is still open whether such a bound is possible for the RTD.

As a partial solution to this open question, we showed that the VC-dimension coincides with the RTD in the special case of maximum classes. Our results, and in particular the remarkable correspondence to unlabeled compression schemes, suggest that the RTD refers to a combinatorial structure that is of high relevance for the complexity of information-efficient learning and sample compression. Analyzing the question whether teaching plans defining the RTD can in general be used to construct compression schemes (and to bound their size) seems to be a promising step towards new insights into the theory of sample compression.

Acknowledgments

We would like to thank Malte Darnstädt and Michael Kallweit for helpful and inspiring discussions. Moreover, we would like to thank three anonymous referees for many helpful suggestions and for pointing out a gap in the proof of Theorem 14.

References

[1] Angluin, D.: Queries and concept learning. Mach. Learn. 2, 319–342 (1988)
[2] Balbach, F.: Measuring teachability using variants of the teaching dimension. Theoret. Comput. Sci. 397, 94–113 (2008)
[3] Ben-David, S., Eiron, N.: Self-directed learning and its relation to the VC-dimension and to teacher-directed learning. Mach. Learn. 33, 87–104 (1998)
[4] Ben-David, S., Litman, A.: Combinatorial variability of Vapnik-Chervonenkis classes with applications to sample compression schemes. Discrete Appl. Math. 86(1), 3–25 (1998)
[5] Floyd, S., Warmuth, M.: Sample compression, learnability, and the vapnik-chervonenkis dimension. Mach. Learn. 21(3), 269–304 (1995)
[6] Goldman, S., Kearns, M.: On the complexity of teaching. J. Comput. Syst. Sci. 50(1), 20–31 (1995)
[7] Goldman, S., Rivest, R., Schapire, R.: Learning binary relations and total orders. SIAM J. Comput. 22(5), 1006–1034 (1993)
[8] Goldman, S., Sloan, R.: The power of self-directed learning. Mach. Learn. 14(1), 271–294 (1994)
[9] Helmbold, D., Sloan, R., Warmuth, M.: Learning nested differences of intersection-closed concept classes. Mach. Learn. 5, 165–196 (1990)
[10] Jackson, J., Tomkins, A.: A computational model of teaching. In: 5th Annl. Workshop on Computational Learning Theory, pp. 319–326 (1992)
[11] Kuhlmann, C.: On teaching and learning intersection-closed concept classes. In: Fischer, P., Simon, H.U. (eds.) EuroCOLT 1999. LNCS (LNAI), vol. 1572, pp. 168–182. Springer, Heidelberg (1999)
[12] Kuzmin, D., Warmuth, M.: Unlabeled compression schemes for maximum classes. J. Mach. Learn. Research 8, 2047–2081 (2007)
[13] Littlestone, N.: Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. Mach. Learn. 2(4), 285–318 (1988)
[14] Littlestone, N., Warmuth, M.: Relating data compression and learnability. Technical report, UC Santa Cruz (1986)
[15] Maass, W., Turán, G.: Lower bound methods and separation results for on-line learning models. Mach. Learn. 9, 107–145 (1992)
[16] Natarajan, B.: On learning boolean functions. In: 19th Annl. Symp. Theory of Computing, pp. 296–304 (1987)
[17] Rubinstein, B., Rubinstein, J.: A geometric approach to sample compression (2009) (unpublished manuscript)
[18] Sauer, N.: On the density of families of sets. J. Comb. Theory, Ser. A 13(1), 145–147 (1972)
[19] Shinohara, A., Miyano, S.: Teachability in computational learning. New Generat. Comput. 8, 337–348 (1991)
[20] Vapnik, V., Chervonenkis, A.: On the uniform convergence of relative frequencies of events to their probabilities. Theor. Probability and Appl. 16, 264–280 (1971)
[21] Welzl, E.: Complete range spaces (1987) (unpublished notes)
[22] Zilles, S., Lange, S., Holte, R., Zinkevich, M.: Teaching dimensions based on cooperative learning. In: 21st Annl. Conf. Learning Theory, pp. 135–146 (2008)

Toward a Classification of Finite Partial-Monitoring Games

Gábor Bartók, Dávid Pál, and Csaba Szepesvári

Department of Computing Science, University of Alberta, Canada
{bartok,dpal,szepesva}@cs.ualberta.ca

Abstract. In a finite partial-monitoring game against Nature, the Learner repeatedly chooses one of finitely many actions, the Nature responds with one of finitely many outcomes, the Learner suffers a loss and receives feedback signal, both of which are fixed functions of the action and the outcome. The goal of the Learner is to minimize its total cumulative loss. We make progress towards classification of these games based on their minimax expected regret. Namely, we classify almost all games with two outcomes: We show that their minimax expected regret is either zero, $\widetilde{\Theta}(\sqrt{T})$, $\Theta(T^{2/3})$, or $\Theta(T)$ and we give a simple and efficiently computable classification of these four classes of games. Our hope is that the result can serve as a stepping stone toward classifying all finite partial-monitoring games.

1 Introduction

A full information matrix game is specified by a finite loss matrix, $L = (\ell_{ij})$, where $1 \le i \le N$ denotes the actions of the row player and $1 \le j \le M$ denotes the actions of the column player, while $\ell_{ij} \in [0,1]$ is the loss suffered by the row player when he chose action i and the opponent chose action j. In games against Nature, Nature plays the role of the column player. In these games, at the beginning of the game Nature chooses an arbitrary sequence of actions of length T, unknown to the row player (henceforth Learner). If the sequence was known, the Learner could select the action that gives rise to the smallest possible cumulated loss. The regret of the Learner is defined by his excess cumulated loss compared to the mentioned best possible cumulated loss. Generally, the regret grows with the time horizon T. If the growth is sublinear then in the long run the Learner can be said to play almost as well as if he knew Nature's sequence of actions in advance. In a *full information matrix game against Nature*, the Learner is told Nature's action after every round, so that he has a chance to make adjustments to what actions to play. The Learner in general needs to randomize to prevent being second-guessed. In this situation, it is known that the Learner can keep his expected regret, R_T, below $\sqrt{T \ln(N)/2}$, independently of M (cf. Chapter 4 and the references in the book by Lugosi and Cesa-Bianchi [1]).

When playing in a *partial-information matrix game*, the main topic of this article, Nature's actions can be masked. More precisely, at the beginning of the

© Springer-Verlag Berlin Heidelberg 2010

game the Learner is given a pair of $N \times M$ matrices, (L, H), where L is the loss matrix as before, while H is a matrix that specifies what information the Learner receives in each round. The elements of H belong to some alphabet, which, without the loss of generality (WLOG), can be assumed to be the set of natural numbers. The way the game is played is modified as follows: in any round, if i and j are the actions chosen by the Learner and Nature, respectively, then instead of j, the Learner receives H_{ij} only as the feedback. It is then the structure of H that determines how much information is revealed in each time step: assuming the learner selects i, H_{ij} may reveal the identity of j (i.e., if $H_{ij} \neq H_{ik}$, $1 \leq j < k \leq M$) or it may mask it completely (i.e., if $H_{i,\cdot} \equiv \text{const}$). The goal of the Learner is still to keep its regret small, but the game has now a new element: The learner might need to give up on using actions with small losses in favour of playing informative actions i.e., the *exploration vs. exploitation tradeoff* appears.

Let us now discuss some previous results and describe our contributions. A special case of partial-information games is when the Learner learns the loss of the action taken (i.e., when $H = L$), also known as the *bandit case*[1]. Then, the INF algorithm due to Audibert and Bubeck [2] is known to achieve a regret bound $O(\sqrt{NT})$. (The Exp3 algorithm due to Auer et al. [3] achieves the same bound up to logarithmic factors.) It is also known that this is the best possible bound [3].

Now, consider another special case: Imagine a 3×2 game (L, H), where the first action of the Learner gives full information about Nature's choice ($H_{11} \neq H_{12}$), but it has a high cost, independently of Nature's choice (say, $\ell_{11} = \ell_{12} = 1$), while the other two actions do not reveal any information about Nature's choice (i.e., $H_{i1} = H_{i2}$, $i = 2, 3$). Further, assume that the cost of action 2 is low if Nature chooses action 1 and the cost of action 3 is low if Nature chooses action 2, say, $\ell_{21} = \ell_{32} = 0$, $\ell_{22} = \ell_{31} = 1$:

$$L = \begin{pmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad H = \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

In this case, it is known that the regret growth-rate is bounded by $\Omega(T^{2/3})$ from below (cf. Theorem 5.1, [4]), showing that a game like this is intrinsically harder than a bandit problem. Further, it is known that if certain conditions hold then the regret of the "general forecaster for partial monitoring" is bounded by $O(T^{2/3})$ (cf. Theorem 3.1, [4]).

It is also clear that in certain cases the best possible regret grows linearly with T (i.e., when no information is received about Nature's actions), while in some other trivial cases the learner can achieve 0 regret.

Thus, we see, that the difficulty of a game depends on the structure of L and H. However, as of yet, it is unclear what determines this difficulty. In fact,

[1] The non-stochatic multi-armed bandit problem with losses constrained to a finite set is an example of a game satisfying $H = L$. However, this condition allows also other types of games where the Learner can recover the losses of actions not chosen.

discussing the rate of decay of the regret per time-step, Cesa-Bianchi et al. [4] note that *"It remains a challenging problem to characterize the class of problems that admit rates of convergence faster than $O(T^{-1/3})$"*.[2] This is exactly the question which motivated the research reported on in this paper. In particular, we wish to answer the following questions:

1. Given L, H, how difficult is the game $G = (L, H)$? That is, given G what is the growth-rate of *the minimax regret*,
$$\mathrm{R}_T(G) = \inf_{A \in \mathcal{A}} \sup_{E \in \mathcal{E}} \mathrm{R}_T(G, A, E) \, , \tag{1}$$
corresponding to G, where $\mathcal{A}$ is the class of randomized strategies for the learner, $\mathcal{E}$ is the class of Nature's strategies and $\mathrm{R}_T(G, A, E)$ denotes the expected regret up to time T when the Learner using strategy A is playing against Nature in game G and Nature uses strategy E.
2. Do there exist games where the exponent of the minimax regret rate is other than 0, 1/2, 2/3, and 1?
3. Does there exist a strategy (and what is it), which, when fed with $G = (L, H)$, achieves the minimax regret?

In this paper, we make some initial steps toward answering these questions. In particular, for games *when Nature has at most two actions*, apart from a set of games of measure zero, we give complete answer to the above questions.

In particular, we show that the answer to the second question above is negative: Only exponents $0, 1/2, 2/3$ and 1 can appear in the growth rate of the minimax regret. As far as the lower bounds are concerned, an exponent of 1/2 follows since non-trivial partial-monitoring games are clearly at least as difficult as full-information games and, for the latter games, as it is well known, a lower bound on the minimax regret with exponent 1/2 holds [5]. Thus, our first contribution is to show that if the exponent of the minimax regret rate is above 1/2 then it cannot be below 2/3. Precisely, we show that if, after sorting the nondominated Learner's actions according to their losses under either one of Nature's actions, there exist two consecutive actions under which Nature's two actions are indistinguishable (i.e., the actions are non-revealing) then Nature can force a high regret. Here, a Learner's action i is called *nondominated* if there exists a distribution over Nature's actions such that action i has the smallest average loss over that distribution. Otherwise it is called *dominated.* An action i is *non-revealing* if $H_{i1} = H_{i2}$, otherwise it is called *revealing.*

Our next contribution is that we give a strategy which, apart from these difficult games, achieves a regret growth rate with exponent 1/2. Here, the insight is that if at least one of any pair of consecutive nondominated actions is a revealing action then the Learner can gain enough information cheaply. Since Corollary 4.2 due to Cesa-Bianchi et al. [4] states that a regret with exponent 2/3 is achievable for all non-trivial partial-monitoring games, we basically get a complete classification of games with $M = 2$.

[2] Here we renamed their n to T to match our notation.

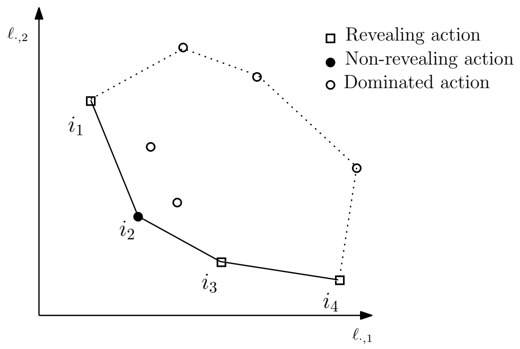


Fig. 1. The figure shows each action i as a point in $\mathbb{R}^2$ with coordinates $(\ell_{i,1}, \ell_{i,2})$. The solid line connects the chain of nondominated actions, which, by convention are ordered according to their loss for the first outcome.

2 Results

Consider a finite partial-monitoring game $G = (L, H)$ of size $N \times M$. The *regret* of the Learner playing sequence $1 \le I_t \le N$ against Nature playing sequence $1 \le J_t \le M$ is defined as

$$\widehat{\mathrm{R}}_T = \sum_{t=1}^{T} \ell_{I_t,J_t} - \min_{1 \le i \le M} \sum_{t=1}^{T} \ell_{i,J_t} . \tag{2}$$

The expected regret is defined by $\mathrm{R}_T = \mathbb{E}[\widehat{\mathrm{R}}_T]$. (Note that since the Learner can randomize, $\widehat{\mathrm{R}}_T$ is a random variable.) In what follows Nature's actions will also be called *outcomes*.

From now on we consider only the case when $M = 2$. Dominated, nondominated, revealing and non-revealing actions were introduced in the introduction. These concepts can be visualized by showing each action i as a point in $\mathbb{R}^2$ with coordinates $(\ell_{i,1}, \ell_{i,2})$. Then the points corresponding to the nondominated actions lie on the boundary of the convex hull of the set of all the actions. See Figure 1. Enumerating the nondominated actions in the counter-clockwise order along the boundary of the convex hull gives rise to a sequence $(i_1, i_2, \dots, i_K)$, which we call the *chain of nondominated actions*.

To avoid trivialities, WLOG we will assume that there are no *duplicate* actions, that is, two actions i, j, $i \ne j$, such that the i-th and the j-th rows of the loss matrix are the same. Clearly, if duplicate actions exist then at least one of them can be removed without changing the min-max expected regret: If both are either revealing or non-revealing, it does not matter which action is removed. Otherwise, we remove the non-revealing action.

To state the classification theorem, we introduce the following conditions.

Separation Condition. *A game G satisfies the* separation condition *if its chain of nondominated actions does* ***not*** *have a pair of consecutive actions*

i_j, i_{j+1} *such that both of them are non-revealing. The set of games satisfying this condition will be denoted by* $\mathcal{S}$.

Non-degeneracy Condition. *A game* G *satisfies the* non-degeneracy condition *if each of its nondominated actions is an extreme*[3] *point of the convex hull of all the actions.*

As we will soon see, the separation condition is the key to distinguish between "hard" and "easy" games. On the other hand, the non-degeneracy condition is merely a technical condition that we need in our proofs. The Lebesgue measure of the class of loss matrices it excludes is zero. We are now ready to state our main result.

Theorem 1 (Classification of two-outcome partial-monitoring games). *Let* $G = (L, H)$ *be a finite partial-monitoring game with two outcomes that satisfies the non-degeneracy condition. Let* K *be the number of nondominated actions in* G. *Let* $\mathcal{S}$ *be the set of games satisfying the separation condition. The min-max expected regret* $\mathrm{R}_T(G)$ *satisfies*[4]

$$\mathrm{R}_T(G) = \begin{cases} 0, & K = 1; & \text{(3a)} \\ \widetilde{\Theta}\left(\sqrt{T}\right), & K \geq 2, G \in \mathcal{S}; & \text{(3b)} \\ \Theta\left(T^{2/3}\right), & K \geq 2, G \notin \mathcal{S},\ \textit{G has a revealing action;} & \text{(3c)} \\ \Theta(T), & \textit{otherwise.} & \text{(3d)} \end{cases}$$

Cases (3a) and (3d) are trivial. The lower bound of case (3b) follows from the fact that even the full information case has expected regret $\Omega(\sqrt{T})$ [5]. The upper bound of case (3c) can be derived from a result of Cesa-Bianchi et al. [4]: Recall that the entries of H can be changed without changing the information revealed to the Learner as long as one does not change the pattern of which elements in a row are equal and different. Cesa-Bianchi et al. [4] show that if the entries of H can be chosen such that $\mathrm{rank}(H) = \mathrm{rank}\begin{pmatrix} H \\ L \end{pmatrix}$ then $O(T^{2/3})$ expected regret is achievable. This condition holds trivially for two-outcome games with at least one revealing action. It remains to prove the upper bound for (3b) and the lower bound for (3c). We prove these in the next sections.

3 Upper Bound

In this section we present our algorithm, AppleTree, for games satisfying the separation condition and the non-degeneracy condition, and prove that it achieves $\widetilde{O}(\sqrt{T})$ regret with high probability. (The choice of the name of the algorithm will be explained later.)

[3] An extreme point of a convex set is a point which is not a non-trivial convex combination of two different points of the set. In our case, the set is a convex polygon and its extreme points are precisely its vertices.

[4] Here, $a_n = \widetilde{\Theta}(b_n)$ stands for $a_n = \Omega(b_n)$ and $a_n = \widetilde{O}(b_n)$, where $\widetilde{O}(\cdot)$ hides polylogarithmic terms.

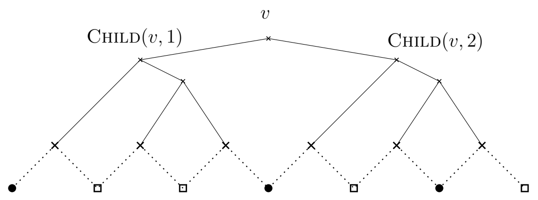


Fig. 2. The binary tree built by the algorithm. The leaf nodes represent neighboring action pairs.

3.1 Algorithm

In the first step of the algorithm we can purify the game by first removing the dominated actions and then the duplicates as mentioned beforehand.

The idea of the algorithm is to recursively split the game until we arrive at games with two actions only. Now, if one has only two actions in a partial-information game, the game must be either a full-information game (if both actions are revealing) or an instance of a one-armed bandit (with one action revealing the outcome, the other revealing no information).

To see why this latter case corresponds to one-armed bandits assume WLOG that the first action is the revealing action. Now, it is easy to see that the regret of a sequence of actions in a game does not change if the loss matrix is changed by subtracting the same number from a column.[5] By subtracting $\ell_{2,1}$ from the first and $\ell_{2,2}$ from the second column we thus get the equivalent game where the second row of the loss matrix is zero. In this game, the Learner knows the loss of the second action independently of the outcome, while, since the first action is revealing, he learns the loss of the first action in any round when that action is played, which is exactly what one has in a one-armed bandit game. Since a one-armed bandit is a special form of a two-armed bandit, one can use Exp3.P due to Auer et al. [3] to achieve the $\widetilde{O}(\sqrt{T})$ regret[6].

Now, if there are more than two actions in the game, then the game is split, putting the first half of the actions into the first and the second half into the second subgame, with a *single common shared action.* Here the actions are ordered according to their losses corresponding to the *first* outcome. This is continued until the split results into games with two actions only. The recursive splitting of the game results in a binary tree (see Figure 2). The idea of the strategy played at an internal node of the tree is as follows: An outcome sequence of length T determines the frequency ρ_T of outcome 2. If this frequency is small, the optimal action is one of the actions of G_1, the first subgame (simply because then the frequency of outcome 1 is high and G_1 contains the actions with the smallest loss

[5] As a result, for any algorithm, if R_T is its regret at time T when measured in the game with the modified loss matrix, the algorithm's "true" regret will also be R_T (i.e., the algorithm's regret when measured in the original, unmodified game). Piccolboni and Schindelhauer [6] exploit this idea, too.

[6] Apparently, this is a new result for this kind of game, also known as apple tasting.

for the first outcome). Conversely, if this frequency is large, the optimal action is one of the actions of G_2. In some intermediate range, the optimal action is the action shared between the subgames. Let the boundaries of this range be $\rho_1^* < \rho_2^*$ (ρ_1^* is thus the solution to $(1-\rho)\ell_{1,s-1} + \rho\ell_{2,s-1} = (1-\rho)\ell_{1,s} + \rho\ell_{2,s}$ and ρ_2^* is the solution to $(1-\rho)\ell_{1,s+1} + \rho\ell_{2,s+1} = (1-\rho)\ell_{1,s} + \rho\ell_{2,s}$, where $s = \lceil K/2 \rceil$ is the index of the action shared between the two subgames.)

If we knew ρ_T, a good solution would be to play a strategy where the actions are restricted to that of either game G_1 or G_2, depending on whether $\rho_T \leq \rho_1^*$ or $\rho_T \geq \rho_2^*$. (When $\rho_1^* \leq \rho_T \leq \rho_2^*$ then it does not matter which action-set we restrict the play to, since the optimal action in this case is included in both sets.) There are two difficulties. First, since the outcome sequence is not known in advance, the best we can hope for is to know the running frequencies $\rho_t = \frac{1}{t}\sum_{s=1}^{t} \mathbb{I}(J_s = 2)$. However, since the game is a partial-information game, the outcomes are not revealed in all time steps, hence, even ρ_t is inaccessible. Nevertheless, for simplicity, assume that ρ_t was available. Then one idea would be to play a strategy restricted to the actions of either game G_1 or G_2 as long as ρ_t stays below ρ_1^* or above ρ_2^*. Further, when ρ_t becomes larger than ρ_2^* while previously the strategy played the action of G_1 then we have to switch to the game G_2. In this case, we start a fresh copy of a strategy playing in G_2. The same happens when a switch from G_2 to game G_1 is necessary. The resets are necessary because at the leaves we play according to strategies that use weights that depend on the cumulated losses of the actions *exponentially*. To see an example when without resets the algorithm fails to achieve a small regret consider the case when there are 3 actions, the middle one being revealing. Assume that during the first $T/2$ time steps the frequency of outcome 2 oscillates between the two boundaries so that the algorithm switches constantly back and forth between the games G_1 and G_2. Assume further that in the second half of the game, the outcome is always 2. This way the optimal action will be 3. Nevertheless, up to time step $T/2$, the player of G_2 will only see outcome 1 and thus will think that action 2 is the optimal action. In the second half of the game, he will not have enough time to recover and will play action 2 for too long. Resetting the algorithms of the subgames avoids this behavior.

If the number of switches was large, the repeated resetting of the strategies could be equally problematic. Luckily this cannot happen, hence the resetting does minimal harm. We will in fact show that this generalizes to the case even when ρ_t is estimated based on partial feedback (see Lemma 3).

Let us now turn to how ρ_t is estimated. In any round, the algorithm receives feedback $h_t \in \{1, 2, *\}$: if a revealing action is played in the round, $h_t = J_t \in \{1, 2\}$, otherwise $h_t = *$. If the algorithm choosing the actions decides with probability $p_t \in (0, 1]$ to play a revealing action (p_t can depend on the history $\mathcal{H}_t$) then $\mathbb{I}(h_t = 2)/p_t$ is a simple unbiased estimate of $\mathbb{I}(J_t = 2)$ (in fact, $\mathbb{E}\left[\mathbb{I}(h_t = 2)/p_t | \mathcal{H}_t\right] = \mathbb{I}(J_t = 2)$). As long as p_t does not drop to a too low value, $\hat{\rho}_t = \frac{1}{t}\sum_{s=1}^{t} \frac{\mathbb{I}(h_t=2)}{p_t}$ will be a relatively reliable estimate of ρ_t (see Lemma 4).

```
function MAIN($G, T, \delta$)
Input: $G = (L, H)$ is a game, $T$ is a horizon, $0 < \delta < 1$ is a confidence parameter
 1: $G \leftarrow$ PURIFY($G$)
 2: BUILDTREE(root, $G, \delta$)
 3: for $t \leftarrow 1$ to $T$ do
 4:     PLAY(root)
 5: end for
```

Fig. 3. The main entry point of the AppleTree algorithm

```
function INITETA($G, T$)
Input: $G$ is a game, $T$ is a horizon
 1: if ISREVEALING($G$, 2) then
 2:     $\eta(v) \leftarrow \sqrt{8 \ln 2 / T}$
 3: else
 4:     $\eta(v) \leftarrow \gamma(v)/4$
 5: end if
```

Fig. 4. The initialization routine InitEta

```
function BUILDTREE($v, G, \delta$)
Input: $G = (L, H)$ is a game, $v$ is a tree node
 1: if NUMOFACTIONS($G$) = 2 then
 2:     if not ISREVEALING($G$, 1) then
 3:         $G \leftarrow$ SWAPACTIONS($G$)
 4:     end if
 5:     $w_i(v) \leftarrow 1/2$, $i = 1, 2$
 6:     $\beta(v) \leftarrow \sqrt{\ln(2/\delta)/(2T)}$
 7:     $\gamma(v) \leftarrow 8\beta(v)/(3 + \beta(v))$
 8:     INITETA($G, T$)
 9: else
10:     $(G_1, G_2) \leftarrow$ SPLITGAME($G$)
11:     BUILDTREE(CHILD($v$, 1), $G_1, \delta/(4T)$ )
12:     BUILDTREE(CHILD($v$, 2), $G_2, \delta/(4T)$ )
13:     $g(v) \leftarrow 1$, $\hat{\rho}(v) \leftarrow 0$, $t(v) \leftarrow 1$
14:     $(\rho'_1(v), \rho'_2(v)) \leftarrow$ BOUNDARIES($G$)
15: end if
16: $G(v) \leftarrow G$
```

Fig. 5. The tree building procedure

However reliable this estimate is, it can still differ from ρ_t. For this reason, we push the boundaries determining game switches towards each other:

$$\rho'_1 = \frac{2\rho^*_1 + \rho^*_2}{3}, \quad \rho'_2 = \frac{\rho^*_1 + 2\rho^*_2}{3}. \tag{4}$$

We call the resulting algorithm AppleTree, because the elementary partial-information 2-action games in the bottom essentially correspond to instances of the apple tasting problem (see Example 2.3 of [4]). The algorithm's main entry point is shown on Figure 3. Its inputs are the game $G = (L, H)$, the time horizon and a confidence parameter $0 < \delta < 1$. The algorithm first eliminates the dominated and duplicate actions. This is followed by building a tree, which is used to store variables necessary to play in the subgames (Figure 5): If the number of actions is 2, the procedure initializes various parameters that are used either by a bandit algorithm (based on Exp3.P [3]), or by the exponentially weighted average algorithm (EWA) [5]. In the other case, it calls itself recursively on the splitted subgames and with an appropriately decreased confidence parameter.

The main worker routine is called Play. This is again a recursive function (see Figure 6). The special case when the number of actions is two is handled in routine PlayAtLeaf, which will be discussed later. When the number of actions is larger, the algorithm recurses to play in the subgame that was remembered as the game to be preferred from the last round and then updates its estimate of the frequency of outcome 2 based on the information received. When this estimate changes so that a switch of the current preferred game is necessary,

```
function PLAY($v$)
Input: $v$ is a tree node
1: if NUMOFACTIONS($G(v)$) $= 2$ then
2:     $(p, h) \leftarrow$ PLAYATLEAF($v$)
3: else
4:     $(p, h) \leftarrow$ PLAY(CHILD($v, g(v)$))
5:     $\hat{\rho}(v) \leftarrow (1 - \frac{1}{t(v)})\hat{\rho}(v) + \frac{1}{t(v)} \frac{\mathbb{I}(h=2)}{p}$
6:     if $g(v) = 2$ and $\hat{\rho}(v) < \rho'_1(v)$ then
7:         RESET(CHILD($v, 1$)); $g(v) \leftarrow 1$
8:     else if $g(v) = 1$ and $\hat{\rho}(v) > \rho'_2(v)$ then
9:         RESET(CHILD($v, 2$)); $g(v) \leftarrow 2$
10:    end if
11:    $t(v) \leftarrow t(v) + 1$
12: end if
13: return $(p, h)$
```

Fig. 6. The recursive function PLAY

```
function RESET($v$)
Input: $v$ is a tree node
1: if NUMOFACTIONS($G(v)$) $= 2$ then
2:     $w_i(v) \leftarrow 1/2$, $i \leftarrow 1, 2$
3: else
4:     $g(v) \leftarrow 1$, $\hat{\rho}(v) \leftarrow 0$, $t(v) \leftarrow 1$
5:     RESET(CHILD($v, 1$))
6: end if
```

Fig. 7. Function RESET

the algorithm resets the algorithms in the subtree corresponding to the game switched to, and changes the variable storing the index of the preferred game. The RESET function used for this purpose, shown on Figure 7, is also recursive.

At the leaves, when there are only two actions, either EWA or Exp3.P is used. These algorithms are used with their standard optimized parameters (see Corollary 4.2 for the tuning of EWA, and Theorem 6.10 for the tuning of Exp3.P, both from the book of Lugosi and Cesa-Bianchi [1]). For completeness, their pseudocodes are shown in Figures 8–9. Note that with Exp3.P (lines 6–14) we use the loss matrix transformation described earlier, hence the loss matrix has zero entries for the second (non-revealing) action, while the entry for action 1 and outcome j is $\ell_{1,j}(v) - \ell_{2,j}(v)$. Here $\ell_{i,j}(v)$ stands for the loss of action i and outcome j in the game $G(v)$ that is stored at node v.

3.2 Proof of the Upper Bound

Theorem 2. *Assume $G = (L, H)$ satisfies the separation condition and the non-degeneracy condition and $\ell_{i,j} \leq 1$. Denote by $\widehat{\mathrm{R}}_T$ the regret of Algorithm* APPLE-TREE *up to time step T. There exist constants c, p such that for any $0 < \delta < 1$ and $T \in \mathbb{N}$, the algorithm with input G, T, δ achieves $\mathbb{P}\left(\widehat{\mathrm{R}}_T \leq c\sqrt{T}\ln^p(2T/\delta)\right) \geq 1 - \delta$.*

Throughout the proof we will analyze the algorithm's behavior at the root node. We will use time indices as follows. Let us define the filtration $\{\mathcal{F}_t = \sigma(I_1, \ldots, I_t)\}_t$, where I_t is the action the algorithm plays at time step t. To any variable $x(v)$ used by the algorithm, we denote by $x_t(v)$ the value of $x(v)$ that is measurable with respect to $\mathcal{F}_t$, but not measurable with respect to $\mathcal{F}_{t-1}$. From now on we abbreviate $x_t(\text{root})$ by x_t. We start with two lemmas. The first lemma shows that the number of switches the algorithm makes is small.

```
function PlayAtLeaf($v$)
Input: $v$ is a tree node
 1: if RevealingActionNumber($G(v)$) = 2 then          ▷ Full information case
 2:     $(p, h) \leftarrow$ Ewa($v$)
 3: else                                               ▷ Partial information case
 4:     $p \leftarrow (1-\gamma(v))\frac{w_1(v)}{w_1(v)+w_2(v)} + \gamma(v)/2$
 5:     $U \sim \mathcal{U}_{[0,1)}$                    ▷ $U$ is uniform in $[0,1)$
 6:     if $U < p$ then                                ▷ Play revealing action
 7:         $h \leftarrow$ CHOOSE(1)                    ▷ $h \in \{1,2\}$
 8:         $L_1 \leftarrow (\ell_{1,h}(v) - \ell_{2,h}(v) + \beta(v))/p$
 9:         $L_2 \leftarrow \beta(v)/(1-p)$
10:         $w_1(v) \leftarrow w_1(v)\exp(-\eta(v)L_1)$
11:         $w_2(v) \leftarrow w_2(v)\exp(-\eta(v)L_2)$
12:     else
13:         $h \leftarrow$ CHOOSE(2)                    ▷ here $h = *$
14:     end if
15: end if
16: return $(p, h)$
```

Fig. 8. Function PlayAtLeaf

```
function Ewa($v$)
Input: $v$ is a tree node
 1: $p \leftarrow \frac{w_1(v)}{w_1(v)+w_2(v)}$
 2: $U \sim \mathcal{U}_{[0,1)}$   ▷ $U$ is uniform in $[0,1)$
 3: if $U < p$ then
 4:     $I \leftarrow 1$
 5: else
 6:     $I \leftarrow 2$
 7: end if
 8: $h \leftarrow$ CHOOSE($I$)          ▷ $h \in \{1,2\}$
 9: $w_1(v) \leftarrow w_1(v)\exp(-\eta(v)\ell_{1,h}(v))$
10: $w_2(v) \leftarrow w_2(v)\exp(-\eta(v)\ell_{2,h}(v))$
11: return $(p, h)$
```

Fig. 9. Function Ewa

Lemma 3. *Let S be the number of times* AppleTree *calls* Reset *at the root node. Then there exists a universal constant c^* such that $S \le \frac{c^* \ln T}{\Delta}$, where $\Delta = \rho'_2 - \rho'_1$, ρ'_1 and ρ'_2 given by (4).*

Note that here we use the non-degeneracy condition to ensure that $\Delta > 0$.

Proof. Let s be the number of times the algorithm switches from G_2 to G_1. Let $t_1 < \ldots < t_s$ be the time steps when $\hat\rho_t$ becomes smaller than ρ'_1. Similarly, let $t'_1 < \ldots < t'_{s+\xi}$, ($\xi \in \{0,1\}$) be the time steps when $\hat\rho_t$ becomes greater than ρ'_2. Note that for all $1 \le j < s$, $t'_j < t_j < t'_{j+1}$. The number of times the algorithm resets is at most $2s+1$. For any $1 \le j \le s$, $\hat\rho_{t'_j} > \rho'_2$ and $\hat\rho_{t_j} < \rho'_1$. According to the update rule we have for any t that

$$\hat\rho_t = \left(1 - \frac{1}{t}\right)\hat\rho_{t-1} + \frac{1}{t}\cdot\frac{\mathbb{I}(J_t = 2)}{p_t} \ge \frac{t-1}{t}\hat\rho_{t-1} = \hat\rho_{t-1} - \frac{1}{t}\hat\rho_{t-1}$$

and hence $\hat\rho_{t-1} - \hat\rho_t \le \frac{1}{t}$. Summing this inequality for all $t'_j + 1 \le t \le t_j$ we get $\Delta \le \hat\rho_{t'_j} - \hat\rho_{t_j} \le \sum_{t=t'_j}^{t_j-1}\frac{1}{t} = O\left(\ln\frac{t_j}{t'_j}\right)$, using that $\Delta = \rho'_2 - \rho'_1$. Thus, there exists $c^* > 0$ such that for all $1 < j \le s$

$$\frac{1}{c^*}\Delta \le \ln\frac{t_j}{t'_j} \le \ln\frac{t_j}{t_{j-1}}\ . \tag{5}$$

Adding (5) for $1 < j \le s$ we get $(s-1)\frac{1}{c^*}\Delta \le \ln\frac{t_s}{t_1} \le \ln T$, which yields the desired statement. □

The next lemma shows that the estimate of the relative frequency of outcome 2 is not far away from its true value.

Lemma 4. *Let* $c = \frac{8}{3\Delta^2}$. *Then for any* $0 < \delta < 1$, *with probability at least* $1-\delta$, *for all* $t \geq c\sqrt{T}\ln(2T/\delta)$, $|\hat{\rho}_t - \rho_t| \leq \Delta$.

Proof. Using *Bernstein's inequality for martingales* (see Lemma A.8 in Lugosi and Cesa-Bianchi [1]) and the fact that, due to the construction of the algorithm, the probability p_t of playing a revealing action at time step t is always greater than $1/\sqrt{T}$, we get that for any t, $\Pr(|\hat{\rho}_t - \rho_t| > \Delta) \leq 2\exp\left(-\frac{3\Delta^2 t}{8\sqrt{T}}\right)$. Reordering the inequality and applying the union bound for all $1 \leq t \leq T$ we get the result. □

Proof. of Theorem 2. To prove that the algorithm achieves the desired regret bound we use induction on the depth of the tree, d. If $d = 1$, AppleTree plays either EWA or Exp3.P. EWA is known to satisfy Theorem 2, and, as we discussed earlier, Exp3.P achieves $O(\sqrt{T}\ln T/\delta)$ regret as well. As the induction hypothesis we assume that Theorem 2 is true for any T and any game such that the tree built by the algorithm has depth $d' < d$.

Let $Q_1 = \{1, \ldots, \lceil K/2 \rceil\}$, $Q_2 = \{\lceil K/2 \rceil, \ldots, K\}$ be the set of actions associated with the subgames in the root[7]. Furthermore, let us define the following values: Let $T_0^0 = 1$, let T_i^0 be the first time step t after T_{i-1}^0 such that $g_t \neq g_{t-1}$. In other words, T_i^0 are the time steps when the algorithm switches between the subgames. Finally, let $T_i = \min(T_i^0, T+1)$. From Lemma 3 we know that $T_{S_{max}+1} = T + 1$, where $S_{max} = \frac{c^* \ln T}{\Delta}$. It is easy to see that T_i are stopping times for any $i \geq 1$.

WLOG, from now on we will assume that the optimal action $i^* \in Q_1$. If $i^* = \lceil K/2 \rceil$ then, since it is contained in both subgames, the bound trivially follows from the induction hypothesis and Lemma 3. In the rest of the proof we assume $i^* < K/2$.

Let $S = \max\{i \geq 1 \mid T_i^0 \leq T\}$ the number of switches and $\mathcal{B}$ be the event that for all $t \geq c\sqrt{T}\ln(4T/\delta)$, $|\hat{\rho}_t - \rho_t| \leq \Delta$. We know from Lemma 4 that $\mathbb{P}(\mathcal{B}) \geq 1-\delta/2$. On $\mathcal{B}$ we have that $|\hat{\rho}_T - \rho_T| \leq \Delta$, and thus, using that $i^* < K/2$, $\rho_T \leq \rho_1^*$. This implies that in the last phase the algorithm plays on G_1. It is also easy to see that before the last switch, at time step $T_S - 1$, $\hat{\rho}$ is between ρ_1^* and ρ_2^*, if T_S is large enough. Thus, up to time step $T_S - 1$, the optimal action is $\lceil K/2 \rceil$, the one that is shared by the two subgames. This implies that $\sum_{t=1}^{T_S-1} \ell_{i^*,J_t} - \ell_{\lceil K/2 \rceil,J_t} \geq 0$. On the other hand, if $T_S \leq c\sqrt{T}\ln(4T/\delta)$ then

$$\sum_{t=1}^{T_S-1} \ell_{i^*,J_t} - \ell_{\lceil K/2 \rceil,J_t} \geq -c\sqrt{T}\ln(4T/\delta) \,.$$

[7] Recall that the actions are ordered with respect to $\ell_{\cdot,1}$.

Thus, we have

$$\begin{aligned}
\widehat{\mathrm{R}}_T &= \sum_{t=1}^{T} \ell_{I_t,J_t} - \ell_{i^*,J_t} \\
&= \sum_{t=1}^{T_S-1} (\ell_{I_t,J_t} - \ell_{i^*,J_t}) + \sum_{t=T_S}^{T} (\ell_{I_t,J_t} - \ell_{i^*,J_t}) \\
&\le \mathbb{I}(\mathcal{B}) \left(\sum_{t=1}^{T_S-1} \left(\ell_{I_t,J_t} - \ell_{\lceil K/2\rceil,J_t}\right) + \sum_{t=T_S}^{T} (\ell_{I_t,J_t} - \ell_{i^*,J_t}) \right) \\
&\quad + \underbrace{c\sqrt{T}\ln(4T/\delta) + (\mathbb{I}(\mathcal{B}^c))\,T}_{D} \\
&\le D + \mathbb{I}(\mathcal{B}) \sum_{r=1}^{S_{max}} \max_{i\in Q_{\pi(r)}} \sum_{t=T_{r-1}}^{T_r-1} (\ell_{I_t,J_t} - \ell_{i,J_t}) \\
&= D + \mathbb{I}(\mathcal{B}) \sum_{r=1}^{S_{max}} \max_{i\in Q_{\pi(r)}} \sum_{m=1}^{T} \mathbb{I}(T_r - T_{r-1} = m) \sum_{t=T_{r-1}}^{T_{r-1}+m-1} (\ell_{I_t,J_t} - \ell_{i,J_t}) \ ,
\end{aligned}$$

where $\pi(r)$ is 1 if r is odd and 2 if r is even. Note that for the last line of the above inequality chain to be well defined, we need outcome sequences of length at most $2T$. It makes us no harm to assume that for all $T < t \le 2T$, say, $J_t = 1$.

Recall that the strategies that play in the subgames are reset after the switches. Hence, the sum $\widehat{\mathrm{R}}_m^{(r)} = \sum_{t=T_{r-1}}^{T_{r-1}+m-1} (\ell_{I_t,J_t} - \ell_{i,J_t})$ is the regret of the algorithm if it is used in the subgame $G_{\pi(r)}$ for $m \le T$ steps. Then, exploiting that T_r are stopping times, we can use the induction hypothesis to bound $\widehat{\mathrm{R}}_m^{(r)}$. In particular, let $\mathcal{C}$ be the event that for all $m \le T$ the sum is less than $c\sqrt{T}\ln^p(2T^2/\delta)$. Since the root node calls its children with confidence parameter $\delta/(2T)$, we have that $\mathbb{P}(\mathcal{C}^c) \le \delta/2$. In summary,

$$\begin{aligned}
\widehat{\mathrm{R}}_T &\le D + \mathbb{I}(\mathcal{C}^c)T + \mathbb{I}(\mathcal{B})\mathbb{I}(\mathcal{C})S_{max}c\sqrt{T}\ln^p 2T^2/\delta \\
&\le \mathbb{I}(\mathcal{B}^c \cup \mathcal{C}^c)T + c\sqrt{T}\ln(4T/\delta) + \mathbb{I}(\mathcal{B})\mathbb{I}(\mathcal{C})\frac{c^*\ln T}{\varDelta}c\sqrt{T}\ln^p 2T^2/\delta.
\end{aligned}$$

Thus, on $\mathcal{B}\cap\mathcal{C}$, $\widehat{\mathrm{R}}_T \le \frac{2^p cc^*}{\varDelta}\sqrt{T}\ln^{p+1}(2T/\delta)$, which, together with $\mathbb{P}(\mathcal{B}^c\cup\mathcal{C}^c) \le \delta$ concludes the proof. □

Remark. The above theorem proves a high probability bound on the regret. We can get a bound on the expected regret if we set δ to $1/T$. Also note that the bound given by the induction grows in the number of nondominated actions as $O(K^{\log_2 K})$.

4 Lower Bound

In this section we present a lower bound for the expected regret in the case when the separation condition does not hold.

Theorem 5. *If G satisfies the non-degeneracy condition and the separation condition does **not** hold then there exists a constant C such that for any algorithm A and time horizon T there exists a sequence of outcomes such that the expected regret* $\mathrm{R}_T(A)$ *of the algorithm satisfies* $\mathrm{R}_T(A) \geq CT^{2/3}$.

Proof. We follow the steps of the lower bound proof for the label efficient prediction from Cesa-Bianchi et al. [4] with a few changes. The most important change, as we will see, is the choice of the models we randomize over.

We can assume WLOG that actions 1 and 2 are the two consecutive non-dominated non-revealing actions, while all the other actions are revealing and $(\ell_{1,1}, \ell_{1,2}) = (0, \alpha)$, $(\ell_{2,1}, \ell_{2,2}) = (1-\alpha, 0)$ with some $\alpha \in [0,1]$. That this can be assumed follows by scaling and a reduction similar to the one we used in Section 3.1. Using the non-degeneracy condition and that actions 1 and 2 are consecutive, we get that for all $i \geq 3$, there exists some $\lambda_i \in \mathbb{R}$ such that

$$\begin{aligned} \ell_{i,1} &> \lambda_i \ell_{1,1} + (1-\lambda_i)\ell_{2,1} = (1-\lambda_i)(1-\alpha) \,, \\ \ell_{i,2} &> \lambda_i \ell_{1,2} + (1-\lambda_i)\ell_{2,2} = \lambda_i \alpha \,. \end{aligned} \tag{6}$$

We denote $\lambda_{min} = \min_{i\geq 3} \lambda_i$, $\lambda_{max} = \max_{i \geq 3} \lambda_i$ and $\lambda^* = \lambda_{max} - \lambda_{min}$.

We construct random outcome sequences as follows. We define two models for generating outcome sequences. We use $p_i(\cdot)$ and $\mathbb{E}_i[\cdot]$ to denote probability mass function and expectation given model $i \in \{1,2\}$, respectively. In model 1 the outcomes are i.i.d. random variables with $p_1(1) = \alpha + \epsilon$ whereas in model 2, $p_2(1) = \alpha - \epsilon$ with $\epsilon < 1$ to be chosen later. Note that, if ϵ is small enough then only actions 1 and 2 can be optimal. Namely, action i is optimal in model i.

Let $h_t \in \{*, 1, 2\}$ denote the observation of the algorithm at time step t, and let h^t denote the observation sequence $(h_1, \ldots, h_t)$. Let $A_t(h^{t-1})$ denote the choice of the algorithm[8] at time step t, given the history of observations h^{t-1}. Let $N_i^j = \mathbb{E}_j[\sum_{t=1}^T \mathbb{I}(I_t = i)]$, that is, the expected number of times action i is played up to time step T, given model j. Finally, let $N_{\geq 3}^j = \sum_{i \geq 3} N_i^j$.

Let $D(p||q)$ be the KL divergence of Bernoulli distributions with parameters p and q. We need the following technical lemma.

Lemma 6. *Let $0 < \epsilon < \alpha$ be such that $\alpha + \epsilon < 1$. Then* $D(\alpha - \epsilon || \alpha + \epsilon) = \frac{2\epsilon^2}{\alpha(1-\alpha)} + O\left(\epsilon^3\right)$.

Proof. The result follows from the definition of KL divergence and the second order Taylor expansion of $\ln(1+x)$. □

The next lemma states that the expected number of times actions 1 and 2 are played by A does not change too much if we change the model:

Lemma 7. *There exists a constant c (depending on α only) such that*

$$N_2^1 \geq N_2^2 - cT\epsilon\sqrt{N_{\geq 3}^2} \qquad \textit{and} \qquad N_1^2 \geq N_1^1 - cT\epsilon\sqrt{N_{\geq 3}^1} \,.$$

[8] Conditioning on the internal randomization of A if necessary, we can assume WLOG that algorithm A is deterministic.

Proof. We only prove the first inequality, the other one is symmetric. We have

$$\begin{aligned} N_2^2 - N_2^1 &= \sum_{h^{T-1}} \left(p_2\left(h^{T-1}\right) - p_1\left(h^{T-1}\right)\right) \sum_{t=1}^{T} \mathbb{I}\left(A_t\left(h^{t-1}\right) = 2\right) \\ &\leq T \sum_{\substack{h^{T-1}: \\ p_2\left(h^{T-1}\right) \geq p_1\left(h^{T-1}\right)}} \left(p_2\left(h^{T-1}\right) - p_1\left(h^{T-1}\right)\right) \\ &= \frac{T}{2}\|p_2 - p_1\|_1 \leq c_1 T \sqrt{D\left(p_2 \| p_1\right)}, \end{aligned}$$

where the last step follows from Pinsker's inequality [7]. Using the chain rule for KL divergence we can write

$$\begin{aligned} D\left(p_2 \| p_1\right) &= \sum_{t=1}^{T} D\left(p_2(h_t|h^{t-1}) \| p_1(h_t|h^{t-1})\right) \\ &= \sum_{t=1}^{T} \sum_{h^{t-1}} p_2(h^{t-1}) \sum_{h_t} p_2(h_t|h^{t-1}) \ln \frac{p_2(h_t|h^{t-1})}{p_1(h_t|h^{t-1})} \\ &= \sum_{t=1}^{T} \sum_{h^{t-1}} \mathbb{I}(A_t(h^{t-1}) \geq 3) p_2(h^{t-1}) \sum_{h_t \in \{1,2\}} p_2(h_t|h^{t-1}) \ln \frac{p_2(h_t|h^{t-1})}{p_1(h_t|h^{t-1})} \qquad (7) \\ &= \sum_{t=1}^{T} \sum_{h^{t-1}} \mathbb{I}(A_t(h^{t-1}) \geq 3) p_2(h^{t-1}) \left(\frac{2\epsilon^2}{\alpha(1-\alpha)} + O\left(\epsilon^3\right)\right) \qquad (8) \\ &= \left(\frac{2\epsilon^2}{\alpha(1-\alpha)} + O\left(\epsilon^3\right)\right) N_{\geq 3}^2 \,. \end{aligned}$$

In (7) we used that if we play action 1 or 2 then our observation h_t will be $*$ in both models 1 and 2, whereas if we play action $i \geq 3$ then $h_t \in \{1, 2\}$, while in (8) we used Lemma 6. □

The expected regret of the algorithm can be bounded in terms of N_i^j:

$$\mathbb{E}_1[\widehat{\mathrm{R}}_T] \geq \underbrace{\left(\ell_1^1(\alpha+\epsilon) + \ell_2^1(1-\alpha-\epsilon) - \alpha(1-\alpha-\epsilon)\right)}_{f_1} N_{\geq 3}^1 + \epsilon N_2^1$$

$$\mathbb{E}_2[\widehat{\mathrm{R}}_T] \geq \underbrace{\left(\ell_1^2(\alpha-\epsilon) + \ell_2^2(1-\alpha+\epsilon) - (1-\alpha)(\alpha-\epsilon)\right)}_{f_2} N_{\geq 3}^2 + \epsilon N_1^2$$

where, for an outcome i, ℓ_i^j is the loss of the best revealing action given model j. Now, by (6), there exists $\tau > 0$ such that for all $i \geq 3$, $\ell_{i,1} \geq (1-\lambda_i)(1-\alpha) + \tau$ and $\ell_{i,2} \geq \alpha\lambda_i + \tau$. Simple algebra gives that $f_1 \geq (1-\lambda_{max})\epsilon + \tau$ and $f_2 \geq \lambda_{min}\epsilon + \tau$. Hence, if ϵ is small enough then both f_1 and f_2 are positive. Therefore, choosing $j = \arg\min_{l \in \{1,2\}}(N_{\geq 3}^l)$ and using Lemma 7 we get

$\mathbb{E}_i[\widehat{\mathrm{R}}_T] \geq f_i N^j_{\geq 3} + \epsilon \left(N^j_{3-i} - cT\epsilon\sqrt{N^j_{\geq 3}} \right)$, $i = 1, 2$. Finally, randomizing over the two models such that each of them is chosen with equal probability and denoting the corresponding expectation by $\mathbb{E}[\cdot]$, setting ϵ to $c_2 T^{-1/3}$ we have $\mathbb{E}[\widehat{\mathrm{R}}_T] \geq (\tau - \frac{\lambda^* c_2 T^{-1/3}}{2}) N^j_{\geq 3} + c_2 T^{2/3} - c_2^2 c T^{1/3} \sqrt{N^j_{\geq 3}} > T^{2/3} \left((\tau - \frac{\lambda^* c_2}{2}) x^2 + c_2 - c_2^2 c x \right)$, where $x = \sqrt{\frac{N^j_{\geq 3}}{T^{2/3}}}$. Now it is easy to see that c_2 can be set such that, independently of x, the right hand side is always positive and thus it is $\Omega(T^{2/3})$. $\square$

5 Discussion

In this paper we classified partial-monitoring games with two outcomes based on their minimax regret. The most important open question is whether our results generalize to games with more outcomes.

A simple observation is that, given a finite partial-monitoring game, if we restrict Nature's set of actions to any two outcomes, the resulting game's hardness serves as a lower bound on the minimax regret of the original game. This gives us a sufficient condition that a game has $\Omega(T^{2/3})$ minimax regret. We believe that the $\Omega(T^{2/3})$ lower bound can also be generalized to situations where two "ϵ-close" outcome distributions are not distinguishable by playing only their respective optimal actions. Generalizing the upper bound result seems more challenging. The algorithm AppleTree heavily exploits the two-dimensional structure of the losses and, as of yet, in general we do not know how to construct an algorithm that achieves $\widetilde{O}(\sqrt{T})$ regret on partial-monitoring games with more than two outcomes.

References

[1] Lugosi, G., Cesa-Bianchi, N.: Prediction, Learning, and Games. Cambridge University Press, Cambridge (2006)
[2] Audibert, J.-Y., Bubeck, S.: Minimax policies for adversarial and stochastic bandits. In: Proceedings of the 22nd Annual Conference on Learning Theory (2009)
[3] Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.E.: The nonstochastic multiarmed bandit problem. SIAM Journal on Computing 32(1), 48–77 (2003)
[4] Cesa-Bianchi, N., Lugosi, G., Stoltz, G.: Regret minimization under partial monitoring. Mathematics of Operations Research 31(3), 562–580 (2006)
[5] Cesa-Bianchi, N., Freund, Y., Haussler, D., Helmbold, D.P., Schapire, R.E., Warmuth, M.K.: How to use expert advice. Journal of the ACM 44(3), 427–485 (1997)
[6] Piccolboni, A., Schindelhauer, C.: Discrete prediction games with arbitrary feedback and loss. In: Helmbold, D.P., Williamson, B. (eds.) COLT 2001 and EuroCOLT 2001. LNCS (LNAI), vol. 2111, pp. 208–223. Springer, Heidelberg (2001)
[7] Cover, T.M., Thomas, J.A.: Elements of Information Theory, 2nd edn. Wiley, New York (2006)

Switching Investments

Wouter M. Koolen[1] and Steven de Rooij[2]

[1] Centrum Wiskunde en Informatica (CWI), P.O. Box 94079,
NL-1090 GB Amsterdam
wmkoolen@cwi.nl

[2] Statistical Laboratory, DPMMS, Wilberforce Road, Cambridge CB3 0WB, UK
steven@statslab.cam.ac.uk

Abstract. We present a simple online two-way trading algorithm that exploits fluctuations in the unit price of an asset. Rather than analysing worst-case performance under some assumptions, we prove a novel, unconditional performance bound that is parameterised either by the actual dynamics of the price of the asset, or by a simplifying model thereof. The algorithm processes T prices in $O(T^2)$ time and $O(T)$ space, but if the employed prior density is exponential, the time requirement reduces to $O(T)$. The result translates to the prediction with expert advice framework, and has applications in data compression and hypothesis testing.

1 Introduction

We consider a two-player game played between *Investor* and *Nature*. Investor starts out with one unit of cash. At each time, Investor decides which fraction of his current capital to invest in an asset (denoted A), and how much to keep in his boot (denoted B). Nature, on the other hand, chooses the price of the asset.

A play for Nature is a function $\Lambda : [0, T] \to \mathbb{R}$ that specifies the natural logarithm of the unit price of A as a function of time. The end-time T is part of Nature's move and unknown to Investor. An example play is shown in Figure 1.

Investor's *payoff* is defined as the natural logarithm of his capital at the end-time T, where shares owned are valued at the final logprice $\Lambda(T)$. In hindsight, it would have been optimal for Investor to follow the strategy S_Λ that invests all capital in A at local minima of Λ, and liquidates all shares into B at local maxima. Let $\boldsymbol{z} = z_0, \ldots, z_m$ denote the sequence of logprices at local extrema of Λ, with $z_0 = \Lambda(0)$ and $z_m = \Lambda(T)$. The payoff of the strategy S_Λ thus equals

$$S_\Lambda {*} \Lambda \;\coloneqq\; \sum_{1 \le i \le m} \max\{0, z_i - z_{i-1}\}.$$

We construct a foresight-free, computationally efficient strategy π that guarantees payoff $\pi {*} \Lambda$ close to $S_\Lambda {*} \Lambda$. The definition of π relies on the selection of a probability density function on $[0, \infty)$ that for convenience we identify with π itself (see Section 2), and we abbreviate $-\ln \pi(h)$ to $\ell(h)$. We then prove

$$\pi {*} \Lambda \;\ge\; S_\Lambda {*} \Lambda - \sum_{1 \le i \le m} \ell\big(|z_i - z_{i-1}|\big) - (m-1)c_\pi - \ln 2 - 2\epsilon_\pi, \qquad (1)$$

© Springer-Verlag Berlin Heidelberg 2010

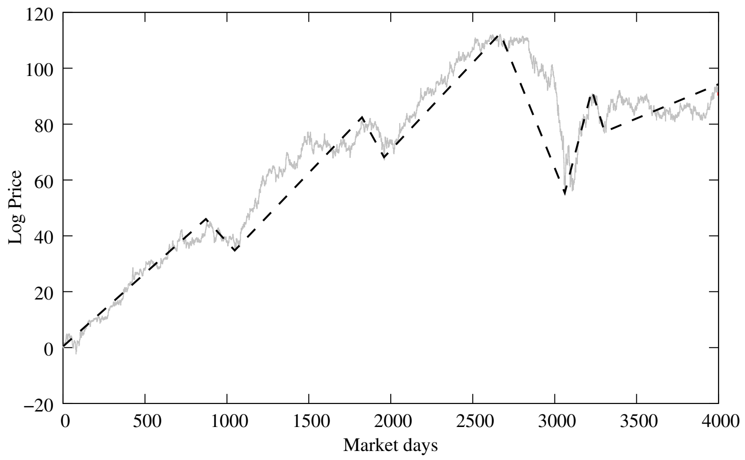


Fig. 1. An example play Λ for Nature, with a regularised trend line Λ'

where c_π and ϵ_π are two constants that depend on π. Thus the payoff of π on Λ falls short of the optimum by an overhead that depends on the complexity of Λ, measured in terms of both the length of the vector $\boldsymbol{z}$, and the sizes of its entries. The bound is entirely independent of the time scale T.

When Λ is simple, i.e. has few large fluctuations, (1) shows that π exploits almost all achievable payoff. The bound degenerates when Λ sports many small fluctuations, for which the overhead $\ell(x)$ exceeds the benefit x of trading. However, we prove that for any *regularisation* Λ' of Λ, as illustrated by the dashed line in Figure 1 and defined precisely in Section 3.2, π's payoff satisfies

$$\pi{*}\Lambda' \;\leq\; \pi{*}\Lambda. \tag{2}$$

Thus, we may pretend that Nature actually played Λ', and apply the bound (1) with Λ' in place of Λ. In fact the regulariser Λ' may be interpreted as a *model* for Nature's play Λ. The most complex model then yields the bound as presented in (1), but we may now concern other models, that strike a better balance between model complexity and goodness of fit. Such tradeoff models will usually yield better bounds. In conclusion, if in hindsight a simple regulariser can be found with large payoff, then π will collect most of that payoff as well.

Example 1. Let Λ and Λ' be, respectively, the play for Nature and the regulariser shown in Figure 1. The extrema of the regulariser are given by $\boldsymbol{z}' = (0, 42, 36, 82, 68, 112, 57, 90, 77, 90)$. Then $S_{\Lambda'}{*}\Lambda' = (42-0) + (82-36) + \ldots + (90-77) = 178$. Now we select the exponential density listed in Table 1 for the definition of π; the values for c_π and ϵ_π are also listed there. We can now apply bounds (2) and (1) to find

$$\pi{*}\Lambda \;\geq\; \pi{*}\Lambda' \;\geq\; 178 - 64.8 - 8\cdot 0.034 - \ln 2 - 2\cdot 3.40 \;\approx\; 105.4.$$

Note that there may be choices of Λ' for which the bound is better, and even for the optimal choice of Λ' the strategy π may perform substantially better than our bound indicates. The actual payoff of π on these data is $\pi{*}\Lambda = 175.4$. $\diamond$

Applications and Related Work. Our model and its analysis are phrased in financial terms. However, it applies much more widely. We list four examples.

One-Way Trading and Two-Way Trading. This is the most direct example. We let Λ be the logarithm of the exchange rate between any two assets, say dollar and yen. If we forbid selling A, we obtain the setting called *One-way Trading.* Efficient algorithms with minimax payoff for one-way trading under various restrictions on Nature's play Λ are known. E.g. fixed daily price growth range [2], fixed price range [7] and bounded quadratic variation [6]. Two-way trading guarantees are derived in [4] by iterating a unidirectional trading algorithm back and forth. Both the algorithms and the bounds are parametrised by the restrictions placed on Nature's play.

Our results are of a different kind. First, no restrictions are placed on Nature's play. Second, our guarantees are expressed in terms of Nature's actual play (or a regularisation thereof), and hence remain informative when Nature does not play to ruin Investor.

Prediction with Expert Advice. Two experts, say A and B sequentially issue predictions. We denote their cumulative loss at time t by $L_{\mathrm{A}}(t)$ and $L_{\mathrm{B}}(t)$. We let $\Lambda(t) = L_{\mathrm{B}}(t) - L_{\mathrm{A}}(t)$. In prediction tasks with so-called *mixable loss* [13], guarantees for our financial game directly translate to expert performance bounds and vice versa. Efficient strategies include the seminal *Fixed Share* [9], the *Switching Method* [12], and the *Switch Distribution* and its derivatives [8, 10]. These algorithms guarantee payoff $\rho{*}\Lambda \geq S_{\Lambda'}{*}\Lambda' - O(m' \ln T)$ for each Λ' with m' blocks. The logarithmic dependence of the bound on the time T of these algorithms means that for any arbitrary number h, if by switching just a single time the payoff could be improved by h, there is a sample size T such that these algorithms are not able to exploit this.

Variable Share [9] switches based on the losses L^{A} and L^{B}. Its payoff guarantee depends logarithmically on the loss of the best reference strategy with m' blocks. However, its analysis assumes so-called *bounded loss*, and does not apply to financial games (which involve logarithmic loss, which is unbounded).

Prefix Coding/Compression. Fix two prefix codes A and B for a sequence of outcomes $x_1, \ldots, x_T$. Let $L_{\mathrm{A}}(t)$ and $L_{\mathrm{B}}(t)$ denote the code-length of A and B on the outcomes $x_1, \ldots, x_t$ measured in nats. Now let $\Lambda(t) = L_{\mathrm{B}}(t) - L_{\mathrm{A}}(t)$. It is well-known that we can build a prefix code that attains code length $\ln(2) + \min\big\{L_{\mathrm{A}}(T), L_{\mathrm{B}}(T)\big\}$ on the data. When different codes are good for different segments of the data, we observe fluctuation in Λ. Using standard information-theoretic methods, e.g. [3], our financial prediction scheme can be transformed into a prefix code that exploits these fluctuations.

Hypothesis Testing. We are given a *null hypothesis* P_0 and an *alternative hypothesis* P_1. Both candidate hypotheses are probabilistic models for some sequence of observations $x_1, x_2, \dots, x_T$. Let $\Lambda(t) = \ln\bigl(P_1(x_1, \dots, x_t)/P_0(x_1, \dots, x_t)\bigr)$ be the loglikelihood ratio between P_1 and P_0. Thus Λ measures the amount of evidence against the null hypothesis and can be used as a test statistic. Traditionally [1], we choose a threshold $\tau > 0$ and reject the null hypothesis when $\Lambda(T) \geq \tau$, an event that is extremely unlikely under P_0. The case where $\Lambda(T)$ is below the threshold τ, while $\Lambda(t) \geq \tau$ at some earlier time t is considered in [11, 5], and tests are presented that lose as little evidence as possible while remaining unbiased. These tests are based on strategies that switch only once, and resemble strategies for one-way trading. By the same method, our strategy induces a fair test statistic that can be used to reject P_0 whenever Λ fluctuates heavily; an event that is also unlikely under P_0.

Outline. We explicate the setting and describe the strategy π for Investor in Section 2. We analyse the payoff of π and prove our payoff guarantee in Section 3. We then show how to implement the strategy π efficiently in Section 4.

2 Setting

We introduce the details of our financial game. We first review Nature's play Λ. We then construct strategies for Investor, culminating in the definition of the strategy π. We conclude this section with a lemma that simplifies all later proofs by exploiting the symmetry between A and B.

2.1 Nature's Play Λ

A play for Nature is a logprice function $\Lambda : [0, T] \to \mathbb{R}$. The end-time T is part of Nature's move, and unknown to Investor.

For simplicity, we restrict attention to the setting where Λ is discrete, i.e. piecewise constant with jumps at integer times. This is sufficient for the practical scenario where Λ is monitored intermittently (albeit possibly very often). Later in the analysis it will be convenient for technical reasons to generalise to piecewise continous plays for Nature with finitely many local extrema and finitely many discontinuities; nevertheless ultimately we remain concerned with the discrete setting only.

Our results do extend quite readily to the wide class of càdlàg logprice functions (right-continuous with left limits). These encompass continuous time models that are often considered in the financial literature, such as Brownian motion with drift, etc. Such theoretically interesting generalisations are deferred to future publications.

2.2 Investor's Strategy π

We now construct the strategy π for Investor in three stages. Two basic strategies exist. Strategy A invests the initial unit capital in the asset, whereas strategy

B keeps all capital in the boot. At the end of the game, all shares are valued at the final logprice $\Lambda(T)$. The payoffs, defined as Investor's final logcapital, of the basic strategies equal

$$\mathrm{A}{*}\Lambda \;:=\; \Lambda(T) - \Lambda(0) \qquad\qquad \text{and} \qquad\qquad \mathrm{B}{*}\Lambda \;:=\; 0.$$

Since we use logprice differences extensively, we abbreviate $\Lambda(t) - \Lambda(s)$ to $\Lambda|_s^t$.

Time-Switched Strategies. From these basic strategies A and B we construct more interesting strategies. Let $\boldsymbol{t} = t_0, t_1, t_2, t_3, \ldots$ be a sequence of times such that $0 = t_0 \leq t_1 \leq t_2 \leq \ldots$ The strategy $\boldsymbol{t}^{\mathrm{A}}$ switches at times $\boldsymbol{t}$ starting with A. That is, $\boldsymbol{t}^{\mathrm{A}}$ invests all capital in A until time t_1. At that time it sells all shares, and keeps all money in B until time t_2. Then it again invests all capital in A until time t_3 etc. Symmetrically, $\boldsymbol{t}^{\mathrm{B}}$ is the strategy that switches at times $\boldsymbol{t}$ starting with B. Thus the payoffs of $\boldsymbol{t}^{\mathrm{A}}$ and $\boldsymbol{t}^{\mathrm{B}}$ when Nature plays Λ are

$$\boldsymbol{t}^{\mathrm{A}}{*}\Lambda \;:=\; \sum_{i=0}^{\infty} \Lambda|_{T\wedge t_{2i}}^{T\wedge t_{2i+1}} \qquad\qquad \text{and} \qquad\qquad \boldsymbol{t}^{\mathrm{B}}{*}\Lambda \;:=\; \sum_{i=0}^{\infty} \Lambda|_{T\wedge t_{2i+1}}^{T\wedge t_{2i+2}}.$$

Of course, a good time switch sequence $\boldsymbol{t}$ for Investor depends on Nature's unknown move Λ. However, Investor may hedge by dividing his initial capital according to some prior distribution ρ on the switch time sequence $\boldsymbol{t}$, and construct time-switched strategies ρ^{A} and ρ^{B} with payoffs

$$\rho^{\mathrm{A}}{*}\Lambda \;:=\; \ln \int \exp\bigl(\boldsymbol{t}^{\mathrm{A}}{*}\Lambda\bigr)\, \mathrm{d}\rho(\boldsymbol{t}) \quad \text{and} \quad \rho^{\mathrm{B}}{*}\Lambda \;:=\; \ln \int \exp\bigl(\boldsymbol{t}^{\mathrm{B}}{*}\Lambda\bigr)\, \mathrm{d}\rho(\boldsymbol{t}),$$

and the meta strategy ρ with payoff $\rho{*}\Lambda := \ln\left(\frac{1}{2}\exp\bigl(\rho^{\mathrm{A}}{*}\Lambda\bigr) + \frac{1}{2}\exp\bigl(\rho^{\mathrm{B}}{*}\Lambda\bigr)\right)$.

Price-Switched Strategies. Price-switched strategies decide when to trade based on the logprice $\Lambda(t)$ instead of the time t itself. This renders their payoff independent of the time-scale. Fix a sequence of nonnegative reals $\boldsymbol{\delta} = \delta_1, \delta_2, \ldots$ We denote by $\boldsymbol{\delta}^{\mathrm{A}}$ the strategy that initially invests all capital in A, and waits until the first time s_1 where the logprice difference $\Lambda|_0^{s_1}$ is at least δ_1. It then sells all shares and puts the money into B, until the first subsequent time s_2 that the logprice difference $\Lambda|_{s_1}^{s_2}$ is at most $-\delta_2$. Then it invests all capital into A again, until the logprice difference $\Lambda|_{s_2}^{s_3}$ is at least δ_3, etc. The strategy $\boldsymbol{\delta}^{\mathrm{B}}$ is defined symmetrically, with switching times $r_0, r_1, \ldots$ The switching time sequences $\boldsymbol{s}$ and $\boldsymbol{r}$ are obtained as follows. First $s_0 = r_0 = 0$. Then recursively

$$\begin{aligned} s_i &:= \min\bigl\{t \geq s_{i-1} \bigm| \Lambda|_{s_{i-1}}^t \geq +\delta_i\bigr\} & r_i &:= \min\bigl\{t \geq r_{i-1} \bigm| \Lambda|_{r_{i-1}}^t \leq -\delta_i\bigr\} & &i \text{ even},\\ s_i &:= \min\bigl\{t \geq s_{i-1} \bigm| \Lambda|_{s_{i-1}}^t \leq -\delta_i\bigr\} & r_i &:= \min\bigl\{t \geq r_{i-1} \bigm| \Lambda|_{r_{i-1}}^t \geq +\delta_i\bigr\} & &i \text{ odd}. \end{aligned}$$

Both $\boldsymbol{s}$ and $\boldsymbol{r}$ are a function of $\boldsymbol{\delta}$ and Λ and satisfy $\boldsymbol{s}(\boldsymbol{\delta}, \Lambda) = \boldsymbol{r}(\boldsymbol{\delta}, -\Lambda)$. By convention, the minimum is infinite if no suitable successor time exists in the domain of Λ, i.e before time T. The payoffs of $\boldsymbol{\delta}^{\mathrm{A}}$ and $\boldsymbol{\delta}^{\mathrm{B}}$ are given by

$$\boldsymbol{\delta}^{\mathrm{A}}{*}\Lambda \;:=\; \boldsymbol{s}^{\mathrm{A}}{*}\Lambda \qquad\qquad \text{and} \qquad\qquad \boldsymbol{\delta}^{\mathrm{B}}{*}\Lambda \;:=\; \boldsymbol{r}^{\mathrm{B}}{*}\Lambda.$$

The strategy $\boldsymbol{\delta}^{\mathrm{A}}$ has the following property. Whenever it sells its shares, say at time s_i for some odd i, the asset price, and hence its capital, has multiplied by *at least* $\exp(\delta_i) \geq 1$ since the acquisition at time s_{i-1}. This holds irrespective of Nature's play. In particular, between time s_i and s_{i+1} for odd i, the logarithm of its capital equals

$$\Lambda|_{s_0}^{s_1} + \Lambda|_{s_2}^{s_3} + \Lambda|_{s_4}^{s_5} + \ldots + \Lambda|_{s_{i-1}}^{s_i} \;\geq\; \delta_1 + \delta_3 + \delta_5 + \ldots + \delta_i.$$

Of course, for each logprice difference sequence $\boldsymbol{\delta}$, the number of switches that is executed, and hence the quality of $\boldsymbol{\delta}^{\mathrm{A}}$ depends on Nature's move Λ. Let $\mathcal{D} = \left\{\boldsymbol{\delta}^{\mathrm{A}}, \boldsymbol{\delta}^{\mathrm{B}} \;\middle|\; \boldsymbol{\delta} \in [0,\infty)^{\infty}\right\}$ be the set of price-switched strategies for Investor.

The Strategy $\boldsymbol{\pi}$. Again, we may hedge by dividing our initial capital according to some prior π on $\boldsymbol{\delta}$, and obtain strategies π^{A} and π^{B} with payoffs

$$\pi^{\mathrm{A}}{*}\Lambda \;:=\; \ln \int \exp\bigl(\boldsymbol{\delta}^{\mathrm{A}}{*}\Lambda\bigr)\,\mathrm{d}\pi(\boldsymbol{\delta}) \quad \text{and} \quad \pi^{\mathrm{B}}{*}\Lambda \;:=\; \ln \int \exp\bigl(\boldsymbol{\delta}^{\mathrm{B}}{*}\Lambda\bigr)\,\mathrm{d}\pi(\boldsymbol{\delta}),$$

and the meta strategy π with payoff $\pi{*}\Lambda := \ln\left(\frac{1}{2}\exp\bigl(\pi^{\mathrm{A}}{*}\Lambda\bigr) + \frac{1}{2}\exp\bigl(\pi^{\mathrm{B}}{*}\Lambda\bigr)\right)$. Note that the price-switched strategies in $\mathcal{D}$ are independent of the time scale, and so are these strategies based on them.

Requirements on π. The above construction works for any prior π. In this paper we analyse the behaviour of strategies π that satisfy these requirements:

1. π is the independent infinite product distribution of some probability density function on $[0,\infty)$. Since the distinction is always clear, we also denote the univariate density by π.
2. the function $x \mapsto \mathrm{e}^x\pi(x)$ is increasing.
3. the density π is log-convex.

The first requirement ensures that we can hedge capital according to π. The second requirement ensures that paying $-\ln\pi(x)$ to gain x is a better deal when x is larger. The third requirement ensures that we rather pay $-\ln\pi(x+y)$ than $-\ln\pi(x) - \ln\pi(y)$ to gain $x+y$. We use the following consequences in our bounds.

Lemma 1. *Let π satisfy the requirements 1–3 above. Then*

1. π is strictly positive.
2. π is strictly decreasing.
3. $\int_h^\infty \pi(x)\,\mathrm{d}x \geq \pi(h)$ for each $h \geq 0$.

Proof. Since π is a convex probability density, it is decreasing and thus $0 < \pi(0) = \mathrm{e}^0\pi(0)$. Since $\mathrm{e}^x\pi(x)$ increases, we have $\pi(x) > 0$ for all x. Then, since π is a non-zero convex probability density, it must be *strictly* decreasing. Finally, for $0 \leq h \leq x$ we have $\pi(x) = \pi(x)\mathrm{e}^x\mathrm{e}^{-x} \geq \pi(h)\mathrm{e}^h\mathrm{e}^{-x}$. Therefore $\int_h^\infty \pi(x)\,\mathrm{d}x \;\geq\; \pi(h)\int_h^\infty \mathrm{e}^{h-x}\,\mathrm{d}x \;=\; \pi(h)$. □

The last fact implies that the density $\pi(x) \leq 1$ for all x. Throughout this paper, we abbreviate $-\ln\pi(x)$ to $\ell(x)$. Thus ℓ is nonnegative, concave and increasing.

Example 2. The densities shown in Table 1, ordered from heavy to light tails, satisfy all the requirements. ◊

Table 1. Example priors

	Fat tail	Pareto	Exponential
$\pi(x)$	$\dfrac{\log(o)}{(x+o)(\log(x+o))^2}$	$(c-1)o^{c-1}(x+o)^{-c}$	$\alpha \mathrm{e}^{-\alpha x}$
Condition	$2 \leq (o-1)\log o$ (Sufficient: $o \geq 2.89$)	$1 < c \leq o$	$0 < \alpha \leq 1$
Parameters	$o = 3$	$c = 2$, $o = 3$	$\alpha = 1/3$
ϵ_π	4.10396	3.55884	3.39788
c_π	0.016645	0.0288849	0.034016

2.3 Exploiting Symmetry

Payoff is measured as (the natural logarithm of) Investor's final amount of cash. Of course, cash and asset are intrinsically symmetric. We make this precise as follows. We say that the following pairs of strategies are *dual*

$$\mathrm{A}, \mathrm{B} \qquad \boldsymbol{t}^{\mathrm{A}}, \boldsymbol{t}^{\mathrm{B}} \qquad \rho^{\mathrm{A}}, \rho^{\mathrm{B}} \qquad \rho, \rho \qquad \boldsymbol{\delta}^{\mathrm{A}}, \boldsymbol{\delta}^{\mathrm{B}} \qquad \pi^{\mathrm{A}}, \pi^{\mathrm{B}} \qquad \pi, \pi$$

and vice versa in each case. The meta strategies ρ and π are self-dual.

Lemma 2 (Duality). *Let S and S' be dual strategies. Then for each Λ*

$$S{*}\Lambda \;=\; S'{*}(-\Lambda) + \Lambda|_0^T.$$

Proof. The lemma is trivial for the dual pair A and B. We proceed to prove the lemma for the dual strategies $\boldsymbol{t}^{\mathrm{A}}$ and $\boldsymbol{t}^{\mathrm{B}}$, the other cases follow simply by definition. Recall that $\exp(\Lambda)$ is the asset price in cash per share, so that $\exp(-\Lambda)$ is the price in shares per cash. Thus $\boldsymbol{t}^{\mathrm{B}}{*}(-\Lambda)$ is the log-number of shares resulting from investing one share according to the strategy $\boldsymbol{t}^{\mathrm{A}}$. Finally, $\Lambda|_0^T = \Lambda(T) - \Lambda(0)$ is the result of exchanging cash to asset initially, and asset to cash at the end. □

3 Payoff Bound

In this section we prove the payoff guarantees for the strategy π that were given in the introduction. We build towards the statement and proof of a more precise version of the bounds in the following subsections. First, in Section 3.1 we show that Nature's worst-case logprice functions are continuous. Then, in Section 3.2 we show that Investor's payoff decreases when Nature plays more regular. In Section 3.3 we analyse Investor's payoff under a regularity assumption on Λ called γ-separation. Finally, in Section 3.4 we show how to establish γ-separation if it does not obtain and establish the bound in the form of Theorem 5.

3.1 Nature Plays a Continuous Logprice Function Λ

We now prove that it is sub-optimal for Nature to play a discontinuous Λ. To do so, we show that Investor's payoff is reduced when Nature eliminates a jump by inserting a linear interpolation. Let Λ have a discontinuity at t. We define Λ', *the t-ironing of Λ*, by $\Lambda'(s) := \Lambda(s)$ for $s < t$, $\Lambda'(s+1) := \Lambda(s)$ for $s > t$, and $\Lambda'(s) := (1+t-s)\Lambda(t-)+(s-t)\Lambda(t)$ for $t \leq s \leq t+1$, where $\Lambda(t-) := \lim_{s\uparrow t} \Lambda(s)$. This definition is illustrated by Figure 2.

Theorem 1 (Continuous Free Lunch). *Fix any play for Nature Λ with a discontinuity at time t, and let Λ' be the t-ironing of Λ. Then*

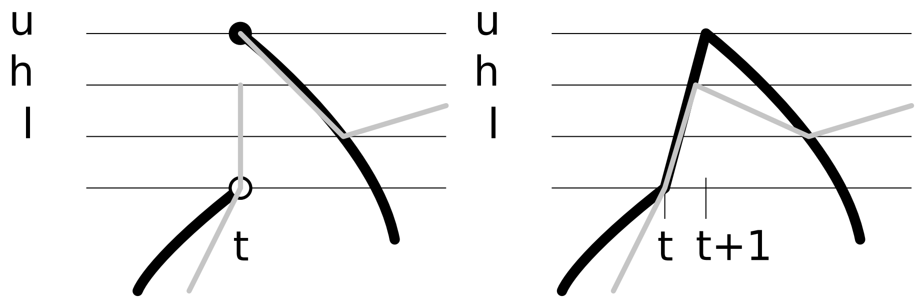
$$\pi * \Lambda' \;\leq\; \pi * \Lambda.$$

Proof. See Figure 2. By duality (Lemma 2), we may assume that the jump is upward. Obviously, any strategy $\boldsymbol{\delta}'$ that does not switch at time t on Λ has identical payoff on Λ and Λ'. Now consider any strategy $\boldsymbol{\delta}' = (\dots, h, h-l, \dots)$, where h prompts a switch at time t on Λ. We now modify the strategy to $\boldsymbol{\delta} = (\dots, h, u-l, \dots)$ and we compare the term corresponding to $\boldsymbol{\delta}'$ in the integral for $\pi*\Lambda'$ to the term corresponding to $\boldsymbol{\delta}$ in the integral for $\pi*\Lambda$:

$$\frac{\exp(\boldsymbol{\delta}' * \Lambda')\pi(\boldsymbol{\delta}')}{\exp(\boldsymbol{\delta} * \Lambda)\pi(\boldsymbol{\delta})} \;=\; \frac{\exp(h-l)\pi(h-l)}{\exp(u-l)\pi(u-l)} \;\leq\; 1,$$

where the inequality uses that $e^h \pi(h)$ is increasing (see Section 2.2). The proof follows by observing that the mapping that takes $\boldsymbol{\delta}'$ to $\boldsymbol{\delta}$ is a translation. □

When Investor follows the strategy π, there is no benefit for Nature to playing a logprice function Λ with jumps. Without loss of generality we henceforth restrict Nature to continuous plays. This simplifies analysis considerably, as it allows us to assume that switches specified by any $\boldsymbol{\delta}$ occur at *exactly* the specified logprices.



(a) Nature's move Λ with jump at time t, and strategy $\boldsymbol{\delta} = (\dots, h, u-l, \dots)$. (b) The t-ironing Λ' of Λ, and strategy $\boldsymbol{\delta}' = (\dots, h, h-l, \dots)$.

Fig. 2. Worst-Case Plays for Nature are Continuous

3.2 Ordering by Regularity

Given a move for Nature $\Lambda : [0,T] \to \mathbb{R}$, we say that another move $\Lambda' : [0,T'] \to \mathbb{R}$ is *more regular* than Λ, denoted $\Lambda' \preccurlyeq \Lambda$, if there is a monotonic function $f : [0,T'] \to [0,T]$ such that $f(0) = 0$, $f(T') = T$ and $\Lambda' = \Lambda \circ f$. That is, the price levels of the *regularisation* Λ' are a subsequence of the price levels of Nature's move Λ, with the same initial and final price, but potentially less fluctuation. We now show that by following a fixed price-switched strategy, Investor gets richer whenever Nature's move is less regular.

Theorem 2 (Monotonicity). *For each price-switched strategy $S \in \mathcal{D}$ and continuous logprice functions Λ and Λ'*

$$\Lambda' \preccurlyeq \Lambda \qquad\qquad \textit{implies} \qquad\qquad S{*}\Lambda' \leq S{*}\Lambda.$$

Proof. First note that $\Lambda' \preccurlyeq \Lambda$ iff $-\Lambda' \preccurlyeq -\Lambda$. So by symmetry (Lemma 2) it suffices to prove the theorem for the strategies in $\mathcal{D}$ that start with A. We proceed by induction on the number of switches executed by the strategy $\boldsymbol{\delta}^{\mathrm{A}}$ on the regulariser Λ'. For the base case, suppose this number is zero, i.e $\Lambda'|_0^t < \delta_1$ for each $0 \leq t \leq T'$. Let $m \geq 1$ denote the number of blocks of $\boldsymbol{\delta}^{\mathrm{A}}$ on Λ. There are two cases. If m is even then $\boldsymbol{\delta}^{\mathrm{A}}$ follows B on the last block. Since $\delta_1 > \Lambda'|_0^{T'}$

$$\boldsymbol{\delta}^{\mathrm{A}}{*}\Lambda = \sum_{1 \leq i < m \text{ odd}} \delta_i \geq \delta_1 > \Lambda'|_0^{T'} = \boldsymbol{\delta}^{\mathrm{A}}{*}\Lambda'.$$

If m is odd, then $\boldsymbol{\delta}^{\mathrm{A}}$ follows A on the last block. Again using Lemma 2, we get

$$\boldsymbol{\delta}^{\mathrm{A}}{*}\Lambda = \sum_{1 \leq i < m \text{ even}} \delta_i + \Lambda|_0^T \geq \Lambda|_0^T = \Lambda'|_0^{T'} = \boldsymbol{\delta}^{\mathrm{A}}{*}\Lambda'.$$

To prove the induction step, suppose a switch is executed, i.e. the first difference δ_1 is present in the regulariser Λ', and hence also in Nature's play Λ, then the strategy $\boldsymbol{\delta}^{\mathrm{A}}$ switches at price level $\Lambda(0) + \delta_1$ on either play, resulting in the same capital. The switches may occur at different times on Λ and Λ'. Nevertheless, the induction hypothesis applies to the tails of the plays since the remainder of the regulariser Λ' is more regular than the remainder of Nature's move Λ. □

Since the theorem holds pointwise in $\mathcal{D}$, it also holds for the mixture strategy π.

3.3 With γ-Separation

Fix a logprice function Λ. Throughout this section, we use the following notation:

Definition 1. *We denote by $\boldsymbol{z} = z_0, z_1, \ldots, z_m$ the sequence of logprices at the local extrema of Λ (attained or not), with $z_0 = \Lambda(0)$ and $z_m = \Lambda(T)$, and we say that Λ has m* blocks. *Let $\boldsymbol{\Delta} = \Delta_1, \ldots, \Delta_m$ denote the sequence of absolute logprice differences, i.e. $\Delta_i := |z_i - z_{i-1}|$.*

Definition 2. *We say that Λ has* γ-separation *if $\Delta_1, \Delta_m \geq \gamma$ and $\Delta_i \geq 2\gamma$ for each $1 < i < m$. That is, the border optima have logprice difference at least γ with the border and each subsequent pair of local extrema has at least logprice difference 2γ.*

We now analyse the payoff of the strategy π, assuming that Λ has γ-separation.

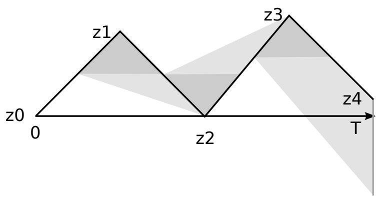


Fig. 3. Domain of Integration Example. Some Λ, with $m = 4$, is shown in black. The height of the dark gray triangles equals γ. This Λ has γ-separation. In particular $\Delta_2 = z_1 - z_2 = 2\gamma$. Theorem 3 integrates over the strategies that are optimal for log-price functions in the light gray region.

Theorem 3 (γ-Separation Payoff). *For each Λ with γ-separation*

$$\pi{*}\Lambda \geq \underbrace{\sum_{1 \leq i \leq m} (z_i - z_{i-1})_+}_{gain} - \underbrace{\sum_{1 \leq i \leq m} \ell(\Delta_i)}_{complexity\ penalty} + (m-1)\underbrace{\ln(1 - \mathrm{e}^{-\gamma})}_{overhead\ per\ switch} - \underbrace{\ln 2}_{parity} .$$

Proof. We saw in Section 2.3 that π is self-dual, so by symmetry (Lemma 2) we may assume that $z_0 \leq z_1$. As our first bound, we use $\pi{*}\Lambda \geq \pi^{\mathrm{A}}{*}\Lambda - \ln 2$. Recall that the payoff $\pi^{\mathrm{A}}{*}\Lambda$ is defined as $\ln \int \exp\big(\boldsymbol{\delta}^{\mathrm{A}}{*}\Lambda\big)\,\mathrm{d}\pi(\boldsymbol{\delta})$. As the next step, we re-parameterise the integral by introducing variables $\boldsymbol{h}$, with $h_i := z_0 - \sum_{1 \leq j \leq i} (-1)^j \delta_j$. That is, h_i is the logprice at the ith switch of $\boldsymbol{\delta}^{\mathrm{A}}$. Then we obtain a lower bound by restricting the domain of integration. For $1 \leq i < m$ we restrict $h_i \in [z_i - \gamma, z_i]$ for odd i and $h_i \in [z_i, z_i + \gamma]$ for even i. Thus, we keep all prior mass on strategies that switch at logprices h_i that are at most γ nats short of the optimal switching logprice level z_i. We restrict the last logprice to $h_m \in [z_m, \infty)$ for even m and $h_m \in (-\infty, z_m]$ for odd m. This ensures that we do not switch between h_{m-1} and z_m. Thus, we only integrate over those strategies that closely follow Λ, as illustrated by Figure 3. We first consider even m. Then

$$\pi^{\mathrm{A}}{*}\Lambda \geq \ln \int_{z_1-\gamma}^{z_1} \mathrm{e}^{h_1 - h_0} \pi(h_1 - h_0) \int_{z_2}^{z_2+\gamma} \pi(h_1 - h_2) \int_{z_3-\gamma}^{z_3} \mathrm{e}^{h_3 - h_2} \pi(h_3 - h_2) \cdots$$
$$\cdots \int_{z_{m-1}}^{z_{m-1}+\gamma} \pi(h_{m-2} - h_{m-1}) \int_{z_m}^{\infty} \mathrm{e}^{z_m - h_{m-1}} \pi(h_m - h_{m-1})\,\mathrm{d}\boldsymbol{h}$$

Apply the tail probability bound (Lemma 1(3)) to the innermost integral to get

$$\int_{z_m}^{\infty} \mathrm{e}^{z_m - h_{m-1}} \pi(h_m - h_{m-1}) \, \mathrm{d}h_m \;\geq\; \mathrm{e}^{z_m - h_{m-1}} \pi(z_m - h_{m-1}).$$

Since $|h_i - h_{i-1}| \leq \Delta_i$ and π decreases (Lemma 1(2)) we get

$$\pi^{\mathrm{A}} {*} \Lambda \;\geq\; \ln \prod_{1 \leq i \leq m} \pi(\Delta_i) \;+$$
$$\ln \left(\mathrm{e}^{z_m - z_0} \int\limits_{z_1 - \gamma}^{z_1} \mathrm{e}^{h_1} \int\limits_{z_2}^{z_2 + \gamma} \mathrm{e}^{-h_2} \int\limits_{z_3 - \gamma}^{z_3} \mathrm{e}^{h_3} \cdots \int\limits_{z_{m-2} - \gamma}^{z_{m-2}} \mathrm{e}^{h_{m-2}} \int\limits_{z_{m-1}}^{z_{m-1} + \gamma} \mathrm{e}^{-h_{m-1}} \, \mathrm{d}\boldsymbol{h} \right)$$

Now all integrals have become independent. Rewrite odd/even instances like

$$\int_{z_1 - \gamma}^{z_1} \mathrm{e}^{h_1} \, \mathrm{d}h_1 \;=\; \mathrm{e}^{z_1}(1 - \mathrm{e}^{-\gamma}) \quad \text{and} \quad \int_{z_2}^{z_2 + \gamma} \mathrm{e}^{-h_2} \, \mathrm{d}h_2 \;=\; \mathrm{e}^{-z_2}(1 - \mathrm{e}^{-\gamma}).$$

By rearranging terms we obtain

$$\pi^{\mathrm{A}} {*} \Lambda \;\geq\; \sum_{1 \leq i \leq m} (z_i - z_{i-1})_+ \;-\; \sum_{1 \leq i \leq m} \ell(\Delta_i) \;+\; (m-1) \ln(1 - \mathrm{e}^{-\gamma}).$$

The case for odd m is analogous. □

3.4 Establishing γ-Separation

Say we have a Λ with γ-separation, and hence a performance guarantee by Theorem 3. If γ is small, then a better bound can be obtained by first regularising Λ to a price function Λ^{ϵ} with ϵ-separation for some $\epsilon > \gamma$, and only then applying the theorem. In this section we quantify the gain of going from $\gamma = 0$ to ϵ, and then derive our main payoff bound by tuning ϵ.

The regulariser Λ^{ϵ} is constructed by the algorithm shown in Figure 4a. The key idea of the algorithm, implemented by lines 4–6, is to iteratively remove the smallest fluctuation from $\boldsymbol{z}$. This process is illustrated by Figure 4b. The solid line shows a segment of the logprice function before regularisation. The logprice difference between the two open circles is too small, i.e. $< 2\epsilon$. The dashed line is the logprice function resulting from fluctuation removal. The other lines of the algorithm establish ϵ-separation at the boundaries of Λ.

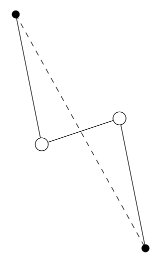
For any sequence $\boldsymbol{z} = z_0, \ldots, z_m$ we abbreviate the terms in the bound of Theorem 3 that depend on $\boldsymbol{z}$ by defining $\boldsymbol{g} = g_1, \ldots, g_m$ and G by

$$g_i \;:=\; (z_i - z_{i-1})_+ - \ell(\Delta_i) \qquad \text{and} \qquad G \;:=\; \sum_{1 \leq i \leq m} g_i.$$

We first study the effect of a single execution of lines 4–6.

```
1: $u \leftarrow (2\epsilon - \Delta_1)_+ \cdot \mathrm{sign}(z_1 - z_0)$.
2: $v \leftarrow (2\epsilon - \Delta_m)_+ \cdot \mathrm{sign}(z_m - z_{m-1})$.
3: $\boldsymbol{z} \leftarrow (z_0, z_1 + u, z_2 + u, \ldots, z_m + u + v)$    ▷ Ensure $\Delta_1, \Delta_m \geq 2\epsilon$
4: while the minimal $\Delta_i$ is (strictly) less than $2\epsilon$ do
5:     $\boldsymbol{z} \leftarrow (z_0, z_1, \ldots, z_{i-2}, z_{i+1}, \ldots, z_m)$    ▷ See (b)
6: end while
7: $\boldsymbol{z} \leftarrow (z_0, z_1 - u, z_2 - u, \ldots, z_m - u - v)$    ▷ Reverse line 3
8: if $\Delta_1 < \epsilon$ then $\boldsymbol{z} \leftarrow (z_0, z_2, z_3, \ldots, z_m)$    ▷ Ensure $\Delta_1 \geq \epsilon$
9: if $\Delta_m < \epsilon$ then $\boldsymbol{z} \leftarrow (z_0, z_1, \ldots, z_{m-2}, z_m)$    ▷ Ensure $\Delta_m \geq \epsilon$
```

(a) ϵ-Pruning Algorithm



(b) Regularise

Fig. 4. ϵ-Pruning Algorithm and its main regularisation

Lemma 3. *Let $\boldsymbol{z}^\circ$ and $\boldsymbol{z}^\dagger$ be the sequences before and after line 5. Then*

$$G^\dagger - G^\circ \;\geq\; (m^\circ - m^\dagger) \min\bigl\{0, \ell(2\epsilon) - \epsilon\bigr\}$$

Proof. Let i be the index of the minimal Δ_i°. Let $l = \Delta_{i-1}^\circ$, $c = \Delta_i^\circ$ and $r = \Delta_{i+1}^\circ$, so that $\Delta_{i-1}^\dagger = l + r - c$ and $2\epsilon > c \leq l, r$. By definition $G^\dagger - G^\circ$ equals

$$\begin{aligned} &\bigl(l + r - c - \ell(l + r - c)\bigr) - \bigl(l + r - \ell(l) - \ell(c) - \ell(r)\bigr) && \text{if } z_{i-1} \leq z_i\text{, or} \\ &\bigl(-\ell(l + r - c)\bigr) - \bigl(c - \ell(l) - \ell(c) - \ell(r)\bigr) && \text{if } z_{i-1} \geq z_i. \end{aligned}$$

In either case $G^\dagger - G^\circ$ simplifies to $-c - \ell(l + r - c) + \ell(l) + \ell(c) + \ell(r)$. Since ℓ is concave, the worst-case values for l and r are c. For the same reason, the worst-case value for c is either 0 or 2ϵ. Then since ℓ is nonnegative

$$G^\dagger - G^\circ \;\geq\; 2\ell(c) - c \;\geq 2 \min\{\ell(0), \ell(2\epsilon) - \epsilon\} \;\geq 2 \min\{0, \ell(2\epsilon) - \epsilon\}. \qquad \square$$

Now fix $\epsilon \geq 0$. Let $\boldsymbol{z}^\epsilon = z_0^\epsilon, z_1^\epsilon, \ldots, z_{m^\epsilon}^\epsilon$ be the result of applying Algorithm 4a with parameter ϵ to the sequence $\boldsymbol{z}$ of local extrema of Λ, and let Λ^ϵ be any continuous function with local extrema $\boldsymbol{z}^\epsilon$. By construction Λ^ϵ has ϵ-separation and regularises Λ. Theorem 3 gives us a bound on the payoff in terms of Λ^ϵ. We now show how to get a bound in terms of the original Λ.

Theorem 4 (Enforcing ϵ-Separation). *For all $\epsilon \geq 0$ such that $\ell(2\epsilon) < \epsilon$*

$$G^\epsilon - G \;\geq\; \bigl(m - m^\epsilon\bigr)\bigl(\ell(2\epsilon) - \epsilon\bigr) - 2\ell(2\epsilon).$$

Proof. Let $\boldsymbol{z}^+, \boldsymbol{z}^\star, \boldsymbol{z}^-$ be the sequences after lines 3, 6 and 7 of Algorithm 4a. Thus the algorithm produces (denoted $\rightarrow$) in order

$$\boldsymbol{z} \;\rightarrow\; \boldsymbol{z}^+ \;\rightarrow\; \boldsymbol{z}^\star \;\rightarrow\; \boldsymbol{z}^- \;\rightarrow\; \boldsymbol{z}^\epsilon.$$

with numbers of blocks $m = m^+ \geq m^\star = m^- \geq m^\epsilon$. By Lemma 3 $G^\star - G^+ \geq \bigl(m^+ - m^\star\bigr)\bigl(\ell(2\epsilon) - \epsilon\bigr)$. It thus remains to show that

$$(G^+ - G) + (G^- - G^\star) + (G^\epsilon - G^-) \;\geq\; (m^\star - m^\epsilon)\bigl(\ell(2\epsilon) - \epsilon\bigr) - 2\ell(2\epsilon).$$

We have $G^+ - G = g_1^+ - g_1 + g_{m^+}^+ - g_m$, $G^- - G^\star = g_1^- - g_1^\star + g_{m^-}^- - g_{m^-}^\star$ and

$$G^\epsilon - G^- = \begin{cases} g_1^\epsilon - g_1^- - g_2^- & \text{if } \Delta_1^- < \epsilon, \\ 0 & \text{otherwise,} \end{cases} + \begin{cases} g_{m^\epsilon}^\epsilon - g_{m^-}^- - g_{m^- -1}^- & \text{if } \Delta_{m^-}^- < \epsilon, \\ 0 & \text{otherwise.} \end{cases}$$

These three expressions are symmetric in the first and last element of the sequences concerned. The contributions of the first elements are

$$\begin{aligned} g_1^+ - g_1 &= u_+ - \ell\big(\Delta_1 + |u|\big) + \ell(\Delta_1), && (3) \\ g_1^- - g_1^\star &= -u_+ - \ell(\Delta_1^-) + \ell\big(\Delta_1^- + |u|\big), && (4) \\ g_1^\epsilon - g_1^- - g_2^- &= -\Delta_1^- + \ell(\Delta_1^-) + \ell(\Delta_2^-) - \ell(\Delta_2^- - \Delta_1^-). && (5) \end{aligned}$$

If $\Delta_1^- \geq \epsilon$ then no element is dropped in line 8. The sum of (3) and (4) equals

$$-\ell\big(\Delta_1 + |u|\big) + \ell(\Delta_1) - \ell(\Delta_1^-) + \ell\big(\Delta_1^- + |u|\big) \;\geq\; -\ell(2\epsilon).$$

Since ℓ increases the last two terms are positive and can be dropped from the bound; the remaining expression is increasing in Δ_1 by concavity of ℓ and is decreasing in $|u|$. Substitute the worst-case values $\Delta_1 = 0$ and $|u| = 2\epsilon$.

If on the other hand $\Delta_1^- < \epsilon$ then one element was dropped in line 8. In this case the sum of (3)–(5) equals

$$-\ell\big(\Delta_1 + |u|\big) + \ell\big(\Delta_1^- + |u|\big) + \ell(\Delta_1) - \Delta_1^- + \ell(\Delta_2^-) - \ell(\Delta_2^- - \Delta_1^-) \;\geq\; -\Delta_1^-.$$

The lower bound is obtained by cancelling the first two terms and the last two terms since ℓ is increasing and $0 \leq \Delta_1 \leq \Delta_1^-$. Since ℓ is nonnegative, we omit the third term as well. We then use $-\Delta_1^- \;\geq\; -\epsilon \;=\; \big(\ell(2\epsilon) - \epsilon\big) - \ell(2\epsilon)$.

The bound for the contribution of the final elements is analogous. In each case, a dropped intermediate elements contributes at most $\ell(2\epsilon) - \epsilon$, while the borders lose at most $\ell(2\epsilon)$ each. □

We now put everything together, and in particular we optimise the value of ϵ.

Theorem 5 (Payoff Bound). *Fix logprice functions Λ and Λ', the latter with associated $\boldsymbol{z}'$, m' and $\boldsymbol{\Delta}'$ as in Definition 1. If $\Lambda' \preccurlyeq \Lambda$ then*

$$\pi{*}\Lambda \;\geq\; \sum_{1 \leq i \leq m'} (z_i' - z_{i-1}')_+ - \sum_{1 \leq i \leq m'} \ell(\Delta_i') - (m' - 1)c_\pi - \ln 2 - 2\epsilon_\pi,$$

where ϵ_π is the unique solution to $\pi(2\epsilon) = \frac{1}{\mathrm{e}^\epsilon - 1}$, and $c_\pi = -\ln(1 - \mathrm{e}^{-\epsilon_\pi})$.

Proof. For each $\epsilon \geq 0$ with $\ell(2\epsilon) < \epsilon$

$$\begin{aligned} \pi{*}\Lambda \;\geq\; \pi{*}\Lambda' \;\geq\; \pi{*}\Lambda^\epsilon \;&\geq\; G^\epsilon + (m^\epsilon - 1)\ln(1 - \mathrm{e}^{-\epsilon}) - \ln 2 \\ &\geq\; G' + (m^\epsilon - 1)\ln(1 - \mathrm{e}^{-\epsilon}) + (m' - m^\epsilon)\big(\ell(2\epsilon) - \epsilon\big) - 2\ell(2\epsilon) - \ln 2 \\ &\geq\; G' + (m' - 1)\min\big\{\ln(1 - \mathrm{e}^{-\epsilon}), \ell(2\epsilon) - \epsilon\big\} - 2\ell(2\epsilon) - \ln 2\;. \end{aligned}$$

The inequalities are twice Theorem 2, then Theorem 3, then Theorem 4. To complete the proof we set ϵ to equalise the arguments of the minimum. □

Typical values for ϵ_π and c_π are shown in Table 1.

4 Implementation

The following algorithm implements the strategy π. For arbitrary prior densities it runs in $O(T^2)$ time. For exponential priors, we reduce the running time to $O(T)$. The key to efficiency is the independent product form of π, which renders the *last* switching price a sufficient statistic.

For concreteness, we measure discrete time in days. As its data structure, the algorithm maintains a set of bank accounts. Each bank account has a *balance*, a *type* that is either A or B, and a *birthday*. The balance of type A accounts is measured in shares, whereas that of type B accounts is measured in cash.

On day zero the initial unit cash is divided evenly into two bank accounts: one account of type B with half a unit of cash, and one account of type A with $\frac{1}{2}\exp(-\Lambda(0))$ shares, i.e. half a unit of cash worth of shares at the initial logprice.

The algorithm then proceeds as follows. Each day $t = 1, 2, \ldots$ the new price $\Lambda(t)$ is announced. The algorithm creates a single new bank account with birthday t. If $\Lambda(t-1) \leq \Lambda(t)$, then the new account is of type B, and a portion of the shares in existing accounts of type A is sold to fill it with cash. On the other hand if $\Lambda(t-1) \geq \Lambda(t)$, then a new account of type A is endowed with shares by investing a fraction of the capital of existing accounts of type B. In either case, the amount traded reestablishes the following invariant. At the end of day t:

- Each account of type A that was created with c shares on birthday i has balance $c\int_\lambda^\infty \pi(h)\,\mathrm{d}h$, where $\lambda = \max_{i\leq j\leq t} \Lambda|_i^j$.
- Each account of type B that was created with capital c on birthday i has balance $c\int_\lambda^\infty \pi(h)\,\mathrm{d}h$, where $\lambda = \max_{i\leq j\leq t} -\Lambda|_i^j$.

To see how this works, consider an A-type account with birthday i and initial balance c, and assume that the invariant was maintained at the end of day $t-1$. First, note that it can only become violated if the maximum changes, that is, if $\Lambda|_i^t$ exceeds the previous maximum $\lambda = \max_{i\leq j<t} \Lambda|_i^j$. Then the balance still is $c\int_\lambda^\infty \pi(h)\,\mathrm{d}h$ but should become $c\int_{\Lambda|_i^t}^\infty \pi(h)\,\mathrm{d}h$. The fraction

$$1 - \frac{\int_{\Lambda|_i^t}^\infty \pi(h)\,\mathrm{d}h}{\int_\lambda^\infty \pi(h)\,\mathrm{d}h} = \frac{\int_\lambda^{\Lambda|_i^t} \pi(h)\,\mathrm{d}h}{\int_\lambda^\infty \pi(h)\,\mathrm{d}h} = \pi\big(H \leq \Lambda|_i^t \big| H \geq \lambda\big) \tag{6}$$

of the balance must be sold to reestablish the invariant, and the resulting cash is transferred to the new account. Note that we only query π via its cumulative distribution function.

Complexity Analysis. After t days, there are $t+2$ bank accounts to maintain, and each bank account potentially requires work each round. Thus, trading for T days takes $O(T^2)$ time and $O(T)$ space.

For exponential priors we can do better by *merging* several bank accounts into a single account with the sum of their balances. This is because for memoryless priors, the fraction (6) to be traded away does not depend on the birthday i, but only on the maximum λ, allowing us to merge bank accounts with the same maximum. Now observe that all bank accounts that are tapped to reestablish the invariant share the same maximum afterwards, and can hence all be merged. This means that a bank account requires work at most *once*, namely when it is merged away. By maintaining two stacks of bank accounts, one for each type, each ordered by the maximum λ, the running time is brought down to $O(T)$. Since we do not know *when* merges happen, the space requirement is still $O(T)$, and the running time is *amortised* $O(1)$ per day.

5 Conclusion

We presented a simple online algorithm that can be applied to two-way trading, but also to prediction with expert advice, data compression and hypothesis testing (see Section 1). Compared to the many hedging algorithms described in the literature, our approach has two novel properties. First, the overhead of our algorithm is independent of the times at which prices are processed, and second, our bound is free of any conditions on the evolution of the price of the asset, and is parameterised either by the asset price function itself or by a regularised model of it.

The surprisingly simple implementation (Section 4) processes a sequence of T asset prices in $O(T^2)$ time and $O(T)$ space. The algorithm models the scale of the fluctuations of the price using a density function on $[0, \infty)$; if an exponential density is employed, the running time is reduced to $O(T)$.

References

[1] Berger, J.O.: Could Fisher, Jeffreys and Neyman have agreed on testing? Statistical Science 18(1), 1–32 (2003)

[2] Chen, G.H., Kao, M.Y., Lyuu, Y.D., Wong, H.K.: Optimal buy-and-hold strategies for financial markets with bounded daily returns. In: Proc. of the 31st Annual ACM Symposium on Theory of Computing, pp. 119–128. ACM, New York (1999)

[3] Cover, T.M., Thomas, J.A.: Elements of Information Theory. John Wiley, Chichester (1991)

[4] Dannoura, E., Sakurai, K.: An improvement on El-Yaniv-Fiat-Karp-Turpin's money-making bi-directional trading strategy. IPL 66(1), 27–33 (1998)

[5] Dawid, A.P., De Rooij, S., Shafer, G., Shen, A., Vereshchagin, N., Vovk, V.: Insuring against loss of evidence in game-theoretic probability, arXiv:1005.1811

[6] DeMarzo, P., Kremer, I., Mansour, Y.: Online trading algorithms and robust option pricing. In: Proc. of the 38 Annual ACM Symposium on Theory of Computing, pp. 477–486. ACM, New York (2006)

[7] El-Yaniv, R., Fiat, A., Karp, R.M., Turpin, G.: Optimal search and one-way trading online algorithms. Algorithmica 30(1), 101–139 (2001)

[8] Van Erven, T., Grünwald, P.D., De Rooij, S.: Catching up faster by switching sooner: a prequential solution to the AIC-BIC dilemma (2008) (submitted); Preprint available as arXiv:0807.1005

[9] Herbster, M., Warmuth, M.K.: Tracking the best expert. Machine Learning 32, 151–178 (1998)

[10] Koolen, W.M., De Rooij, S.: Combining expert advice efficiently. In: Proc. of the 21st Annual Conference on Learning Theory, pp. 275–286 (2008)
[11] Shafer, G., Shen, A., Vereshchagin, N., Vovk, V.: Test martingales, Bayes factors, and p-values, arXiv:0912.4269
[12] Volf, P., Willems, F.: Switching between two universal source coding algorithms. In: Proc. of the Data Compression Conference, Snowbird, Utah, pp. 491–500. IEEE Computer Society Press, Los Alamitos (1998)
[13] Vovk, V.: A game of prediction with expert advice. Journal of Computer and System Sciences 56, 153–173 (1998)

Prediction with Expert Advice under Discounted Loss

Alexey Chernov and Fedor Zhdanov

Computer Learning Research Centre and Department of Computer Science,
Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK
{chernov,fedor}@cs.rhul.ac.uk

Abstract. We study prediction with expert advice in the setting where the losses are accumulated with some discounting and the impact of old losses can gradually vanish. We generalize the Aggregating Algorithm and the Aggregating Algorithm for Regression, propose a new variant of exponentially weighted average algorithm, and prove bounds on the cumulative discounted loss.

1 Introduction

Prediction with expert advice is a framework for online sequence prediction. Predictions are made step by step. The quality of each prediction (the discrepancy between the prediction and the actual outcome) is evaluated by a real number called loss. The losses are accumulated over time. In the standard framework for prediction with expert advice (see the monograph [2] for a comprehensive review), the losses from all steps are just summed. In this paper, we consider a generalization where older losses can be devalued; in other words, we use discounted cumulative loss.

Predictions are made by Experts and Learner according to Protocol 1. In this protocol, Ω is the set of possible outcomes and $\omega_1, \omega_2, \omega_3 \ldots$ is the sequence to predict; Γ is the set of admissible predictions, and $\lambda\colon \Gamma \times \Omega \to [0, \infty]$ is the loss function. The triple $(\Omega, \Gamma, \lambda)$ specifies the game of prediction. The most common examples are the binary square loss, log loss, and absolute loss games.

Protocol 1. Prediction with expert advice under general discounting

```
$\mathcal{L}_0 := 0$.
$\mathcal{L}_0^\theta := 0$, $\theta \in \Theta$.
for $t = 1, 2, \ldots$ do
    Accountant announces $\alpha_{t-1} \in (0, 1]$.
    Experts announce $\gamma_t^\theta \in \Gamma$, $\theta \in \Theta$.
    Learner announces $\gamma_t \in \Gamma$.
    Reality announces $\omega_t \in \Omega$.
    $\mathcal{L}_t^\theta := \alpha_{t-1}\mathcal{L}_{t-1}^\theta + \lambda(\gamma_t^\theta, \omega_t)$, $\theta \in \Theta$.
    $\mathcal{L}_t := \alpha_{t-1}\mathcal{L}_{t-1} + \lambda(\gamma_t, \omega_t)$.
end for
```

© Springer-Verlag Berlin Heidelberg 2010

They have $\Omega = \{0,1\}$ and $\Gamma = [0,1]$, and their loss functions are $\lambda^{\mathrm{sq}}(\gamma,\omega) = (\gamma-\omega)^2$, $\lambda^{\log}(\gamma,0) = -\log(1-\gamma)$ and $\lambda^{\log}(\gamma,1) = -\log\gamma$, $\lambda^{\mathrm{abs}}(\gamma,\omega) = |\gamma-\omega|$, respectively.

The players in the game of prediction are Experts θ from some pool Θ, Learner, and also Accountant and Reality. We are interested in (worst-case optimal) strategies for Learner, and thus the game can be regarded as a two-player game, where Learner opposes the other players. The aim of Learner is to keep his total loss $\mathcal{L}_t$ small as compared to the total losses $\mathcal{L}_t^\theta$ of all experts $\theta \in \Theta$.

The standard protocol of prediction with expert advice (as described in [18, 19]) is a special case of Protocol 1 where Accountant always announces $\alpha_t = 1$, $t = 0,1,2,\ldots$. The new setting gives some more freedom to Learner's opponents.

Another important special case of Protocol 1 is the exponential (geometric) discounting $\alpha_t = \alpha \in (0,1)$ for all $t = 0,1,2,\ldots$. Exponential discounting is widely used in finance and economics (see, e. g., [15]), time series analysis (see, e. g., [9]), reinforcement learning [17], and other applications. In the context of prediction with expert advice, Freund and Hsu [7] noted that the discounted loss provides an alternative to "tracking the best expert" framework [12]. Indeed, an exponentially discounted sum depends almost exclusively on the last $O(\log(1/\alpha))$ terms. If the expert with the best one-step performance changes at this rate, then Learner observing the α-discounted losses will mostly follow predictions of the current best expert. Under our more general discounting, more subtle properties of best expert changes may be specified by varying the discount factor. In particular, one can cause Learner to "restart mildly" giving $\alpha_t = 1$ (or $\alpha_t \approx 1$) most of the time and $\alpha_t \ll 1$ at crucial moments.

Our discounting scheme has a straightforward financial interpretation, if we interpret λ as gains rather than losses. (These interpretations are interchangeable in the case of bounded loss functions if we consider bounds in terms of the number of steps and not of the minimal loss.) We save our gains in cash, and there is inflation of this currency. Inflation $n\%$ in period t corresponds to $\alpha_t = 1/(1+n/100)$. Then $\mathcal{L}_T$ is the value of our total savings expressed in some conventional currency corrected for inflation or in quantities of some goods.

Cesa-Bianchi and Lugosi [2, § 2.11] discuss another kind of discounting

$$L_T = \sum_{t=1}^{T} \beta_{T-t} l_t\,, \tag{1}$$

where l_t are one-step losses and β_t are some decreasing discount factors. To see the difference, let us rewrite our definition in the same style:

$$\mathcal{L}_T = \alpha_{T-1}\mathcal{L}_{T-1} + l_T = \sum_{t=1}^{T} \alpha_t \cdots \alpha_{T-1} l_t = \frac{1}{\beta_T}\sum_{t=1}^{T} \beta_t l_t\,, \tag{2}$$

where $\beta_t = 1/\alpha_1 \cdots \alpha_{t-1}$, $\beta_1 = 1$. The sequence β_t is *non*-decreasing, $\beta_1 \le \beta_2 \le \beta_3 \le \ldots$; but it is applied "in the reverse order" compared to (1). So, in both definitions, the older losses are the less weight they are ascribed. However, according to (1), the losses l_t have different relative weights in L_T, L_{T+1} and so on, whereas (2) fixes the relative weight of l_t with respect to all previous losses forever starting from the moment t. The latter property allows us to get uniform algorithms for Learner with loss guarantees that hold for all $T = 1, 2, \ldots$; in contrast, Theorem 2.8 in [2] gives a guarantee only at one moment T chosen in advance. The only kind of discounting that can be expressed both as (1) and as (2) is the exponential discounting $\sum_{t=1}^{T} \alpha^{T-t} l_t$. Under this discounting, NormalHedge algorithm is analysed in [7]; we briefly compare the obtained bounds in Section 3.

The rest of the paper is organized as follows. In Section 2, we propose a generalization of the Aggregating Algorithm [19] and prove the same bound as in [19] but for the discounted loss. In Section 3, we consider convex loss functions and propose an algorithm that extends the Weak Aggregating Algotihm [14] and the exponentially weighted average forecaster with time-varying learning rate [2, § 2.3], with a similar loss bound. In Section 4, we consider the use of prediction with expert advice for the regression problem and adapt the Aggregating Algorithm for Regression [21] (applied to spaces of linear functions and to reproducing kernel Hilbert spaces) to the discounted square loss. All our algorithms are inspired by the methodology of defensive forecasting [4]. We do not explicitly use or refer to this technique here. However, we need it in the full version [6] of this paper (for Theorem 3), and also illustrate the ideas of defensive forecasting on the regression task in Appendix A.2 of [6]; Appendix A.1 contains some proofs omitted here.

2 Linear Bounds for Learner's Loss

In this section, we assume that the set of experts is finite, $\Theta = \{1, \ldots, K\}$, and show how Learner can achieve a bound of the form $\mathcal{L}_t \le c\mathcal{L}_t^k + (c \ln K)/\eta$ for all Experts k, where $c \ge 1$ and $\eta > 0$ are constants. Bounds of this kind were obtained in [18]. Loosely speaking, such a bound holds for certain c and η if and only if the game $(\Omega, \Gamma, \lambda)$ has the following property:

$$\exists \gamma \in \Gamma \, \forall \omega \in \Omega \quad \lambda(\gamma, \omega) \le -\frac{c}{\eta} \ln \left(\sum_{i \in I} p_i \mathrm{e}^{-\eta \lambda(\gamma_i, \omega)} \right) \tag{3}$$

for any finite index set I, for any $\gamma_i \in \Gamma$, $i \in I$, and for any $p_i \in [0, 1]$ such that $\sum_{i \in I} p_i = 1$. This property turns out to be sufficient for the discounted case too.

Theorem 1. *Suppose that the game $(\Omega, \Gamma, \lambda)$ satisfies condition (3) for certain $c \ge 1$ and $\eta > 0$. In the game played according to Protocol 1, Learner has a strategy guaranteeing that, for any T and for any $k \in \{1, \ldots, K\}$, it holds*

$$\mathcal{L}_T \le c\mathcal{L}_T^k + \frac{c \ln K}{\eta} \,. \tag{4}$$

For the standard undiscounted case (Accountant announces $\alpha_t = 1$ at each step t), this theorem was proved by Vovk in [18] with the help of the Aggregating

Algorithm (AA) as Learner's strategy. It is known ([11, 19]) that this bound is asymptotically optimal for large pools of Experts (for games satisfying some assumptions): if the game does not satisfy (3) for some $c \geq 1$ and $\eta > 0$, then, for sufficiently large K, there is a strategy for Experts and Reality (recall that Accountant always says $\alpha_t = 1$) such that Learner cannot secure (4). For the special case of $c = 1$, bound (4) is tight for any fixed K as well [20]. These results imply optimality of Theorem 1 in the new setting with general discounting (when we allow arbitrary behaviour of Accountant with the only requirement $\alpha_t \in (0, 1]$). However, they leave open the question of lower bounds under different discounting assumptions (that is, when Accountant moves are fixed); a particularly interesting case is the exponential discounting $\alpha_t = \alpha \in (0, 1)$.

Proof. As Learner's strategy we exploit a minor modification of the Aggregating Algorithm, the AA with Discounting (AAD). The pseudocode is given as Algorithm 1.

Algorithm 1. Aggregating algorithm with discounting

```
1: Initialize weights of Experts $w_0^k := 1$, $k = 1, \dots, K$.
2: for $t = 1, 2, \dots$ do
3:     Get discount $\alpha_{t-1} \in (0, 1]$.
4:     Get Experts' predictions $\gamma_t^k \in \Gamma, k = 1, \dots, K$.
5:     Calculate $g_t(\omega) = -\frac{c}{\eta} \ln \left( \sum_{k=1}^K \frac{1}{K} (w_{t-1}^k)^{\alpha_{t-1}} \mathrm{e}^{-\eta\lambda(\gamma_t^k, \omega)} \right)$, for all $\omega \in \Omega$.
6:     Output $\gamma_t := \sigma(g_t) \in \Gamma$.
7:     Get $\omega_t \in \Omega$.
8:     Update the weights $w_t^k := (w_{t-1}^k)^{\alpha_{t-1}} \mathrm{e}^{\eta\lambda(\gamma_t, \omega_t)/c - \eta\lambda(\gamma_t^k, \omega_t)}$, $k = 1, \dots, K$.
9: end for.
```

The algorithm has three parameters, which depend on the game $(\Omega, \Gamma, \lambda)$: $c \geq 1$, $\eta > 0$, and a function $\sigma \colon \mathbb{R}^\Omega \to \Gamma$. The function σ is called a *substitution function* and must have the following property: $\lambda(\sigma(g), \omega) \leq g(\omega)$ for all $\omega \in \Omega$ if for $g \in \mathbb{R}^\Omega$ there exists any $\gamma \in \Gamma$ such that $\lambda(\gamma, \omega) \leq g(\omega)$ for all $\omega \in \Omega$. A natural example of substitution function is given by

$$\sigma(g) = \arg\min_{\gamma \in \Gamma} \bigl(\lambda(\gamma, \omega) - g(\omega)\bigr) \tag{5}$$

(if the minimum is attained in several points, one can take any of them). An advantage of this σ is that one can use in line 8 of the algorithm the update rule $w_t^k := (w_{t-1}^k)^{\alpha_{t-1}} \mathrm{e}^{-\eta\lambda(\gamma_t^k, \omega_t)}$, which does not contain Learner's losses. Indeed, multiplying all w_t^k by a constant (independent of k), we add to all $g_t(\omega)$ a constant (independent of ω), and $\sigma(g_t)$ does not change.

Assume that c and η are such that condition (3) holds for the game. Now let us show by induction over t that Algorithm 1 preserves the following condition:

$$\sum_{k=1}^K \frac{1}{K} w_t^k \leq 1 \, . \tag{6}$$

Indeed, assume that $\sum_{k=1}^K w_{t-1}^k/K \le 1$. Then we have $\sum_{k=1}^K (w_{t-1}^k)^{\alpha_{t-1}}/K \le \left(\sum_{k=1}^K w_{t-1}^k/K\right)^{\alpha_{t-1}} \le 1$, since the function $x \mapsto x^\alpha$ is concave and monotone for $\alpha \in (0,1]$ and $x \ge 0$.

Let $\tilde{w}^k$ be any reals such that $\tilde{w}^k \ge (w_{t-1}^k)^{\alpha_{t-1}}/K$ and $\sum_{k=1}^K \tilde{w}^k = 1$ (for example, $\tilde{w}^k = (w_{t-1}^k)^{\alpha_{t-1}} \Big/ \sum_{j=1}^K (w_{t-1}^j)^{\alpha_{t-1}}$). Due to condition (3), there exists $\gamma \in \Gamma$ such that for all $\omega \in \Omega$ we have $\lambda(\gamma,\omega) \le -\frac{c}{\eta}\ln\left(\sum_{k=1}^K \tilde{w}^k e^{-\eta\lambda(\gamma_t^k,\omega)}\right) \le -\frac{c}{\eta}\ln\left(\sum_{k=1}^K \frac{1}{K}(w_{t-1}^k)^{\alpha_{t-1}} e^{-\eta\lambda(\gamma_t^k,\omega)}\right) = g_t(\omega)$ (the second inequality holds due to our choice of $\tilde{w}^k$). Thus, due to the property of σ, we have $\lambda(\gamma_t,\omega) \le g_t(\omega)$ for all $\omega \in \Omega$. In particular, this holds for $\omega = \omega_t$, and we get (6).

To get the loss bound (4), it remains to note that $\ln w_t^k = \eta\left(\mathcal{L}_t/c - \mathcal{L}_t^k\right)$. Indeed, this is trivial for $t = 0$. If this holds for w_{t-1}^k, then $\ln w_t^k = \alpha_{t-1}\ln(w_{t-1}^k) + \eta\lambda(\gamma_t,\omega_t)/c - \eta\lambda(\gamma_t^k,\omega_t) = \alpha_{t-1}\eta\left(\mathcal{L}_{t-1}/c - \mathcal{L}_{t-1}^k\right) + \eta\lambda(\gamma_t,\omega_t)/c - \eta\lambda(\gamma_t^k,\omega_t) = \eta\left((\alpha_{t-1}\mathcal{L}_{t-1} + \lambda(\gamma_t,\omega_t))/c - (\alpha_{t-1}\mathcal{L}_{t-1}^k + \lambda(\gamma_t^k,\omega_t))\right) = \eta\left(\mathcal{L}_t/c - \mathcal{L}_t^k\right)$. Thus, condition (6) is equivalent to

$$\sum_{k=1}^K \frac{1}{K} e^{\eta\left(\mathcal{L}_t/c - \mathcal{L}_t^k\right)} \le 1\,, \tag{7}$$

and (4) follows by lower-bounding the sum by any of the additive terms. □

Remark 1. Everything in this section remains valid, if we replace the equal initial Experts' weights $1/K$ by arbitrary non-negative weights w^k, $\sum_{k=1}^K w^k = 1$. This leads to a variant of (4), where the last additive term is replaced by $\frac{c}{\eta}\ln\frac{1}{w^k}$. Additionally, we can consider any measurable space Θ of Experts and a non-negative weight function $w(\theta)$, and replace sums over K by integrals over Θ. Then the algorithm and its analysis remain valid (if we impose natural integrability conditions on Experts' predictions γ_t^θ; see [21] for more detailed discussion)—this will be used in Section 4.

3 Learner's Loss in Bounded Convex Games

For many games (for example, the absolute loss game), condition (3) does not hold with $c = 1$ (for any $\eta > 0$), and one cannot get a bound of the form $\mathcal{L}_t \le \mathcal{L}_t^k + O(1)$. Since Experts' losses $\mathcal{L}_T^\theta$ may grow as T in the worst case, any bound of the form (4) with $c > 1$ may be loose and Learner's loss may exceed an Expert's loss by δT for some constant $\delta > 0$. However, for a large class of interesting games (including the absolute loss game), one can obtain guarantees of the form $\mathcal{L}_T \le \mathcal{L}_T^k + O(\sqrt{T})$ in the undiscounted case. In this section, we prove an analogous result for the discounted setting.

A game $(\Omega, \Gamma, \lambda)$ is non-empty if Ω and Γ are non-empty. The game is called *bounded* if $L = \max_{\omega,\gamma}\lambda(\gamma,\omega) < \infty$. One may assume that $L = 1$ (if not, consider the scaled loss function λ/L). The game is called *convex* if for any predictions $\gamma_1,\ldots,\gamma_M \in \Gamma$ and for any weights $p_1,\ldots,p_M \in [0,1]$, $\sum_{m=1}^M p_m = 1$, there exists $\gamma \in \Gamma$ such that $\lambda(\gamma, \omega) \le \sum_{m=1}^{M} p_m \lambda(\gamma_m, \omega)$ for all $\omega \in \Omega$. Note that if Γ is a convex set (e. g., $\Gamma = [0, 1]$) and $\lambda(\gamma, \omega)$ is convex in γ (e. g., λ^{abs}), then the game is convex.

Theorem 2. *Suppose that $(\Omega, \Gamma, \lambda)$ is a non-empty convex game and $\lambda(\gamma, \omega) \in [0, 1]$ for all $\gamma \in \Gamma$ and $\omega \in \Omega$. In the game played according to Protocol 1, Learner has a strategy guaranteeing that, for any T and for any $k \in \{1, \dots, K\}$, it holds*

$$\mathcal{L}_T \le \mathcal{L}_T^k + \sqrt{\ln K}\sqrt{\frac{B_T}{\beta_T}}\,, \tag{8}$$

where $\beta_t = 1/(\alpha_1 \cdots \alpha_{t-1})$ and $B_T = \sum_{t=1}^{T} \beta_t$.

Note that B_T/β_T is the maximal predictors' loss, which incurs when the predictor suffers the maximal possible loss $l_t = 1$ at each step. In the undiscounted case, $\alpha_t = 1$, thus $\beta_t = 1$, $B_T = T$, and (8) becomes $\mathcal{L}_T \le \mathcal{L}_T^k + \sqrt{T \ln K}$. A similar bound (but with worse constant $\sqrt{2}$ instead of 1 before $\sqrt{T \ln K}$) is obtained in [2, Theorem 2.3]: $\mathcal{L}_T \le \mathcal{L}_T^k + \sqrt{2T \ln K} + \sqrt{(\ln K)/8}$. For the exponential discounting $\alpha_t = \alpha$, we have $\beta_t = \alpha^{-t+1}$ and $B_T = (1-\alpha^{-T})/(1-1/\alpha)$, and (8) transforms into $\mathcal{L}_T \le \mathcal{L}_T^k + \sqrt{\ln K}\sqrt{(1-\alpha^T)/(1-\alpha)} \le \mathcal{L}_T^k + \sqrt{(\ln K)/(1-\alpha)}$. A similar bound (with worse constants) is obtained in [7] for NormalHedge: $\mathcal{L}_T \le \mathcal{L}_T^k + \sqrt{(8 \ln 2.32K)/(1-\alpha)}$. The NormalHedge algorithm has an important advantage: it can guarantee the last bound without knowledge of the number of experts K (see [3] for a precise definition). We can achieve the same with the help of a more complicated algorithm but at the price of a worse bound (see also the remark after the proof).

Proof. The pseudocode of Learner's strategy is given as Algorithm 2. It contains a constant $a > 0$, which we will choose later in the proof.

Similarly to Theorem 1, let us show by induction over t that Algorithm 2 always can find γ in lines 6–7 and preserves the following condition:

$$\sum_{k=1}^{K} \frac{1}{K} w_t^k \le 1\,. \tag{9}$$

First note that $\alpha_{t-1}\eta_t/\eta_{t-1} \le 1$ for $\eta_t = a\sqrt{\beta_t/B_t}$. Indeed, substituting $\alpha_{t-1} = \beta_{t-1}/\beta_t$, we get $\alpha_{t-1}(\eta_t/\eta_{t-1}) = (\beta_{t-1}/\beta_t)\left(a\sqrt{\beta_t/B_t}/a\sqrt{\beta_{t-1}/B_{t-1}}\right) = \sqrt{(\beta_{t-1}/\beta_t)(B_{t-1}/B_t)} = \sqrt{\alpha_{t-1}}\sqrt{B_{t-1}/(B_{t-1}+\beta_t)} \le 1$.

Assume that $\sum_{k=1}^{K} w_{t-1}^k/K \le 1$. By the same argument as in Theorem 1, we can deduce that $\sum_{k=1}^{K} (w_{t-1}^k)^{\alpha_{t-1}\eta_t/\eta_{t-1}}/K \le \left(\sum_{k=1}^{K} w_{t-1}^k/K\right)^{\alpha_{t-1}\eta_t/\eta_{t-1}} \le 1$.

Let $\tilde{w}^k$ be any reals such that $\tilde{w}^k \ge (w_{t-1}^k)^{\alpha_{t-1}\eta_t/\eta_{t-1}}/K$ and $\sum_{k=1}^{K} \tilde{w}^k = 1$. By the Hoeffding inequality (see, e. g., [2, Lemma 2.2]), we have for any $\omega \in \Omega$

$$\ln\left(\sum_{k=1}^{K} \tilde{w}^k e^{-\eta_t \lambda(\gamma_t^k, \omega)}\right) \le -\eta_t \sum_{k=1}^{K} \tilde{w}^k \lambda(\gamma_t^k, \omega) + \frac{\eta_t^2}{8}\,, \tag{10}$$

Algorithm 2. Learner's strategy for convex games

```
1: Initialize weights of Experts $w_0^k := 1$, $k = 1, \dots, K$.
   Set $\beta_1 = 1$, $B_0 = 0$.
2: for $t = 1, 2, \dots$ do
3:     Get discount $\alpha_{t-1} \in (0,1]$; update $\beta_t = \beta_{t-1}/\alpha_{t-1}$, $B_t = B_{t-1} + \beta_t$.
4:     Compute $\eta_t = a\sqrt{\beta_t / B_t}$.
5:     Get Experts' predictions $\gamma_t^k \in \Gamma$, $k = 1, \dots, K$.
6:     Find $\gamma \in \Gamma$ s.t. for all $\omega \in \Omega$
7:         $\lambda(\gamma, \omega) \le -\frac{1}{\eta_t} \ln \left( \sum_{k=1}^K \frac{1}{K} \left(w_{t-1}^k\right)^{\alpha_{t-1}\eta_t/\eta_{t-1}} \mathrm{e}^{-\eta_t \lambda(\gamma_t^k, \omega) - \eta_t^2/8} \right)$.
8:     Output $\gamma_t := \gamma$.
9:     Get $\omega_t \in \Omega$.
10:    Update the weights $w_t^k := \left(w_{t-1}^k\right)^{\alpha_{t-1}\eta_t/\eta_{t-1}} \mathrm{e}^{\eta_t \left(\lambda(\gamma_t, \omega_t) - \lambda(\gamma_t^k, \omega_t)\right) - \eta_t^2/8}$,
11:        $k = 1, \dots, K$.
12: end for.
```

Remark:
If $\lambda(\gamma, \omega)$ is convex in γ, lines 6–7 can be replaced by $\gamma = \sum_{k=1}^K \tilde{w}^k \gamma_t^k$, see (11).

since $\lambda(\gamma, \omega) \in [0, 1]$ for any $\gamma \in \Gamma$ and $\omega \in \Omega$. Since the game is convex, there exists $\gamma \in \Gamma$ such that $\lambda(\gamma, \omega) \le \sum_{k=1}^K \tilde{w}^k \lambda(\gamma_t^k, \omega)$ for all $\omega \in \Omega$. Therefore, for this γ and for all $\omega \in \Omega$ we have

$$\lambda(\gamma, \omega) \le \sum_{k=1}^K \tilde{w}^k \lambda(\gamma_t^k, \omega) \le -\frac{1}{\eta_t} \ln \left(\sum_{k=1}^K \tilde{w}^k \mathrm{e}^{-\eta_t \lambda(\gamma_t^k, \omega) - \eta_t^2/8} \right)$$
$$\le -\frac{1}{\eta_t} \ln \left(\sum \frac{1}{K} \left(w_{t-1}^k\right)^{\alpha_{t-1}\eta_t/\eta_{t-1}} \mathrm{e}^{-\eta_t \lambda(\gamma_t^k, \omega) - \eta_t^2/8} \right) \tag{11}$$

(the second inequality follows from (10), and the third inequality holds due to our choice of $\tilde{w}^k$). Thus, one can always find γ in lines 6–7 of Algorithm 2. It remains to note that the inequality in line 7 with γ_t substituted for γ and ω_t substituted for ω is equivalent to $1 \ge \sum_{k=1}^K \left(w_{t-1}^k\right)^{\alpha_{t-1}\eta_t/\eta_{t-1}} \mathrm{e}^{\eta_t \lambda(\gamma_t, \omega_t) - \eta_t \lambda(\gamma_t^k, \omega_t) - \eta_t^2/8} / K = \sum_{k=1}^K w_t^k / K$.

It is easy to check that the update rule in line 10 of the algorithm implies

$$\ln w_t^k = \eta_t \left(\mathcal{L}_t - \mathcal{L}_t^k\right) - \frac{\eta_t}{8\beta_t} \sum_{\tau=1}^t \beta_\tau \eta_\tau \, .$$

Condition (9) implies that $w_T^k \le K$ for all k and T, hence we get a loss bound

$$\mathcal{L}_T \le \mathcal{L}_T^k + \frac{\ln K}{\eta_T} + \frac{1}{8\beta_T} \sum_{t=1}^T \beta_t \eta_t \, . \tag{12}$$

It remains to estimate the last term. Since $\eta_t = a\sqrt{\beta_t / B_t}$, the following inequality helps (see [6, Appendix A.1] for the proof).

Lemma 1. *Let β_t be any reals such that $1 \le \beta_1 \le \beta_2 \le \dots$. Let $B_T = \sum_{t=1}^{T} \beta_t$. Then, for any T, it holds*

$$\frac{1}{\beta_T}\sum_{t=1}^{T}\beta_t\sqrt{\frac{\beta_t}{B_t}} \le 2\sqrt{\frac{B_T}{\beta_T}}\,.$$

Then (12) implies

$$\mathcal{L}_T \le \mathcal{L}_T^k + \frac{\ln K}{a}\sqrt{\frac{B_T}{\beta_T}} + \frac{2a}{8}\sqrt{\frac{B_T}{\beta_T}} = \mathcal{L}_T^k + \left(\frac{\ln K}{a} + \frac{a}{4}\right)\sqrt{\frac{B_T}{\beta_T}}\,.$$

Choosing $a = 2\sqrt{\ln K}$, we finally get the bound. □

Remark 2. Algorithm 2 was obtained as a modification of the "Fake Defensive Forecasting" algorithm from [5], but it turned out to be an extension of the exponentially weighted average forecaster with time-varying learning rate [2, § 2.3] and the Weak Aggregating Algorithm [14], and its analysis here follows the lines of [2, Theorem 2.2] and [13]. A more involved version of Algorithm 2 can achieve a bound for ϵ-quantile regret [3], but the analysis becomes more complicated, requires application of the supermartingale technique, and gives a worse bound (see [6]).

4 Regression with Discounted Loss

In this section we consider a task of regression, where Learner must predict "labels" $y_t \in \mathbb{R}$ for input instances $x_t \in \mathbf{X} \subseteq \mathbb{R}^n$. The predictions proceed according to Protocol 2. This task can be embedded into prediction with expert

Protocol 2. Competitive online regression

```
for $t = 1, 2, \dots$ do
    Reality announces $x_t \in \mathbf{X}$.
    Learner announces $\gamma_t \in \Gamma$.
    Reality announces $y_t \in \Omega$.
end for
```

advice if Learner competes with all functions $x \to y$ from some large class serving as a pool of (imaginary) Experts.

4.1 The Framework and Linear Functions as Experts

Let the input set be $\mathbf{X} \subseteq \mathbb{R}^n$, the set of predictions be $\Gamma = \mathbb{R}$, and the set of outcomes be $\Omega = [Y_1, Y_2]$. In this section we consider the square loss $\lambda^{\mathrm{sq}}(\gamma, y) = (\gamma - y)^2$. Learner competes with a pool of experts $\Theta = \mathbb{R}^n$ (treated as linear functionals on $\mathbb{R}^n$). Each individual expert is denoted by $\theta \in \Theta$ and predicts $\theta' x_t$ at step t.

Let us take any distribution over the experts $P(d\theta)$. It is known from [18] that (3) holds for the square loss with $c = 1$, $\eta = \frac{2}{(Y_2 - Y_1)^2}$:

$$\exists \gamma \in \Gamma \, \forall y \in \Omega = [Y_1, Y_2] \quad (\gamma - y)^2 \le -\frac{1}{\eta} \ln \left(\int_\Theta \mathrm{e}^{-\eta(\theta' x_t - y)^2} P(d\theta) \right) .$$

Denote by X the matrix of size $T \times n$ consisting of the rows of the input vectors $x_1', \dots, x_T'$. Let also $W_T = \operatorname{diag}(\beta_1/\beta_T, \beta_2/\beta_T, \dots, \beta_T/\beta_T)$, i.e., W_T is a diagonal matrix $T \times T$. In a manner similar to [21], we prove the following upper bound for Learner's loss.

Theorem 3. *For any $a > 0$, there exists a prediction strategy for Learner in Protocol 2 achieving, for every T and for any linear predictor $\theta \in \mathbb{R}^n$,*

$$\sum_{t=1}^T \frac{\beta_t}{\beta_T} (\gamma_t - y_t)^2 \le \sum_{t=1}^T \frac{\beta_t}{\beta_T} (\theta' x_t - y_t)^2 + a\|\theta\|^2 + \frac{(Y_2 - Y_1)^2}{4} \ln \det \left(\frac{X' W_T X}{a} + I \right) . \tag{13}$$

If, in addition, $\|x_t\|_\infty \le Z$ for all t, then

$$\sum_{t=1}^T \frac{\beta_t}{\beta_T} (\gamma_t - y_t)^2 \le \sum_{t=1}^T \frac{\beta_t}{\beta_T} (\theta' x_t - y_t)^2 + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln \left(\frac{Z^2}{a} \frac{\sum_{t=1}^T \beta_t}{\beta_T} + 1 \right) . \tag{14}$$

Without discounting ($\alpha_t = 1$ for all t), the bounds in the theorem coincide with the bounds for the Aggregating Algorithm for Regression [21, Theorem 1] with $Y_2 = Y$ and $Y_1 = -Y$, since, as remarked after Theorem 2, $\beta_t = 1$ and $\left(\sum_{t=1}^T \beta_t\right)/\beta_T = T$ in the case without discounting. Recall also that in the case of the exponential discounting ($\alpha_t = \alpha \in (0,1)$) we have $\beta_t = \alpha^{-t+1}$ and $\left(\sum_{t=1}^T \beta_t\right)/\beta_T = (1 - \alpha^{T-1})/(1-\alpha) \le 1/(1-\alpha)$.

4.2 Functions from an RKHS as Experts

In this section we apply the kernel trick to the linear method to compete with wider sets of experts. Each expert $f \in \mathcal{F}$ predicts $f(x_t)$. Here $\mathcal{F}$ is a reproducing kernel Hilbert space (RKHS) with a positive definite kernel $k \colon \mathbf{X} \times \mathbf{X} \to \mathbb{R}$. For the definition of RKHS and its connection to kernels see [16]. Each kernel defines a unique RKHS. We use the notation $\mathbf{K}_T = \{k(x_i, x_j)\}_{i,j=1,\dots,T}$ for the kernel matrix for the input vectors at step T. In a manner similar to [8], we prove the following upper bound on the discounted square loss of Learner.

Theorem 4. *For any $a > 0$, there exists a strategy for Learner in Protocol 2 achieving, for every positive integer T and any predictor $f \in \mathcal{F}$,*

$$\sum_{t=1}^{T} \frac{\beta_t}{\beta_T}(\gamma_t - y_t)^2 \le \sum_{t=1}^{T} \frac{\beta_t}{\beta_T}(f(x_t) - y_t)^2 + a\|f\|_{\mathcal{F}}^2 + \frac{(Y_2 - Y_1)^2}{4} \ln\det\left(\frac{\sqrt{W_T}\mathbf{K}_T\sqrt{W_T}}{a} + I\right). \quad (15)$$

Corollary 1. *Assume that $c_{\mathcal{F}}^2 = \sup_{x\in\mathbf{X}} k(x,x) < \infty$ for the RKHS $\mathcal{F}$. Under the conditions of Theorem 4, given in advance any constant $\mathcal{T}$ such that $\left(\sum_{t=1}^{T}\beta_t\right)/\beta_T \le \mathcal{T}$, one can choose parameter a such that the strategy in Theorem 4 achieves for any $f \in \mathcal{F}$*

$$\sum_{t=1}^{T} \frac{\beta_t}{\beta_T}(\gamma_t - y_t)^2 \le \sum_{t=1}^{T} \frac{\beta_t}{\beta_T}(f(x_t) - y_t)^2 + \left(\frac{(Y_2 - Y_1)^2}{4} + \|f\|_{\mathcal{F}}^2\right) c_{\mathcal{F}}\sqrt{\mathcal{T}}. \quad (16)$$

Proof. The determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Chapter 2, Theorem 7 in [1]), and thus we have

$$\ln\det\left(I + \frac{\sqrt{W_T}\mathbf{K}_T\sqrt{W_T}}{a}\right) \le T\ln\left(1 + \frac{c_{\mathcal{F}}^2\left(\prod_{t=1}^{T}\frac{\beta_t}{\beta_T}\right)^{1/T}}{a}\right)$$
$$\le T\frac{c_{\mathcal{F}}^2}{a}\left(\prod_{t=1}^{T}\frac{\beta_t}{\beta_T}\right)^{1/T} \le T\frac{c_{\mathcal{F}}^2}{a\beta_T}\frac{\sum_{t=1}^{T}\beta_t}{T} \le \frac{c_{\mathcal{F}}^2\mathcal{T}}{a}$$

(we use $\ln(1+x) \le x$ and the inequality between the geometric and arithmetic means). Choosing $a = c_{\mathcal{F}}\sqrt{\mathcal{T}}$, we get bound (16) from (15). □

Recall again that $\left(\sum_{t=1}^{T}\beta_t\right)/\beta_T = (1-\alpha^{T-1})/(1-\alpha) \le 1/(1-\alpha)$ in the case of the exponential discounting ($\alpha_t = \alpha \in (0,1)$), and we can take $\mathcal{T} = 1/(1-\alpha)$.

Without discounting ($\alpha_t = 1$), we have $\left(\sum_{t=1}^{T}\beta_t\right)/\beta_T = T$, so we need to know the number of steps in advance. Then, bound (16) matches the bound obtained in [22, the displayed formula after (33)]. If we do not know an upper bound $\mathcal{T}$ in advance, it is still possible to achieve a bound similar to (16) using the Aggregating Algorithm with discounting to merge Learner's strategies from Theorem 4 with different values of parameter a, in the same manner as in [22, Theorem 3]—see [6, Corollary 2] for details.

4.3 Proofs of Theorems 3 and 4

Let us begin with several technical lemmas from linear algebra. For complete proofs of them see [6, Appendix A.1].

Lemma 2. *Let A be a symmetric positive definite matrix of size $n \times n$. Let $\theta, b \in \mathbb{R}^n$, and $c \in \mathbb{R}$. Let $Q(\theta) = \theta' A\theta + b'\theta + c$ and $Q_0 = \min_{\theta\in\mathbb{R}^n} Q(\theta)$. Then*

$$\int_{\mathbb{R}^n} \mathrm{e}^{-Q(\theta)} d\theta = \mathrm{e}^{-Q_0} \frac{\pi^{n/2}}{\sqrt{\det A}}\,.$$

The proof of this lemma can be found in [10, Theorem 15.12.1].

Lemma 3. *Let A be a symmetric positive definite matrix $n \times n$, $b, z \in \mathbb{R}^n$, and*

$$F(A,b,z) = \min_{\theta\in\mathbb{R}^n} (\theta' A\theta + b'\theta + z'\theta) - \min_{\theta\in\mathbb{R}^n} (\theta' A\theta + b'\theta - z'\theta)\,.$$

Then $F(A,b,z) = -b'A^{-1}z$.

Lemma 4. *Let A be a symmetric positive definite matrix of size $n \times n$. Let $\theta, b_1, b_2 \in \mathbb{R}^n$, c_1, c_2 be real numbers, and $Q_1(\theta) = \theta' A\theta + b_1'\theta + c_1$, $Q_2(\theta) = \theta' A\theta + b_2'\theta + c_2$. Then*

$$\frac{\int_{\mathbb{R}^n} \mathrm{e}^{-Q_1(\theta)} d\theta}{\int_{\mathbb{R}^n} \mathrm{e}^{-Q_2(\theta)} d\theta} = \mathrm{e}^{c_2 - c_1 - \frac{1}{4}(b_2+b_1)' A^{-1}(b_2-b_1)}\,.$$

The previous three lemmas were implicitly used in [21] to derive a bound on the cumulative undiscounted square loss of the algorithm competing with linear experts.

Lemma 5. *For any matrix B of size $n \times m$, any matrix C of size $m \times n$, and any real a such that the matrices $aI_m + CB$ and $aI_n + BC$ are nonsingular, it holds that*

$$B(aI_m + CB)^{-1} = (aI_n + BC)^{-1}B\,,$$

where I_n, I_m are the unit matrices of sizes $n \times n$ and $m \times m$, respectively.

Proof. Note that this is equivalent to $(aI_n + BC)B = B(aI_m + CB)$. □

Lemma 6. *For any matrix B of size $n \times m$, any matrix C of size $m \times n$, and any real number a, it holds*

$$\det(aI_n + BC) = \det(aI_m + CB)\,,$$

where I_n, I_m are the unit matrices of sizes $n \times n$ and $m \times m$, respectively.

Proof of Theorem 3. We take the Gaussian initial distribution over the experts with a parameter $a > 0$:

$$P_0(d\theta) = \left(\frac{a\eta}{\pi}\right)^{n/2} \mathrm{e}^{-a\eta\|\theta\|^2} d\theta.$$

and use "Algorithm 1 with infinitely many Experts". Repeating the derivations from the proof of Theorem 1, we obtain the following analogue of (7):

$$\left(\frac{a\eta}{\pi}\right)^{n/2} \int_{\Theta} \mathrm{e}^{\eta\left(\sum_{t=1}^{T} \frac{\beta_t}{\beta_T}(\gamma_t - y_t)^2 - \sum_{t=1}^{T} \frac{\beta_t}{\beta_T}(\theta' x_t - y_t)^2\right)} \mathrm{e}^{-a\eta\|\theta\|^2} d\theta \le 1.$$

The simple equality

$$\sum_{t=1}^{T}\frac{\beta_t}{\beta_T}(\theta' x_t - y_t)^2 + a\|\theta\|^2 = \theta'(aI + X'W_T X)\theta - 2\sum_{t=1}^{T}\frac{\beta_t}{\beta_T}y_t\theta' x_t + \sum_{t=1}^{T}\frac{\beta_t}{\beta_T}y_t^2 \quad (17)$$

shows that the integral can be evaluated with the help of Lemma 2:

$$\left(\frac{a\eta}{\pi}\right)^{n/2}\int_{\Theta} \mathrm{e}^{-\eta\left(\sum_{t=1}^{T}\frac{\beta_t}{\beta_T}(\theta' x_t - y_t)^2 + a\|\theta\|^2\right)}d\theta$$
$$= \left(\frac{a}{\pi}\right)^{n/2}\mathrm{e}^{-\eta\min_\theta\left(\sum_{t=1}^{T}\frac{\beta_t}{\beta_T}(\theta' x_t - y_t)^2 + a\|\theta\|^2\right)}\frac{\pi^{n/2}}{\sqrt{\det(aI + X'W_T X)}}.$$

We take the natural logarithms of both parts of the bound and using the value $\eta = \frac{2}{(Y_2 - Y_1)^2}$ obtain (13). As in the proof of Corollary 1, we note that $\det\left(\frac{X'W_T X}{a} + I\right) \le \left(\frac{Z^2\sum_{t=1}^{T}\beta_t}{a\beta_T} + 1\right)^n$, and (14) follows.

Proof of Theorem 4. We need to prove that the guarantee (15) is satisfied for each T and each $(x_1, y_1, \ldots, x_T, y_T) \in (\mathbf{X}\times\mathbb{R})^T$. Fix T and $(x_1, y_1, \ldots, x_T, y_T)$. Fix an isomorphism between the linear span of $k_{x_1}, \ldots, k_{x_T}$ obtained for the Riesz Representation theorem and $\mathbb{R}^{\tilde{T}}$, where $\tilde{T}$ is the dimension of the linear span of $k_{x_1}, \ldots, k_{x_T}$. Let $\tilde{x}_1, \ldots, \tilde{x}_T \in \mathbb{R}^{\tilde{T}}$ be the images of $k_{x_1}, \ldots, k_{x_T}$ under this isomorphism. We have then $k(\cdot, x_i) = \langle\cdot, \tilde{x}_i\rangle$ for any x_i.

We apply the strategy from Theorem 3 to $\tilde{x}_1, \ldots, \tilde{x}_T$. The predictions of the strategies are the same due to Proposition 1 below. Any expert $\theta \in \mathbb{R}^{\tilde{T}}$ in bound (13) can be represented as $\theta = \sum_{i=1}^{T} c_i\tilde{x}_i = \sum_{i=1}^{T} c_i k(\cdot, x_i)$ for some $c_i \in \mathbb{R}$. Thus the experts' predictions are $\theta'\tilde{x}_t = \sum_{i=1}^{T} c_i k(x_t, x_i)$, and the norm is $\|\theta\|^2 = \sum_{i,j=1}^{T} c_i c_j k(x_i, x_j)$.

Denote by $\tilde{X}$ the $T\times\tilde{T}$ matrix consisting of the rows of the vectors $\tilde{x}_1', \ldots, \tilde{x}_T'$. From Lemma 6 we have

$$\det\left(\frac{\tilde{X}'W_T\tilde{X}}{a} + I\right) = \det\left(\frac{\sqrt{W_T}\tilde{X}\tilde{X}'\sqrt{W_T}}{a} + I\right).$$

Therefore, using $\mathbf{K}_T = \tilde{X}\tilde{X}'$, we obtain the upper bound

$$\sum_{t=1}^{T}\frac{\beta_t}{\beta_T}(\gamma_t - y_t)^2 \le \sum_{t=1}^{T}\frac{\beta_t}{\beta_T}\left(\sum_{i=1}^{T} c_i k(x_t, x_i) - y_t\right)^2$$
$$+ a\sum_{i,j=1}^{T} c_i c_j k(x_i, x_j) + \frac{(Y_2 - Y_1)^2}{4}\ln\det\left(\frac{\sqrt{W_T}\mathbf{K}_T\sqrt{W_T}}{a} + I\right)$$

for any $c_i \in \mathbb{R}$, $i = 1, \ldots, T$. By the Representer theorem (see Theorem 4.2 in [16]) the minimum of $\sum_{t=1}^{T}\frac{\beta_t}{\beta_T}(f(x_t) - y_t)^2 + a\|f\|_{\mathcal{F}}^2$ over all $f \in \mathcal{F}$ is achieved on one of the linear combinations from the bound obtained above. This concludes the proof.

4.4 Regression Algorithms

In this subsection we derive explicit form of the prediction strategies for Learner used in Theorems 3 and 4.

Strategy for Theorem 3. In [21] Vovk suggests the following substitution function satisfying (5) for the square loss:

$$\gamma_T = \frac{Y_2 + Y_1}{2} - \frac{g_T(Y_2) - g_T(Y_1)}{2(Y_2 - Y_1)} .$$

It allows us to calculate g_T with unnormalized weights (and omitting the multiplicative factor):

$$g_T(y) = -\frac{1}{\eta}\left(\ln \int_\Theta \mathrm{e}^{-\eta\left(\theta' A_T \theta - 2\theta'(b_{T-1} + y x_T) + \sum_{t=1}^{T-1} \frac{\beta_t}{\beta_T} y_t^2 + y^2\right)} d\theta\right)$$

for any $y \in \Omega = [Y_1, Y_2]$ (here we use the expansion (17)), where

$$A_T = aI + \sum_{t=1}^{T-1} \frac{\beta_t}{\beta_T} x_t x_t' + x_T x_T' = aI + X' W_T X,$$

and $b_{T-1} = \sum_{t=1}^{T-1} \frac{\beta_t}{\beta_T} y_t x_t$. The direct calculation of g_T is inefficient: it requires numerical integration. Instead, we notice that

$$\begin{aligned}\gamma_T &= \frac{Y_2 + Y_1}{2} - \frac{g_T(Y_2) - g_T(Y_1)}{2(Y_2 - Y_1)} \\ &= \frac{Y_2 + Y_1}{2} - \frac{1}{2(Y_2 - Y_1)\eta} \ln \frac{\int_\Theta \mathrm{e}^{-\eta\left(\theta' A_T \theta - 2\theta'(b_{T-1} + Y_1 x_T) + \sum_{t=1}^{T-1} \frac{\beta_t}{\beta_T} y_t^2 + Y_1^2\right)} d\theta}{\int_\Theta \mathrm{e}^{-\eta\left(\theta' A_T \theta - 2\theta'(b_{T-1} + Y_2 x_T) + \sum_{t=1}^{T-1} \frac{\beta_t}{\beta_T} y_t^2 + Y_2^2\right)} d\theta} \\ &= \frac{Y_2 + Y_1}{2} - \frac{1}{2(Y_2 - Y_1)\eta} \ln \mathrm{e}^{\eta\left(Y_2^2 - Y_1^2 - \left(b_{T-1} + \left(\frac{Y_2+Y_1}{2}\right) x_T\right)' A_T^{-1} \left(\frac{Y_2 - Y_1}{2} x_T\right)\right)} \\ &= \left(b_{T-1} + \left(\frac{Y_2 + Y_1}{2}\right) x_T\right)' A_T^{-1} x_T , \end{aligned} \quad (18)$$

where the third equality follows from Lemma 4.

The strategy which predicts according to (18) requires $O(n^3)$ operations per step. The most time-consuming operation is the inversion of matrix A_T.

Strategy for Theorem 4. We use following notation. Let

$\mathbf{k}_T$ be the last column of the matrix $\mathbf{K}_T, \mathbf{k}_T = \{k(x_i, x_T)\}_{i=1}^T$,
$\mathbf{Y}_T$ be the column vector of the outcomes $\mathbf{Y}_T = (y_1, \ldots, y_T)'$.

When we write $\mathbf{Z} = (\mathbf{V}; \mathbf{Y})$ or $\mathbf{Z} = (\mathbf{V}'; \mathbf{Y}')'$ we mean that the column vector $\mathbf{Z}$ is obtained by concatenating two column vectors $\mathbf{V}, \mathbf{Y}$ vertically or $\mathbf{V}', \mathbf{Y}'$ horizontally.

As it is clear from the proof of Theorem 4, we need to prove that the strategy for this theorem is the same as the strategy for Theorem 3 in the case when the kernel is the scalar product.

Proposition 1. *The predictions* (18) *can be represented as*

$$\gamma_T = \left(\mathbf{Y}_{T-1}; \frac{Y_2+Y_1}{2}\right)' \sqrt{W_T}\left(aI + \sqrt{W_T}\mathbf{K}_T\sqrt{W_T}\right)^{-1}\sqrt{W_T}\mathbf{k}_T \qquad (19)$$

for the scalar product kernel $k(x,y) = \langle x, y\rangle$, the unit $T \times T$ matrix I, and $a > 0$.

Proof. For the scalar product kernel we have we have $\mathbf{K}_T = X'X$ and $\sqrt{W_T}\mathbf{k}_T = \sqrt{W_T}Xx_T$. By Lemma 5 we obtain

$$\left(aI + \sqrt{W_T}XX'\sqrt{W_T}\right)^{-1}\sqrt{W_T}Xx_T = \sqrt{W_T}X\left(aI + X'W_TX\right)^{-1}x_T\,.$$

It is easy to see that

$$\left(\mathbf{Y}_{T-1}; \frac{Y_2+Y_1}{2}\right)' W_TX = \left(\sum_{t=1}^{T-1}\frac{\beta_t}{\beta_T}y_tx_t + \left(\frac{Y_2+Y_1}{2}\right)x_T\right)'$$

and

$$X'W_TX = \sum_{t=1}^{T-1}\frac{\beta_t}{\beta_T}x_tx_t' + x_Tx_T'\,.$$

Thus we obtain (18) from (19). □

Acknowledgements. We are grateful to Yura Kalnishkan and Volodya Vovk for numerous illuminating discussions. This work was supported by EPSRC (grant EP/F002998/1).

References

[1] Beckenbach, E.F., Bellman, R.: Inequalities. Springer, Berlin (1961)
[2] Cesa-Bianchi, N., Lugosi, G.: Prediction, learning, and games. Cambridge University Press, Cambridge (2006)
[3] Chaudhuri, K., Freund, Y., Hsu, D.: A parameter-free hedging algorithm. In: Advances in Neural Information Processing Systems, vol. 22, pp. 297–305 (2009)
[4] Chernov, A., Kalnishkan, Y., Zhdanov, F., Vovk, V.: Supermartingales in prediction with expert advice. Theoretical Computer Science 411, 2647–2669 (2010); See also: arXiv:1003.2218 [cs.LG]
[5] Chernov, A., Vovk, V.: Prediction with advice of unknown number of experts. In: Proc. of 26th Conf. on Uncertainty in Artificial Intelligence, pp. 117–125 (2010); See also: arXiv:1006.0475 [cs.LG]
[6] Chernov, A., Zhdanov, F.: Prediction with expert advice under discounted loss. Technical report, arXiv:1005.1918v1 [cs.LG] (2010)

[7] Freund, Y., Hsu, D.: A new hedging algorithm and its application to inferring latent random variables. Technical report, arXiv:0806.4802v1 [cs.GT] (2008)
[8] Gammerman, A., Kalnishkan, Y., Vovk, V.: On-line prediction with kernels and the complexity approximation principle. In: Proc. of 20th Conf. on Uncertainty in Artificial Intelligence, pp. 170–176 (2004)
[9] Gardner, E.S.: Exponential smoothing: The state of the art – part II. International Journal of Forecasting 22, 637–666 (2006)
[10] Harville, D.A.: Matrix algebra from a statistician's perspective. Springer, Heidelberg (1997)
[11] Haussler, D., Kivinen, J., Warmuth, M.: Sequential prediction of individual sequences under general loss functions. IEEE Transactions on Information Theory 44, 1906–1925 (1998)
[12] Herbster, M., Warmuth, M.K.: Tracking the best expert. Machine Learning 32, 151–178 (1998)
[13] Kalnishkan, Y., Vyugin, M.: The weak aggregating algorithm and weak mixability. Technical report, CLRC-TR-03-01 (2003)
[14] Kalnishkan, Y., Vyugin, M.: The weak aggregating algorithm and weak mixability. Journal of Computer and System Sciences 74(8), 1228–1244 (2008)
[15] Muth, J.F.: Optimal properties of exponentially weighted forecasts. Journal of the American Statistical Association 55, 299–306 (1960)
[16] Schölkopf, B., Smola, A.J.: Learning with kernels: Support Vector Machines, regularization, optimization, and beyond. MIT Press, Cambridge (2002)
[17] Sutton, R., Barto, A.: Reinforcement learning: An introduction. MIT Press, Cambridge (1998)
[18] Vovk, V.: Aggregating strategies. In: Proc. of 3rd COLT, pp. 371–383 (1990)
[19] Vovk, V.: A game of prediction with expert advice. Journal of Computer and System Sciences 56, 153–173 (1998)
[20] Vovk, V.: Derandomizing stochastic prediction strategies. Machine Learning 35, 247–282 (1999)
[21] Vovk, V.: Competitive on-line statistics. Int. Stat. Review 69, 213–248 (2001)
[22] Vovk, V.: On-line regression competitive with reproducing kernel Hilbert spaces. Technical report, arXiv:cs/0511058 [cs.LG] (2005)

A Regularization Approach to Metrical Task Systems

Jacob Abernethy[1,⋆], Peter L. Bartlett[1,⋆⋆], Niv Buchbinder[2], and Isabelle Stanton[1,⋆⋆⋆]

[1] UC Berkeley
{jake,bartlett,isabelle}@eecs.berkeley.edu
[2] Microsoft Research
nivbuchb@microsoft.com

Abstract. We address the problem of constructing randomized online algorithms for the Metrical Task Systems (MTS) problem on a metric δ against an oblivious adversary. Restricting our attention to the class of "work-based" algorithms, we provide a framework for designing algorithms that uses the technique of *regularization*. For the case when δ is a uniform metric, we exhibit two algorithms that arise from this framework, and we prove a bound on the competitive ratio of each. We show that the second of these algorithms is $\ln n + O(\log \log n)$ competitive, which is the current state-of-the art for the uniform MTS problem.

1 Introduction

Consider the problem of driving on a congested multi-lane highway with the goal of getting home as fast as possible. You are always able to estimate the speed of all of the lanes, and must pick some lane in which to drive. At any time you are able to switch lanes, but pay an additional penalty for doing so proportional to the distance from your current lane. How should you pick lanes and when should you switch?

This is a concrete example of the *metrical task system* (MTS) problem, first introduced by Borodin, Linial and Saks [1]. The problem is defined on a space of n states with an associated distance metric function. The input to the problem is a series of cost vectors $\mathbf{c} \in \mathbb{R}^n_+$. A MTS algorithm must choose a state i after seeing each $\mathbf{c}$ and must pay the service cost c_i. In addition, the algorithm pays a cost for switching between states that is their distance in the given metric. An alternative model, and the focus of the present work, is to imagine a *randomized* algorithm that maintains a distribution over the states on each round, and pays the expected switching and servicing cost.

Metrical task systems form a very general framework in which many well-known online problems can be posed. The k-server problem on an n-state metric [2], for example, can be modeled as a metrical task system problem with $\binom{n}{k}$ states[1] Another example is process migration over a compute cluster - in this view each node is a state, the distance metric represents the amount of time it takes for a process to migrate from one node to another and the cost vector represents the current load on the machine.

The randomized MTS problem looks strikingly similar to one much more familiar to the learning community: the "experts" setting [3]. The experts problem also requires

⋆ Supported by a Yahoo! PhD Fellowship.
⋆⋆ Supported by NSF grant 0830410.
⋆⋆⋆ Supported by a National Defense Science and Engineering Graduate Fellowship.

[1] The reduction of making each k-subset a state leads to a bound that is linear in k, which is much greater than the conjectured $O(\log k)$ ratio.

© Springer-Verlag Berlin Heidelberg 2010

choosing a distribution on $[n]$ on each of a sequence of rounds, witnessing a cost vector, and paying the associated expected cost of the selected distribution. The two primary distinctions are that (a) no switching cost is paid in the experts problem and (b) the MTS *comparator*, i.e. the offline strategy against which we compare the online algorithm, is given much more flexibility. In the experts problem, the algorithm is only compared to an offline algorithm that must fix its state throughout the game, whereas the MTS offline comparator may choose the cheapest sequence of states knowing all service cost vectors in advance.

The most common measure of the quality of an MTS algorithm is the *competitive ratio*, which takes the performance of the online algorithm on a worst-case sequence of cost vectors and divides this by the cost of the optimal offline comparator on the same sequence. This is a notable departure from the notion of *regret*, which measures the *difference* between the worst-case online and offline cost, and is a much more common metric for evaluating learning algorithms. This extension is necessary because the complexity of the MTS comparator grows over time.

Prior Work. Borodin, Linial and Saks [1] showed that the lower bound on the competitive ratio of any deterministic algorithm over any metric is $2n - 1$. They also designed an algorithm, the Work-Function algorithm, that achieves exactly this bound. This algorithm was further analyzed by Schafer and Sivadasan using the smoothed analysis techniques of Spielman and Teng to show that the average competitive ratio can be improved to $o(n)$ when the topological features of the metric are taken into account [4].

Results improve dramatically when randomization is allowed. The first result for general metrics was an algorithm that achieved a competitive ratio of $\frac{e}{e-1}n - \frac{1}{e-1}$, [5] by Irani and Seiden. In a breakthrough result, Bartal, Blum, Burch and Tompkins [6] gave the first poly-logarithmic competitive algorithm for all metric spaces. This algorithm uses Bartal's result for probabilistically embedding general metric spaces into hierarchically well-separated trees [7, 8]. Fiat and Mendel [9] improved this result further to the currently best competitive algorithm that is $O(\log^2 n \log\log n)$. Recently, Bansal, Buchbinder and Naor [10] proposed another algorithm for general metrics based on a primal-dual approach that is only $O(\log^3 n)$-competitive, but has an optimal competitive factor with respect to service costs. The best known lower bound on the competitive ratio for general metrics is $\Omega(\log n / \log^2 \log n)$ [11]. This improves upon the previous bound of $\Omega(\sqrt{\log n / \log^2 \log n})$ [12]. A widely believed conjecture is that an $O(\log n)$-competitive algorithm exists for all metric spaces.

Better bounds are known for some special metrics. For example, for the line metric a slightly better result of $O(\log^2 n)$ is known [9]. Another metric for which better results are known is the weighted star metric which has an $O(\log n)$-competitive algorithm [9, 13]. The best understood, and most extensively studied metric space is the uniform metric. For the uniform case, Bartal, Linial and Saks [1] showed a lower bound on the competitive ratio for any algorithm of H_n, the nth harmonic number. They [1] also designed an algorithm, Marking, that has competitive ratio $2H_n$. An alternate algorithm, Odd-Exponent [6], bears some similarity to one of the algorithms in this paper, and has a $4\log n + 1$ competitive ratio on the uniform metric. This upper bound was further improved by the Exponential algorithm [5] to $H_n + O(\sqrt{\log n})$. Recently, the Wedge algorithm [14] was introduced with competitive ratio of $\frac{3}{2}H_n - \frac{1}{2n}$. They claim that this achieves a better competitive ratio when $n < 10^8$. Bansal, Buschbinder and Naor [15, 16] designed an algorithm for the uniform metric that is based on a previous primal-dual approach and has near optimal competitive ratio.

Our Contributions. We make several contributions to the randomized metrical task system problem. In Section 2, we propose a clear and coherent framework for developing and analyzing algorithms for the MTS problem. We appeal to the class of *work-based* algorithms for which the probability distribution is chosen as a function of the *work vector*, to be defined in the Preliminaries. We provide the most comprehensive set of analytical tools for bounding the competitive ratio of work-based algorithms.

In Section 3, we develop an approach to the MTS problem using a *regularization* framework. This provides a generic template for constructing randomized MTS algorithms based on certain parameters of the regularized objective. For the case of the uniform metric, we employ the entropy function as a regularizer and exhibit two novel algorithms. The second of these achieves the current state-of-the-art competitive ratio of $H_n + O(\log\log n)$. We discuss potential methods for constructing general-metric algorithms as well.

1.1 Preliminaries

The set $[n] := \{1, \dots, n\}$ is a *metric space* if there exists a distance metric $\delta : [n] \times [n] \to \mathbb{R}_+$. The primary feature of metrics that we will use is that they satisfy the triangle inequality. Given $\mathbf{p}^1, \mathbf{p}^2 \in \Delta_n$, where Δ_n is the n-simplex, we define the Earth Mover Distance (EMD), or Wasserstein Distance, $\text{dist}_\delta(\mathbf{p}^1, \mathbf{p}^2)$, as the least expensive way to transition between $\mathbf{p}^1$ and $\mathbf{p}^2$. It can be computed by the program

$$\begin{aligned} \min \quad & \textstyle\sum_{i,j\in[n]} \delta(i,j)x_{i,j} \\ \text{subject to} \quad & \mathbf{1}_n^\top [x_{i,j}] = \mathbf{p}^1,\ [x_{i,j}]\mathbf{1}_n = \mathbf{p}^2,\ x_{i,j} \geq 0\ \forall i,j \in [n] \end{aligned}$$

We note that, when working with the uniform metric, the EMD is simply the total variational distance. In addition, in an important special case, we can express the EMD in closed form, as described by the following Lemma.

Lemma 1. *Assume we are given $\mathbf{p}^1, \mathbf{p}^2 \in \Delta_n$ with the property that $\mathbf{p}^1$ dominates $\mathbf{p}^2$ at every coordinate but i, that is $p_j^1 \geq p_j^2$ whenever $j \neq i$. Then*

$$\textit{dist}_\delta(\mathbf{p}^1, \mathbf{p}^2) = \sum_{j\in[n]\setminus\{i\}} (p_j^1 - p_j^2)\delta(i,j)$$

The Randomized Metrical Task Systems Problem. Given n states and a metric δ over $[n]$, a randomized algorithm is given a sequence of *service cost vectors* $\mathbf{c}^1, \mathbf{c}^2, \dots, \mathbf{c}^T \in \mathbb{R}_+^n$ as input and must choose a sequence of distributions $\mathbf{p}^1, \mathbf{p}^2, \dots, \mathbf{p}^T \in \Delta_n$. The *cost* of some algorithm A is the total expected "servicing cost" plus the total "moving" cost, i.e.

$$\text{cost}_A(\mathbf{c}^1, \dots, \mathbf{c}^T) := \sum_{t=1}^T \left(\mathbf{p}^t \cdot \mathbf{c}^t + \text{dist}_\delta(\mathbf{p}^t, \mathbf{p}^{t-1})\right)$$

where $\mathbf{p}^0$ is some default distribution, $\langle 1, 0, \ldots, 0\rangle$ by convention.

An *offline* MTS algorithm may select $\mathbf{p}^t$ with knowledge of the entire sequence of cost vectors $\mathbf{c}^1, \ldots, \mathbf{c}^T$. We refer to the optimal offline algorithm by OPT$(\mathbf{c}^1, \ldots, \mathbf{c}^T)$. In this Section we discuss how OPT is computed easily with a simple dynamic program.

An *online* MTS algorithm can select $\mathbf{p}^t$ with knowledge only of $\mathbf{c}^1, \ldots, \mathbf{c}^t$. Notice that, unlike in the usual "expert setting", we let an online algorithm have access to the cost vector $\mathbf{c}^t$ before the distribution $\mathbf{p}^t$ is chosen and the cost $\mathbf{p}^t \cdot \mathbf{c}^t$ is paid.

We measure the performance of an online algorithm by its *Competitive Ratio* (CR), which is the ratio of the cost of this algorithm relative to the cost of the optimal offline algorithm on a worst-case sequence. More precisely, the CR is the infimal value $C > 0$ for which there is some b such that, for any T and any sequence $\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^T$,

$$\text{cost}_A(\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^T) \quad \leq \quad C \cdot \text{cost}_{\text{OPT}}(\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^T) + b$$

The additive term b, which can depend on the fixed parameters of the problem, is included to deal with potential one-time "startup costs".

The Work Function. We observe that the offline algorithm OPT need not play in a randomized fashion because the optimal distributions $\mathbf{p}^t$ will occur at the corners of the simplex. Hence, computing OPT is not difficult, and can be reduced to a simple dynamic programming problem. The elements of this dynamic program are fundamental to all of the results in this paper, and we now define it precisely. Given a sequence $\mathbf{c}^1, \ldots, \mathbf{c}^T$, we define the *work function vector* $\mathbf{W}^t$ at time t by the following recursive definition:

$$\mathbf{W}^0 := \langle 0, \infty, \ldots, \infty\rangle \qquad W_i^t := \min_{j\in[n]} \left\{ W_j^{t-1} + \delta(i,j) + c_j^t \right\}$$

The work function value W_i^t on cost sequence $\mathbf{c}^1, \ldots, \mathbf{c}^t$ is exactly the smallest total cost incurred by an offline algorithm for which $\mathbf{p}^t = \mathbf{e}_i$, i.e. one which must be at location i at time t. Indeed, if we define $W_{\min}^t := \min_i W_i^t$, then we see that OPT$(\mathbf{c}^1, \ldots, \mathbf{c}^T) = W_{\min}^T$.

If we think of the work vector $\mathbf{W}^t$ as a function from $[n]$ to $\mathbb{R}$, where $\mathbf{W}^t(i) := W_i^t$, then it is easily checked that $\mathbf{W}^t$ is *1-Lipschitz* with respect to the metric δ. That is, for all $i, j \in [n]$, $|W_i^t - W_j^t| \leq \delta(i,j)$. We define a notion of a *supported state* which occurs when this Lipschitz constraint becomes tight.

Definition 1. *Given some work vector $\mathbf{W}^t$ with respect to a metric δ, the state i is* supported *if there exists a $j \neq i$ such that $W_i^t = W_j^t + \delta(i,j)$. In this case, we say that state i is* supported by j.

Intuitively, when a state i becomes supported by j at time t, it has essentially become "useless" for an offline algorithm. In such a case, the total cost of arriving at i after t rounds is no more than the total cost of arriving at j plus the cost $\delta(i,j)$ of switching to i. By a simple application of the triangle inequality, we may conclude that there is an optimal offline algorithm that visits only unsupported states.

Throughout this text, when it is unnecessary, we will drop the superscript t from $\mathbf{W}^t$, W_i^t, $\mathbf{p}^t$ and p_i^t.

2 The Work-Based MTS Framework

We now develop a very general framework, characterized by a set of conditions and assumptions on the algorithm and cost sequences, in which to develop techniques for the randomized MTS problem. The only actual restriction we impose on the algorithm is Condition 1, although we conjecture that this can be made without loss of optimality. The remaining Conditions either follow from the latter, or can be made without loss of generality as we describe.

Condition 1. *The algorithm will be "work-based", that is, we choose* $\mathbf{p}^t = \mathbf{p}(\mathbf{W}^t)$ *for some fixed function* $\mathbf{p}$ *regardless of the sequence of cost vectors that resulted in* $\mathbf{W}$.

This paper focuses entirely on the construction of work-based algorithms, where the algorithm can forget about the sequence of cost vectors $\mathbf{c}^1, \ldots, \mathbf{c}^t$ and simply use $\mathbf{W}^t$ to choose $\mathbf{p}^t$. This algorithmic restriction has been used as early as [1] and appears elsewhere. It has not been shown, to the best of our knowledge, that this restriction is made *without sacrificing optimality*. We conjecture this to be true.

Conjecture 1. There is an optimal randomized MTS algorithm that is work-based. In other words, there is an optimal algorithm such that, after receiving $\mathbf{c}^1, \ldots, \mathbf{c}^t$, the probability $\mathbf{p}^t$ need only depend on the resulting work vector $\mathbf{W}^t$.

Strictly speaking, we need not settle this conjecture to proceed with developing algorithms within this restricted class. However, if it were settled in the affirmative this would suggest that the algorithmic design problem can be safely restricted to this smaller class of algorithms. Indeed, by making this assumption we gain a number of other properties that we list below.

Condition 2. *All cost vectors are "elementary": every* $\mathbf{c}^t$ *has the form* $\alpha \mathbf{e}_i$ *for some* $\alpha > 0$ *and some* i.

It has been shown that a worst-case adversary need only assign cost to a single state on each round. Intuitively this is because, rather than revealing the costs of several states at once to the player, an adversary can spread these costs out over a sequence of rounds at no cost to OPT. This intuition can be more formally justified, and we refer the reader to Irani and Seiden [5] for more details.

Condition 3. *The algorithm will be "reasonable": whenever* $W_i = W_j + \delta(i,j)$ *for some* j *(i.e.* i *is a supported state) then it must be that* $p_i(\mathbf{W}) = 0$.

To reiterate, this condition requires that the probability assigned to state i must vanish whenever i is support within $\mathbf{W}$. This is an unusual condition but it is *required* for any work-based MTS algorithm and it follows from Condition 1. Whenever this property is broken an adversary can induce an unbounded competitive ratio. If $p_i(\mathbf{W}^{t-1}) > 0$ and $W_i^{t-1} = W_j^{t-1} + \delta(i,j)$ for some j, then the adversary can select, say, the cost vector $\mathbf{c}^t = \mathbf{e}_i$ (or any positive scaling of $\mathbf{e}_i$). By construction, the work vector will be unchanged, $\mathbf{W}^t = \mathbf{W}^{t-1}$, and hence the work-based algorithm will not change its distribution, $\mathbf{p}(\mathbf{W}^t) = \mathbf{p}(\mathbf{W}^{t-1})$. However, the algorithm will pay at least $\mathbf{p}^t \cdot \mathbf{c}^t =$

$p_i(\mathbf{W}^t) = p_i(\mathbf{W}^{t-1}) > 0$. The adversary can repeat this process, leaving the work vector (and hence the cost of OPT) unchanged, leading to an unbounded cost for the algorithm. For more discussion, see Bartal et al [6].

Condition 4. *The cost vectors will be "reasonable" as well: Given a current work vector* $\mathbf{W}$*, if a cost* $\mathbf{c}^{t+1} = \alpha \mathbf{e}_i$ *is received then* $\alpha \leq W_j^t - W_i^t + \delta(i, j)$ *for all* j*.*

This assumption can be made, without loss of generality, when Condition 3 is satisfied. More specifically, we can convert a sequence of elementary cost vectors which does not satisfy this property to a sequence which does, without any change to the online algorithm or offline cost OPT. Consider an elementary cost vector $\mathbf{c}^t = \alpha \mathbf{e}_i$ for some state i and some $\alpha > W_j^{t-1-1} - W_i^t + \delta(i, j)$ for some j, and imagine converting this to $\mathbf{c}^t = \alpha' \mathbf{e}_i$ defined by "rounding down", $\alpha' := W_j^{t-1} - W_i^{t-1} + \delta(i, j)$. The resulting work vector $\mathbf{W}^{t\prime}$ after $\mathbf{c}^{t\prime}$ is identical to $\mathbf{W}^t$ after $\mathbf{c}^t$, and the algorithm's distribution $\mathbf{p}(\mathbf{W}^t)$ is also identical. Furthermore, as a result of Condition 3, the servicing cost is 0, i.e. $\mathbf{p}(\mathbf{W}^t) \cdot \mathbf{c}^{t\prime} = \mathbf{p}(\mathbf{W}^t) \cdot \mathbf{c}^t = 0$. Hence, we may assume that α is already rounded down and thus $\mathbf{c}^t$ is reasonable. The observation was first shown by Fiat and Mendel [9]. With this condition we arrive at a useful Lemma:

Lemma 2. *Under the assumption that the sequence of cost vectors* $\mathbf{c}^1, \ldots, \mathbf{c}^t$ *is reasonable, the work vector is precisely* $\mathbf{W}^t = \mathbf{c}^1 + \ldots + \mathbf{c}^t$*.*

Condition 5. *The algorithm will be "conservative": Given a work vector* $\mathbf{W}$*, whenever a cost* $\mathbf{c} = \alpha \mathbf{e}_i$ *is received, then for each* $j \neq i$ *we have* $p_j(\mathbf{W}) \leq p_j(\mathbf{W} + \alpha \mathbf{e}_i)$ *– that is, the probabilities at the locations not receiving cost can not decrease.*

We include this condition to make the analysis easier, in particular because we may now use the more convenient form of the Earth Mover Distance described in Lemma 1. In general it is not strictly necessary to require an MTS algorithm to be conservative. On the other hand, it is easy to show that it is a beneficial assumption, and it is used throughout the literature.

2.1 Relationship to the Experts Setting

Before proceeding, let us show why the proposed framework brings us closer to a well-understood problem, the "experts" setting [3]. Here, the algorithm must choose a probability distributions $\mathbf{p}^t \in \Delta_n$ on each round t, and an adversary then chooses a loss vector $\boldsymbol{l}^t \in [0, 1]^n$. Let $\mathbf{L}^t = \sum_{s=1}^t \boldsymbol{l}^s$. Then the algorithm's goal is to minimize $\sum_{t=1}^T \mathbf{p}^t \cdot \boldsymbol{l}^t$ relative to the loss of the "best expert", i.e. $\min_i L_i^t$.

Within our MTS framework, the story is quite similar. The algorithm and adversary choose $\mathbf{p}^t$ and $\mathbf{c}^t$ on each round. By Lemma 2, $\mathbf{W}^T = \sum_{t=1}^T \mathbf{c}^t$, and the algorithm's goal is to pay as little as possible relative to $\min_i W_i^t$.

These problems have a strong resemblance, yet there are several critical differences:

- The MTS algorithm has *one-step lookahead*: it can select $\mathbf{p}^t$ with knowledge of $\mathbf{c}^t$
- An additional penalty $\text{dist}_\delta(\mathbf{p}^{t-1}, \mathbf{p}^t)$ for moving is added to the objective for MTS
- The algorithm must be "reasonable", requiring that the probability p_i^t must vanish under certain conditions of $\mathbf{W}^t$.

While the first point would appear quite advantageous for MTS, the benefit is spoiled by the latter two. In the expert setting we can ensure that the average cost of the algorithm approaches the comparator $\min_i L_i^t$ using an algorithm like Hedge, whereas in the MTS setting a lower bound shows that this ratio is at least $\Omega(\log n/\log^2 \log n)$ for the work function comparator [11]. At a high level, this is because charging the algorithm for adjusting its distribution *and* requiring that the probability vanishes on certain states causes the algorithm to pay a huge amount in transportation.

In Section 3, we borrow some tools from the experts setting such as *entropy regularization* and potential functions. Algorithms from the experts setting have been used on the MTS problem before, most notably by [17]. Their approach is quite different from the one we take. They imagine competing against a "switching expert" and modify known results developed by [18]. Their approach, while quite interesting, is not a work-based algorithm and does not achieve an optimal bound.

2.2 Bounding Costs Using Potential Functions

We turn our attention to bounding the cost of a work-based MTS algorithm $\mathbf{p}$ on a worst-case sequence of costs. First, we make a simple observation about work-based algorithms that adhere to our framework. Given a work vector $\mathbf{W}$, consider the cost to the algorithm when vector $\mathbf{c} = \epsilon \mathbf{e}_i$ is received and the work vector becomes $\mathbf{W}^1 = \mathbf{W} + \epsilon \mathbf{e}_i$. The probability distribution transitions to $\mathbf{p}(\mathbf{W}^1)$, and the service cost is $\mathbf{p}(\mathbf{W}^1) \cdot \mathbf{c} = \epsilon p_i(\mathbf{W}^1)$. By the conservative assumption, we compute the switching cost by appealing to Lemma 1. Hence, the total cost is

$$\mathbf{p}(\mathbf{W}^1) \cdot \mathbf{c} + \mathrm{dist}_\delta(\mathbf{p}(\mathbf{W}), \mathbf{p}(\mathbf{W}^1)) = \epsilon p_i(\mathbf{W}^1) + \sum_{j \in [n] \setminus \{i\}} (p_j(\mathbf{W}^1) - p_j(\mathbf{W}))\delta(i,j). \tag{1}$$

In the present work, we will consider designing algorithms with $\mathbf{p}(\mathbf{W})$ which are both continuous and differentiable. With this in mind, we can take (1) a step further and let $\epsilon \to 0$ to get the instantaneous increase in cost to the algorithm as we add cost to state i. Using continuity, we see that $\mathbf{W}^1 \to \mathbf{W}$ as $\epsilon \to 0$, which gives that the instantaneous cost at $\mathbf{W}$ in the direction of $\mathbf{e}_i$ as $p_i(\mathbf{W}) + \sum_{j \in [n] \setminus \{i\}} \frac{\partial p_j(\mathbf{W})}{\partial W_i} \delta(i,j)$.

Ultimately, we need to bound the total cost of the algorithm on any sequence. The typical way to achieve this is with a potential function that maintains an upper bound on the worst case sequence of cost vectors that results in the current $\mathbf{W}$. There is a natural "best" potential function $\Phi^*_{\mathbf{p}}(w)$ for a given algorithm $\mathbf{p}$, which we now construct.

For any measurable function $I : \mathbb{R}_+ \to [n]$, we can define a continuous path through the space of work vectors by $\mathbf{W}_I(s) = \int_0^s \mathbf{e}_{I(\alpha)} d\alpha$. This is exactly the continuous version of Lemma 2. The function $I(s)$ specifies which coordinate of $\mathbf{W}_I(s)$ is increasing at time s. Let $\rho(\mathbf{W})$ be the set of all functions I which induce paths starting at $\mathbf{0}$ that lead to $\mathbf{W}$. We now construct a potential function,

$$\Phi^*_{\mathbf{p}}(\mathbf{W}) = \sup_{I \in \rho(\mathbf{W})} \int_0^{T : \mathbf{W}_I(T) = \mathbf{W}} \left(p_{I(s)}(\mathbf{W}_I(s)) + \sum_{j \neq I(s)} \frac{\partial p_j(\mathbf{W}_I(s))}{\partial \mathbf{W}_I(s)} \delta(i,j) \right) ds.$$

This potential function measures precisely the worst case cost of arriving at a work vector $\mathbf{W}$.

Lemma 3. *For any sequence of reasonable elementary vectors* $\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^T$ *with* $\mathbf{W} = \sum_t \mathbf{c}^t$*, the cost to algorithm* $\mathbf{p}$ *is no more than* $\Phi^*_{\mathbf{p}}(\mathbf{W})$*. Furthermore,* $\Phi^*_{\mathbf{p}}(\mathbf{W})$ *is the supremal cost over all possible cost sequences* $\{\mathbf{c}^t\}$ *that lead to* $\mathbf{W}$*.*

Proof. This fact is straightforward and we sketch the proof. For any $\mathbf{W}$ and any cost vector $\mathbf{c} = \epsilon \mathbf{e}_i$ (enough by Condition 2), the cost to the algorithm is expressed in Equation (1). Now, instead of applying the cost all at once, consider applying it in a continuous fashion, then the cost is

$$\int_0^\epsilon \left(p_i(\mathbf{W} + s\mathbf{e}_i) + \sum_{j \neq i} \frac{\partial p_j(\mathbf{W} + s\mathbf{e}_i)}{\partial W_i} \delta(i,j) \right) ds.$$

By Condition 5, $p_i(\mathbf{W} + s\mathbf{e}_i) \geq p_i(\mathbf{W} + \epsilon\mathbf{e}_i)$ for any $\mathbf{s} \in [0, \epsilon]$ and hence this expression is an upper bound on Equation (1). In addition, for any sequence of $\mathbf{c}^t$'s, we can construct an associated smooth path I that leads to $\mathbf{W}$ by integrating the cost smoothly for each $\mathbf{c}^t$ in the same fashion. But $\Phi^*_{\mathbf{p}}(\mathbf{W})$ was defined as the supremum cost over such paths. Thus, both the lower and upper bound follow.

Once we have $\Phi^*_{\mathbf{p}}$, the competitive ratio of $\mathbf{p}$ has the following characterization.

Lemma 4. *The competitive ratio of algorithm* $\mathbf{p}$ *is the infimal value* C *such that* $\Phi^*_{\mathbf{p}}(\mathbf{W}) - CW_{\min}$ *is bounded away from* $+\infty$ *for all* $\mathbf{W}$*.*

Certain work-based algorithms, which we will call *shift-invariant* algorithms, satisfy $\mathbf{p}(\mathbf{W}) = \mathbf{p}(\mathbf{W} + c\mathbf{1})$ for any $\mathbf{W}$ and any c.

Lemma 5. *The competitive ratio of a shift-invariant algorithm is* $\mathbf{1} \cdot \nabla \Phi^*_{\mathbf{p}}(\mathbf{W})$ *for any* $\mathbf{W}$*.*

Finding the optimal $\Phi^*_{\mathbf{p}}$ for the algorithm $\mathbf{p}$ may be difficult. To prove an upper bound on the competitive ratio, however, we need only construct a *valid* Φ. Precisely, define $\Phi(\mathbf{W})$ to be valid with respect to the algorithm $\mathbf{p}$ if, for all $\mathbf{W}$ and all i, we have

$$\frac{\partial \Phi(\mathbf{W})}{\partial W_i} \geq p_i(\mathbf{W}) + \sum_{j \neq i} \frac{\partial p_j(\mathbf{W})}{\partial W_i} \delta(i,j)$$

Lemma 6. *Given any* $\mathbf{p}$ *and any valid potential* Φ*,* C *is an upper bound on the competitive ratio if* $\Phi(\mathbf{W}) - CW_{\min}$ *is bounded away from* $+\infty$*.*

In the following Section, we show how to design algorithms and construct potentials for the case of uniform metrics using regularization techniques.

3 Work-Based Algorithms via Regularization

We begin this Section by providing a general tool for the construction of work-based MTS algorithms. We present a *regularization* approach, common in the adversarial online learning community, which we modify to ensure the required conditions for the

MTS setting. We present two algorithms for the uniform metric from this framework, with associated potential functions, and we prove a bound on the competitive ratio of each. We finish by discussing how to extend this approach to general metric spaces.

3.1 Regularization and Achieving Reasonableness

We now turn our attention to the problem of designing competitive work-based algorithms for the case when δ is the uniform metric. The uniform metric is such that all states are the same distance from each other–that is, we assume without loss of generality $\delta(i,j) = 1$ whenever $i \neq j$ and 0 otherwise.

To obtain a competitive work-based algorithm, we need to find a function $\mathbf{p}$ and construct an associated potential function Φ with the following properties:

- (Conservativeness) We require that $\frac{\partial p_j(\mathbf{W})}{\partial W_i} \geq 0$ for any $\mathbf{W}$ and $\forall j \neq i$
- (Reasonableness) The probability $p_i(\mathbf{W})$ must vanish whenever i is a supported state for $\mathbf{W}$, i.e. when $W_i = W_j + \delta(i,j)$ for some j
- (Valid Potential) $\forall \mathbf{W}, i$, the potential Φ must satisfy $\frac{\partial \Phi(\mathbf{W})}{\partial W_i} \geq p_i(\mathbf{W}) - \frac{\partial p_i(\mathbf{W})}{\partial W_i}$

Notice that the term $-\frac{\partial p_i(\mathbf{W})}{\partial W_i}$ has replaced $\sum_{j\neq i} \frac{\partial p_j(\mathbf{W})}{\partial W_i}\delta(i,j)$ in the last expression. These two quantities are equal when δ is the uniform metric, precisely because for any j we have $\sum_i \frac{\partial p_i(\mathbf{W})}{\partial W_j} = 0$ since $\sum_i p_i(\mathbf{W}) = 1$.

In order to obtain an algorithm with a low competitive ratio, we must construct a slowly-changing $\mathbf{p}(\mathbf{W})$ and a valid potential $\Phi(\mathbf{W})$ that controls the motion of $\mathbf{p}(\mathbf{W})$ as $\mathbf{W}$ varies in each direction. In other words, we would like to enforce a level of stability in $\mathbf{p}(\mathbf{W})$. Stability is a central concept within both the batch-learning and the adversarial online-learning literature. The most common and thoroughly analyzed approach is to employ *regularization*. To describe this approach, let us return our attention to the experts setting discussed in Section 2.1. Recall that, at time t, a distribution $\mathbf{p}^t \in \Delta_n$ is to be chosen with knowledge of $\boldsymbol{l}^1, \ldots, \boldsymbol{l}^{t-1}$. This can be achieved by solving the following regularized objective,

$$\mathbf{p}^t = \underset{\mathbf{p}\in\Delta_n}{\operatorname{argmin}} \left(R(\mathbf{p}) + \lambda \sum_{s=1}^{t-1} \mathbf{p} \cdot \boldsymbol{l}^s\right) \tag{2}$$

where generally the “regularizer” R is selected as some smooth convex function and λ is a learning parameter. How to select the correct regularizer is a major area of research, but for the experts setting the most common is the negative of the *entropy function*, $R(\mathbf{p}) := \sum_{i\in[n]} p_i \log p_i$. This choice leads to the well-known exponential weights:

$$p_i^t = \frac{\exp\left(-\lambda \sum_{s=1}^{t-1} l_i^s\right)}{\sum_j \exp\left(-\lambda \sum_{s=1}^{t-1} l_j^s\right)} \tag{3}$$

Regularization in online learning appears in the literature at least as early as [19] and [20], and more modern analyses can be found in [21] and [22].

In this paper, we use the regularization framework to produce an algorithm $\mathbf{p}(\mathbf{W})$. It is tempting to suggest solving the equivalent objective of Equation (2), where we treat $\mathbf{W}$ as the cumulative costs; this leads to setting

$$\mathbf{p}(\mathbf{W}) = \underset{\mathbf{p}}{\operatorname{argmin}} \, (R(\mathbf{p}) + \lambda \mathbf{p} \cdot \mathbf{W}). \tag{4}$$

This approach can indeed guarantee stability with the correct R, and it's easy to check that the objective induces a conservative algorithm. Unfortunately, it does *not* enforce the reasonableness property that we require. (It has been shown that an unreasonable work-based algorithm must admit an unbounded competitive ratio [17].).

The question we are thus left with is, how can we adjust the objective to maintain stability and ensure reasonableness? Recall, when δ is the uniform metric, the reasonableness property requires that $p_i(\mathbf{W}) \to 0$ whenever $1 + W_j - W_i$ approaches 0 for any j, or equivalently when $1 + W_{\min} - W_i \to 0$. To guarantee this behavior, we propose replacing the term $\mathbf{p} \cdot \mathbf{W}$ in Equation (4) with $\sum_i p_i f_i(\mathbf{W}, \lambda)$ where the function $f_i(\mathbf{W}, \lambda)$ will be a *Lipschitz penalty*: for any metric δ on $[n]$ and any $1-$Lipschitz vector $\mathbf{W}$ with respect to δ, we say that $f_i(\mathbf{W}, \lambda)$ is a Lipschitz penalty function if $f_i(\mathbf{W}, \lambda) \to \infty$ as $\min_j \big(W_j - W_i + \delta(i,j)\big) \to 0$. λ is a learning parameter that may be tuned. Hence, we propose the following method to find $\mathbf{p}(\mathbf{W})$:

$$\mathbf{p}(\mathbf{W}) = \underset{\mathbf{p}}{\operatorname{argmin}} \, (R(\mathbf{p}) + \sum_i p_i f_i(\mathbf{W}, \lambda)). \tag{5}$$

For both algorithms in the following Section, we employ the entropy function for our regularizer $R(\mathbf{p})$.

3.2 Two Resulting Algorithms for the Uniform Metric

We will consider the following two Lipschitz penalty functions, and analyze the resulting algorithms:

$$\begin{aligned} &\text{(Alg 1)} \quad f_i(\mathbf{W}, \lambda) = -\lambda \log(1 + W_{\min} - W_i) \\ &\text{(Alg 2)} \quad f_i(\mathbf{W}, \lambda) = -\log(e^{\lambda(1+W_{\min}-W_i)} - 1) \end{aligned}$$

The analysis of both algorithms proceeds by solving the regularization function to find p_i as a function of $\mathbf{W}$ and then using the potential function technique of Section 2.2 to bound the switching and servicing costs regardless of which state receives cost. For both, we separate the analysis into two cases: when increasing W_i causes $W_{\min} = \min_j W_j$ to increase, and when increasing W_i does not affect $W_{\min}$.

Theorem 1. *Choosing $R(\mathbf{p}) := \sum_{i\in[n]} p_i \log p_i$, and with Lipschitz penalty $f_i(\mathbf{W}, \lambda) = -\lambda \log(1 + W_{\min} - W_i)$, when $\lambda = \log n$, we achieve an algorithm with competitive ratio no more than $e \log n + 1$ for the uniform metric.*

Proof. We can solve (5) explicitly when $R(\cdot)$ is the negative entropy function. By computing the Lagrangian, we arrive at

$$p_i = \frac{(1 + W_{\min} - W_i)^\lambda}{\sum_{j=1}^n (1 + W_{\min} - W_j)^\lambda}$$

We will show that each of the components of the cost of the algorithm is bounded by a multiple of the following potential function:

$$\Phi(\mathbf{W}) = cW_{\min} - \log \sum_{i=1}^{n} (1 + W_{\min} - W_i)^\lambda$$

The parameters will be set so that $c = e(\log n - 1) + 1$ and $\lambda = \log n$. We will show that these have been tuned optimally.

As discussed in the beginning of this section, we must show that $p_i - \frac{\partial p_i}{\partial W_i} \leq \frac{\partial \Phi}{\partial W_i}$ for all i. We will vary from that slightly and show that when $i \neq \min$, $p_i - \frac{\partial p_i}{\partial W_i} \leq (1 + \frac{1}{\lambda}) \frac{\partial \Phi}{\partial W_i}$ and if $i = \min$ then $p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} \leq \frac{\partial \Phi}{\partial W_{\min}}$. Combining these facts, the competitive ratio will be upper bounded by $(1 + \frac{1}{\lambda})c$.

First, we will show that if $i \neq \min$, $p_i - \frac{\partial p_i}{\partial W_i} \leq (1 + \frac{1}{\lambda}) \frac{\partial \Phi}{\partial W_i}$.

$$\begin{aligned}
p_i - \frac{\partial p_i}{\partial W_i} &= \frac{(1 + W_{\min} - W_i)^\lambda}{\sum_j (1 + W_{\min} - W_j)^\lambda} + \frac{\lambda (W_{\min} - W_i + 1)^{\lambda - 1}}{(\sum_j (1 + W_{\min} - W_j)^\lambda)^2} \\
&\leq \frac{(1 + W_{\min} - W_i)^{\lambda - 1} (\lambda + 1 + W_{\min} - W_i)}{\sum_j (1 + W_{\min} - W_j)^\lambda} \\
&\leq \frac{(\lambda + 1)(W_{\min} - W_i + 1)^{\lambda - 1}}{\sum_j (1 + W_{\min} - W_j)^\lambda} = \frac{\lambda + 1}{\lambda} \frac{\partial \Phi}{\partial W_i}
\end{aligned}$$

Next, we consider $p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} \leq \frac{\partial \Phi}{\partial W_{\min}}$. Notice that $p_{\min} = \frac{1}{Z}$ where $Z = \sum_j (1 + W_{\min} - W_j)^\lambda$. We have

$$p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} = \frac{1}{Z} + \frac{1}{Z^2} \frac{\partial Z}{\partial W_{\min}} \leq 1 + \frac{1}{Z^2} \frac{\partial Z}{\partial W_{\min}}$$

In addition, we see that $\frac{\partial \Phi}{\partial W_{\min}} = c - \frac{1}{Z} \frac{\partial Z}{\partial W_{\min}}$. In order to show that $p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} \leq \frac{\partial \Phi}{\partial W_{\min}}$, using the above two statements it is equivalent to show that

$$\frac{1}{Z} \frac{\partial Z}{\partial W_{\min}} + \frac{1}{Z^2} \frac{\partial Z}{\partial W_{\min}} \leq c - 1$$

We now show this fact. First, let $\alpha_j := 1 + W_{\min} - W_j$. Now we need to maximize

$$\left(1 + \frac{1}{1 + \sum_{j \neq \min} \alpha_j^\lambda}\right) \frac{\lambda \sum_{j \neq \min} \alpha_j^{\lambda - 1}}{1 + \sum_{j \neq \min} \alpha_j^\lambda}$$

This is maximized when $\alpha_j = (\frac{\lambda - 1}{n - 1})^{1/\lambda}$ and attains a max value of $\frac{\lambda + 1}{\lambda} (\lambda - 1)(n - 1)^{1/\lambda} (\lambda - 1)^{-1/\lambda}$. This can be seen by first noting that it is maximized when all α_j are some value α and then taking the derivative with respect to α and setting it equal to 0.

We note that as $\lambda \to \infty$, $(\lambda-1)^{-1/\lambda} \to 1$, as does $\frac{\lambda+1}{\lambda}$. Thus, we only concern ourselves with the limit of $(n-1)^{1/\lambda}$. Let this quantity be L. By L'Hopital's rule:

$$\lim_{n\to\infty} \log L = \lim_{n\to\infty} \frac{\log(n-1)}{\lambda} = \lim_{n\to\infty} \frac{\frac{1}{n-1}}{\frac{d\lambda}{dn}}$$

If we let $\lambda = \log n$ then we have $\frac{1}{(n-1)}/\frac{1}{n} \to 1$. Thus, $L = e$ and $p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} \le \frac{\partial \Phi}{\partial W_{\min}}$ if $c - 1 > (\lambda-1)(n-1)^{1/\lambda} = e(\log n - 1)$. Therefore, $c = e(\log n - 1) + 1$.

Finally, we note that we have both requirements, $p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} \le \frac{\partial \Phi'}{\partial W_{\min}}$ and $p_i - \frac{\partial p_i}{\partial W_i} \le \frac{\partial \Phi'}{\partial W_i}$ for $\Phi' = (1+\frac{1}{\lambda})\Phi$. Therefore, the total cost of this algorithm is bounded by $(1+\frac{1}{\lambda})c\text{OPT} = (1+\frac{1}{\log n})(e(\log n - 1)+1)\text{OPT} \le (e\log n + e + 1)\text{OPT}$.

The previous algorithm demonstrates our analysis technique for a very simple and natural Lipschitz-penalty function. However, it has a somewhat unsatisfying competitive ratio of $e \log n$. Even the very simple Marking algorithm has a better competitive ratio of $2H_n$. Next, we will show that a different Lipschitz penalty function, $f_i(\mathbf{W},\lambda) = \log(\exp(\lambda(1+W_{\min}-W_i)) - 1)$, produces an algorithm that achieves the current best competitive ratio for the uniform MTS problem.

Theorem 2. *If we employ the Lipschitz penalty* $f_i(\mathbf{W},\lambda) = -\log(\exp(\lambda(1+W_{\min} - W_i)) - 1)$ *with* $\lambda = \log n + 2\log\log n$, *with* $\mathbb{R}(\cdot)$ *the negative entropy as before, then we achieve a competitive ratio of no more than* $\log n + O(\log\log n)$ *for the uniform metric.*

Proof. Solving the regularization problem when $f_i(\mathbf{W},\lambda) = \log(\exp(\lambda(1+W_{\min} - W_i)) - 1)$ results in

$$p_i = \frac{e^{\lambda(1+W_{\min}-W_i)} - 1}{\sum_j e^{\lambda(W_{\min}-W_j+1)} - 1}$$

We will show that the following is a valid potential function:

$$\Phi(\mathbf{W}) = cW_{\min} - \frac{1+\lambda}{\lambda}\log\sum_{i=1}^{n}(e^{\lambda(1+W_{\min}-W_i)} - 1).$$

This analysis requires tuning the parameter λ, which we will do at the end.

In the same vein as the previous proof, we will show that $p_i - \frac{\partial p_i}{\partial W_i} \le \frac{\partial \Phi}{\partial W_i}$. We will break this up into two steps, one where $i \ne \min$ and when $i = \min$.

Let us consider the case when $i \ne \min$. Let $Z = \sum_j (e^{\lambda(1+W_{\min}-W_j)} - 1)$, the normalization term of the above distribution. For any $i \ne \min$, we see that

$$\begin{aligned} p_i - \frac{\partial p_i}{\partial W_i} &= p_i + \frac{\lambda e^{\lambda(1+W_{\min}-W_i)}}{Z} + \frac{1}{Z^2}\frac{\partial Z}{\partial W_i}(e^{\lambda(1+W_{\min}-W_i)} - 1) + \frac{\lambda}{Z} - \frac{\lambda}{Z} \\ &= p_i + \frac{\lambda(e^{\lambda(1+W_{\min}-W_i)} - 1)}{Z} + \frac{\lambda}{Z} + p_i\frac{1}{Z}\frac{\partial Z}{\partial W_i} \\ &= (1+\lambda+\frac{1}{Z}\frac{\partial Z}{\partial W_i})p_i + \frac{\lambda}{Z} \quad \le \quad (1+\lambda)p_i + \frac{\lambda}{Z} \end{aligned}$$

Notice that the final inequality follows since $\frac{\partial Z}{\partial W_i} \leq 0$.

Then, we consider $\frac{\partial \Phi}{\partial W_i}$.

$$\begin{aligned}\frac{\partial \Phi}{\partial W_i} = \frac{\lambda+1}{\lambda}\frac{1}{Z}\frac{\partial Z}{\partial W_i} &= \frac{\lambda+1}{\lambda}\frac{1}{Z}(\lambda e^{\lambda(W_{\min}-W_i+1)}) \\ &= \frac{1+\lambda}{Z}e^{\lambda(W_{\min}-W_i+1)} + \frac{1+\lambda}{Z} - \frac{1+\lambda}{Z} = (1+\lambda)(p_i + 1/Z)\end{aligned}$$

$p_i - \frac{\partial p_i}{\partial W_i} \leq \frac{\partial \Phi}{\partial W_i}$ follows immediately.

Now let $i = \min$. Notice that $p_{\min} = \frac{e^{\lambda}-1}{Z}$, so we have

$$p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} = p_{\min} + (e^{\lambda}-1)\frac{1}{Z^2}\frac{\partial Z}{\partial W_{\min}} = p_{\min}\left(1 + \frac{1}{Z}\frac{\partial Z}{\partial W_{\min}}\right)$$

Furthermore,

$$\frac{\partial \Phi}{\partial W_{\min}} = c - \frac{1+\lambda}{\lambda}\frac{1}{Z}\frac{\partial Z}{\partial W_{\min}}$$

We compute

$$\begin{aligned}\frac{1}{Z}\frac{\partial Z}{\partial W_{\min}} = \frac{\lambda}{Z}\sum_{j\neq\min} e^{\lambda(W_{\min}-W_j+1)} &= \frac{\lambda}{Z}\sum_{j\neq\min}(e^{\lambda(W_{\min}-W_j+1)} - 1) + \lambda\frac{n-1}{Z} \\ &= \lambda\left(1 - p_{\min} + \frac{n-1}{Z}\right)\end{aligned}$$

Putting the last three statements together, we can restate $p_{\min} - \frac{\partial p_{\min}}{\partial W_i} \leq \frac{\partial \Phi}{\partial W_{\min}}$ as

$$\begin{aligned}p_{\min}\left(1 + \lambda\left(1 - p_{\min} + \frac{n-1}{Z}\right)\right) &\leq c - (1+\lambda)\left(1 - p_{\min} + \frac{n-1}{Z}\right) \\ \frac{n-1}{Z}(1+\lambda+\lambda p_{\min}) + 1 + \lambda(1-p_{\min}^2) &\leq c\end{aligned}$$

Noting that $Z \geq e^{\lambda}-1$ and $\lambda p_{\min} \leq \lambda$, it is equivalent to show that $\frac{(2\lambda+1)n}{e^{\lambda}-1} + 1 + \lambda \leq c$. Setting $\lambda = \log n + 2\log\log n$ gives that the first term is $o(1)$, and we can then set $c = \lambda + 1 + o(1)$. Thus the competitive ratio of this algorithm is $\log n + O(\log\log n)$, the best achieved thus far.

3.3 Extending to General Metrics

It has become relatively well-established in the online learning literature that the negative entropy function is an ideal regularizer when we want to control the L1-stability of our hypothesis, which is the relevant distance function for distributions over a uniform metric space. On the other hand, notice that the algorithmic template we propose in ([5]) does not rely on the uniform metric, and can be posed in general. Constructing algorithms for arbitrary metrics has been the biggest challenge for the MTS problem, and we still have a gap in the minimax competitive ratio between $\Omega(\log n)$ and $O(\log^2 n)$. Unfortunately, extending our results to general metrics does not lead to an algorithm with an $O(\log n)$-competitive ratio.

For other metrics, it is clear that entropy is not at all the correct regularizer. Instead, what is needed is a regularization function that controls the stability of $\mathbf{p}$ with respect to the norm induced by the Earth Mover Distance $\mathrm{dist}_\delta(\cdot, \cdot)$. It would be of particular interest if such a function existed and could be constructed.

Conjecture 2. For any metric δ on $[n]$, there is some regularization function $R(\cdot)$ such that the algorithm resulting from Equation (5) is $O(\log n)$-competitive.

As an example, in the case of the *weighted star* metric, which is slightly more general than the uniform metric, we conjecture that the weighted entropy [23] is the correct choice of regularizer. We note that the resulting algorithm is similar in flavor to the MTS algorithm of Bansal et al [13], which is known to achieve an $O(\log n)$ algorithm for this metric.

4 Conclusions and Open Problems

We have introduced a framework for developing and analyzing algorithms for the metrical task system problem. This framework presupposes that an optimal algorithm that is work-based exists, and we conjecture that this is this the case. Given this framework we are able to use the popular entropy regularization approach to develop state-of-the-art algorithms. We believe this system gives good insight into how to develop algorithms for the general metric case.

Our work leaves open several important questions. The most obvious are the answers to our conjectures - is it true that assuming that the algorithm will be work vector based does not preclude optimality? All of the current algorithms for general metrics rely on embedding the metric into a hierarchical search tree and then using MTS algorithms for this metric space and none are known to be based on the work vector.

There is also an open question with regards to the regularization approach. What is the correct regularization function for general distance metrics? We believe that an algorithm for the general metric with even a poly$\log n$ bound on the competitive ratio that is worse than the current results achieved by metric embedding would be interesting due to its potential relative simplicity.

References

[1] Borodin, A., Linial, N., Saks, M.: An optimal on-line algorithm for metrical task system. JACM: Journal of the ACM 39(4), 745–763 (1992)

[2] Manasse, M., McGeoch, L., Sleator, D.: Competitive algorithms for server problems. J. Algorithms 11(2), 208–230 (1990)

[3] Freund, S.: A decision-theoretic generalization of on-line learning and an application to boosting. JCSS: Journal of Computer and System Sciences 55 (1997)

[4] Schafer, G., Sivadasan, N.: Topology matters: Smoothed competitiveness of metrical task systems. TCS: Theoretical Computer Science 341 (2005)

[5] Irani, S., Seiden, S.: Randomized algorithms for metrical task systems. Theoretical Computer Science 194 (1998)

[6] Bartal, Y., Blum, A., Burch, C., Tomkins, A.: A polylog(n)-competitive algorithm for metrical task systems. In: Symposium on Theory Of Computing (STOC), pp. 711–719 (1997)

[7] Bartal, Y.: On approximating arbitrary metrics by tree metrics. In: Symposium Theory Of Computing (STOC), pp. 161–168 (1998)

[8] Fakcharoenphol, J., Rao, S., Talwar, K.: A tight bound on approximating arbitrary metrics by tree metrics. In: Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of computing (STOC), pp. 448–455 (2003)

[9] Fiat, A., Mendel, M.: Better algorithms for unfair metrical task systems and applications. SIAM Journal on Computing 32 (2003)

[10] Bansal, N., Buchbinder, N., Naor, S.: Metrical task systems and the k-server problem on hsts (2009) (manuscript)

[11] Bartal, Y., Bollobás, B., Mendel, M.: A ramsey-type theorem for metric spaces and its applications for metrical task systems and related problems. In: IEEE Symposium on Foundations of Computer Science (FOCS), pp. 396–405 (2001)

[12] Blum, A., Karloff, H., Rabani, Y., Saks, M.: A decomposition theorem and lower bounds for randomized server problems. SIAM Journal on Computing 30, 1624–1661 (2000)

[13] Bansal, N., Buchbinder, N., Naor, J.: A primal-dual randomized algorithm for weighted paging. In: IEEE Symposium on Foundations of Computer Science, FOCS (2007)

[14] Bein, W., Larmore, L., Noga, J.: Uniform metrical task systems with a limited number of states. IPL: Information Processing Letters 104 (2007)

[15] Bansal, N., Buchbinder, N., Naor, S.: Towards the randomized k-server conjecture: A primal-dual approach. In: ACM-SIAM Symposium on Discrete Algorithms, SODA (2010)

[16] Buchbinder, N., Naor, S.: The design of competitive online algorithms via a primal-dual approach. Foundations and Trends in Theoretical Computer Science 3(2-3), 93–263 (2009)

[17] Blum, A., Burch, C.: On-line learning and the metrical task system problem. Machine Learning 39(1), 35–58 (2000)

[18] Herbster, M., Warmuth, M.K.: Tracking the best expert. Machine Learning 32, 151 (1998)

[19] Kivinen, J., Warmuth, M.: Exponentiated gradient versus gradient descent for linear predictors. Information and Computation (1997)

[20] Gordon, G.: Regret bounds for prediction problems. In: Proceedings of the Twelfth Annual Conference on Computational Learning Theory, pp. 29–40. ACM, New York (1999)

[21] Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, New York (2006)

[22] Rakhlin, A.: Lecture Notes on Online Learning DRAFT (2009)

[23] Guiau, S.: Weighted entropy. Reports on Mathematical Physics 2(3), 165–179 (1971)

Solutions to Open Questions for Non-U-Shaped Learning with Memory Limitations

John Case[1] and Timo Kötzing[2,⋆]

[1] Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716-2586, USA
case@cis.udel.edu

[2] Max-Planck-Institut für Informatik, Campus E 1 4, 66123 Saarbrücken, Germany
koetzing@mpi-inf.mpg.de

Abstract. A U-shape occurs when a learner first learns, then unlearns, and, finally, relearns, some target concept. Within the framework of Inductive Inference, previous results have shown, for example, that U-shapes are unnecessary for explanatory learning, but are necessary for behaviorally correct learning.

This paper solves the following two problems regarding non-U-shaped learning posed in the prior literature.

First, it is shown that there are classes learnable with three memory states that are not learnable non-U-shapedly with any finite number of memory states. This result is surprising, as for learning with one or two memory states, U-shapes are known to be unnecessary.

Secondly, it is shown that there is a class learnable memorylessly with a single feedback query such that this class is not learnable non-U-shapedly memorylessly with any finite number of feedback queries. This result is complemented by the result that any class of *infinite* languages learnable memorylessly with finitely many feedback queries is so learnable without U-shapes – in fact, all classes of infinite languages learnable with *complete* memory are learnable memorylessly with finitely many feedback queries and without U-shapes. On the other hand, we show that there is a class of infinite languages learnable memorylessly with a single feedback query, which is *not* learnable with*out* U-shapes by any particular bounded number of feedback queries.

Keywords: Inductive Inference.

1 Introduction and Motivation

In Section 1.1 we explain U-shaped learning and in Section 1.2 memory-limited learning. In Section 1.3 we summarize our *main* results of the present paper with pointers to later sections where they are treated in more detail.

⋆ Timo Kötzing was supported by the Deutsche Forschungsgemeinschaft (DFG) under grant no. NE 1182/5-1.

© Springer-Verlag Berlin Heidelberg 2010

1.1 U-Shaped Learning

U-shaped learning occurs when a learner first learns a correct behavior, then abandons that correct behavior and finally returns to it once again. This pattern of learning has been observed by cognitive and developmental psychologists in a variety of child development phenomena, such as language learning [SS82], understanding of temperature [SS82], weight conservation [SS82], object permanence [SS82] and face recognition [Car82]. The case of language acquisition is paradigmatic. In the case of the past tense of English verbs, it has been observed that children learn correct syntactic forms (call/called, go/went), then undergo a period of overregularization in which they attach regular verb endings such as 'ed' to the present tense forms even in the case of irregular verbs (break/breaked, speak/speaked) and finally reach a final phase in which they correctly handle both regular and irregular verbs. This example of U-shaped learning behavior has figured prominently in cognitive science [MPU+92, TA02].

While the prior cognitive science literature on U-shaped learning was typically concerned with modeling *how* humans achieve U-shaped behavior, [BCM+08, CCJS07a] are motivated by the question of *why* humans exhibit this seemingly inefficient behavior. Is it a mere harmless evolutionary inefficiency or is it *necessary* for full human learning power? A technically answerable version of this question is: are there some formal learning tasks for which U-shaped behavior is logically necessary? We first need to describe some formal criteria of successful learning.

An algorithmic learning function h is, in effect, fed an infinite sequence consisting of the elements of a (formal) language L in arbitrary order with possibly some pause symbols $\#$ in between elements. During this process the machine outputs a corresponding sequence $p(0), p(1), \ldots$ of hypotheses (grammars) which may generate the language L to be learned. A fundamental criterion of successful learning of a language is called *explanatory learning* (TxtEx-*learning*) and was introduced by Gold [Gol67]. Explanatory learning requires that the learner's output conjectures stabilize in the limit to a *single* conjecture (grammar/program, description/explanation) that generates the input language. Formally, *behaviorally correct learning* [CL82, OW82] requires, for successful learning, only convergence in the limit to possibly infinitely many syntactically distinct but correct conjectures. Another interesting class of criteria features *vacillatory learning* [Cas99, JORS99]. This paradigm involves learning criteria which allow the learner to vacillate in the limit between *at most* some finite number of syntactically distinct but correct conjectures. For each criterion that we consider above (and below), a *non U-shaped learner* is naturally modeled as a learner that never *semantically* returns to a previously abandoned correct conjecture on languages it learns according to that criterion.

[BCM+08] showed that every TxtEx-learnable class of languages is TxtEx-learnable by a non U-shaped learner, that is, for TxtEx-learnability, U-shaped learning is *not* necessary. Furthermore, based on a proof in [FJO94], [BCM+08] noted that, by contrast, for behaviorally correct learning [CL82, OW82], U-shaped learning *is* necessary for full learning power. In [CCJS07a] it is shown that, for non-trivial vacillatory learning, U-shaped learning is again necessary (for full learning power). Thus, in many contexts, seemingly inefficient U-shaped learning can actually increase one's learning power.

1.2 Memory-Limited Learning

It is clear that human learning involves memory limitations. In the present paper (as in [CCJS07b]) we consider the necessity of U-shaped learning in some formally *memory-limited* versions of language learning. In the prior literature many types of memory-limited learning have been studied [LZ96, WC80, Wie76, OSW86, FJO94, CJLZ99, CCJS07b]. Herein we study the types from [CCJS07b] about which that paper has some results, and answer the open questions from that paper about those types.

1.3 Brief Summary of Main Results

The paper [CCJS07b] introduces *Bounded Memory State (BMS) learning.* Associated learners do *not* have access to any previously seen data. Instead, after each datum presented, the learner may choose one out of a bounded number of memory states, and, when presented with another datum, will be passed this memory state along with the new datum. Thus, each output of new conjecture and new memory state may depend only on the new datum and the just previous memory state. Intuitively, such a learner can be pictured as a finite state machine, where the transitions depend on each new datum.[1]

In [CCJS07b], the authors show that Bounded Memory States learning with up to two memory states does not require U-shapes.

As an open problem (Problem 40) they ask whether or not U-shapes are similarly unnecessary for higher numbers of memory states. Surprisingly, Theorem 3 says that there is a class learnable with *three* memory states which is not learnable for *any* number of memory states and with*out* U-shapes. Hence, in all but the bottom two cases for number of memory states available, *U-shapes are necessary* for increased learning power.

Also in [CCJS07b], *Memoryless Feedback (MLF) learning* is introduced. This is similar to BMS learning in that a learner does not have access to any strictly previously seen data. Instead, for a given natural number n, the learner may *query*, in parallel, for up to n different datapoints, whether those datapoints have

[1] For such a learner with a number of memory states equal $c \geq 1$, intuitively, the learner can store any one out of c different *values* in its long term memory [FKS95, KS95]. For example, when $c = 2^k$, the memory is equivalent to the learner having k bits of memory.

For the criteria studied for example in [Wie76, CJLZ99, JK09, JLMZ10], learning functions also have access to their just prior output conjecture (if any), *but*, for the criteria studied herein, learning functions have no such access.

been seen previously. No query may depend on the outcome of another query, and all queries will be individually answered truthfully, so that the learner knows for each queried datum whether it has been seen before.

In [CCJS07b], the authors show that, for each choice of parameter $n > 0$, U-shapes are necessary for the full learning power of MLF learning. As an open problem (Problem 39), they ask, for any given parameter $m > 0$, whether there is a parameter $n > m$ such that MLF learning with a (possibly high) parameter of n allows for non-U-shaped learning of all classes of languages that are MLF learnable with parameter m. We answer this question negatively, and show a much stronger result: Theorem 5 below says that there is a class of languages learnable memorylessly with a *single* feedback query which is not non-U-shapedly MLF learnable with *arbitrarily many sequential recall queries* For this, the learner may even continue asking queries, *dependent on the outcome of previous queries*, and not be limited to any finite number.

We complement this latter result by showing that any class of *infinite* languages learnable memorylessly with finitely many feedback queries is so learnable without U-shapes. Even stronger, Theorem 6 states that each TxtEx-learnable class of infinite languages is learnable memorylessly with arbitrarily many feedback queries and without U-shapes.

For this theorem, it is essential that the number of feedback queries is not bounded: Theorem 7 states that there is a class of infinite languages learnable memorylessly with a single feedback query, which is *not* learnable with*out* U-shapes by any *particular bounded* number of feedback queries.

We conclude our analysis of MLF learning by showing that it is *essential* that a query can be used to find out whether the current datum has been seen before (see Theorem 9).

Many proofs involve subtle infinitary program self-reference arguments employing the Operator Recursion Theorem (ORT) and variants from [Cas74, Cas94]. Because of space limitations, some proofs are omitted herein.

2 Mathematical Preliminaries

Unintroduced computability-theoretic notions follow [Rog67].

$\mathbb{N}$ denotes the set of natural numbers, $\{0, 1, 2, \ldots\}$.

The symbols $\subseteq, \subset, \supseteq, \supset$ respectively denote the subset, proper subset, superset and proper superset relation between sets. The symbol $\setminus$ denotes set-difference.

The quantifier $\forall^\infty x$ means "for all but finitely many x", the quantifier $\exists^\infty x$ means "for infinitely many x". For any set A, $\mathrm{card}(A)$ denotes its cardinality.

With $\mathfrak{P}$ and $\mathfrak{R}$ we denote, respectively, the set of all partial and of all total functions $\mathbb{N} \to \mathbb{N}$. With dom and range we denote, respectively, domain and range of a given function.

We sometimes denote a partial function f of $n > 0$ arguments $x_1, \ldots, x_n$ in lambda notation (as in Lisp) as $\lambda x_1, \ldots, x_n \centerdot f(x_1, \ldots, x_n)$. For example, with $c \in \mathbb{N}$, $\lambda x \centerdot c$ is the constantly c function of one argument.

For any predicate P, we let $\mu x \centerdot P(x)$ denote the least x such that $P(x)$.

We fix any computable 1-1 and onto pairing function $\langle\cdot,\cdot\rangle : \mathbb{N}\times\mathbb{N}\rightarrow\mathbb{N}$.[2] Whenever we consider tuples of natural numbers as input to $f \in \mathfrak{P}$, it is understood that the general coding function $\langle\cdot,\cdot\rangle$ is used to (left-associatively) code the tuples into a single natural number.

If $f \in \mathfrak{P}$ is not defined for some argument x, then we denote this fact by $f(x)\uparrow$, and we say that f on x *diverges*; the opposite is denoted by $f(x)\downarrow$, and we say that f on x *converges*. If f on x converges to p, then we denote this fact by $f(x)\downarrow = p$.

We say that $f \in \mathfrak{P}$ *converges to* p iff $\forall^\infty x : f(x)\downarrow = p$; we write $f \rightarrow p$ to denote this.[3]

A partial function $\psi \in \mathfrak{P}$ is *partial computable* iff there is a deterministic, multi-tape Turing machine which, on input x, returns $\psi(x)$ if $\psi(x)\downarrow$, and loops infinitely if $\psi(x)\uparrow$. $\mathcal{P}$ and $\mathcal{R}$ denote, respectively, the set of all partial computable and the set of all computable functions $\mathbb{N}\rightarrow\mathbb{N}$. The functions in $\mathcal{R}$ are called *computable functions*.

We let φ be any fixed acceptable programming system for the partial computable functions $\mathbb{N}\rightarrow\mathbb{N}$ with associated complexity measure Φ. Further, we let φ_p denote the partial computable function computed by the φ-program with code number p, and we let Φ_p denote the partial computable *complexity* function of the φ-program with code number p.

A set $L \subseteq \mathbb{N}$ is *computably enumerable (ce)* iff it is the domain of a computable function. Let $\mathcal{E}$ denote the set of all ce sets. We let W be the mapping such that $\forall e : W(e) = \mathrm{dom}(\varphi_e)$. For each e, we write W_e instead of $W(e)$. W is, then, a mapping from $\mathbb{N}$ *onto* $\mathcal{E}$. We say that e is an index, or program, (in W) for W_e.

Whenever we consider sequences or finite sets as input to functions, we assume these objects to be appropriately coded as natural numbers.

2.1 Learning in the Limit

In this section we formally define several criteria for learning in the limit.

A *learner* is a partial computable function.

A *language* is a ce set $L \subseteq \mathbb{N}$. Any total function $T : \mathbb{N}\rightarrow\mathbb{N}\cup\{\#\}$ is called a *text*. For any given language L, a *text for* L is a text T such that $\mathrm{content}(T) = L$.

A *sequence generating operator* is an operator β taking as arguments a learner h and a text T and that outputs a function p. We call p the *learning sequence* of h given T.

Let β be a sequence generating operator and h a learner. We proceed by giving definitions for β-learning, and, additionally, for non-U-shaped variants. The non-U-shaped variants will require a learner never to change *semantically* any correct conjecture (on a path to successful learning).

We say that h *β-learns a language* L iff, for all texts T for L and $p = \beta(h, T)$, there is i_0 such that, for all $i \geq i_0$, $p(i) \in \{?, p(i_0)\}$ and $W_{p(i_0)} = L$. We denote the class of all languages β-learned by h with $\beta(h)$. By β we denote the set of all classes of languages β-learnable by learners.

[2] For a linear-time example, see [RC94, Section 2.3].

[3] $f(x)$ converges should not be confused with f converges *to*.

As the first example sequence generating operator we define **TxtEx** thus. $\forall h, T, i : \mathbf{TxtEx}(h,T)(i) = h(T[i])$. Then, for example, we see that h **TxtEx**-learns L iff, for all texts T for L, for some i_0, the sequence of learner h's outputs, $h(T[i_{i_0}]), h(T[i_{i_0}+1]), h(T[i_{i_0}+2]), \ldots$, begins with some correct W-program $p(i_0)$ for L and, after that the elements of the sequence are either $p(i_0)$ or ?.

We say that h $\mathbf{NU}\beta$*-learns a language* L iff h β-learns L and, for all texts T for L and $p = \beta(h,T)$, for all i_0 such that $W_{p(i_0)} = L$ and all $i \geq i_0$, $W_{p(i)} = L$. We denote the class of all languages $\mathbf{NU}\beta$-learned by h with $\mathbf{NU}\beta(h)$. With $\mathbf{NU}\beta$ we denote the set of all classes of languages $\mathbf{NU}\beta$-learnable by learners.

For Bounded Memory States learning with $n \geq 1$ states, learners are functions of the kind $\langle h, f\rangle$, i.e., learners with two outputs: the first for a new conjecture, the second for a new memory state. Given such a learner $\langle h, f\rangle$ and a text T, we define recursively the $\mathbf{BMS}_n$ learning sequence p and the sequence q of states[4] of $\langle h, f\rangle$ given T thus.

$$p(0) = ?; \tag{1}$$
$$q(0) = 0; \tag{2}$$
$$\forall i : p(i+1) = h(T(i), q(i)); \tag{3}$$
$$\forall i : q(i+1) = \min(n-1, f(T(i), q(i))). \tag{4}$$

The sequence generating operator $\mathbf{BMS}_n$ is defined accordingly.

Memoryless Feedback learning, as given in [CCJS07b], is a learning criterion where the learner works in two stages. In the first stage, the learner is presented a datum and uses it to compute a finite *set*. In the second stage, the learner computes a new conjecture, given the same datum and, additionally, for each element x of the finite set computed in the first stage, an indicator of whether x occurred previously in the current text.

Intuitively, each element x in the set resulting from the first stage represents the question "have I seen datum x previously?".

Variants of memoryless feedback learning, where the size of each such set is bounded by a fixed parameter $n \in \mathbb{N}$, are also studied in [CCJS07b]. Herein, we additionally study a variant where a learner is allowed to make queries *sequentially*, which we call memoryless *recall* learning (MLR).

We will not give formal details for modeling the associated sequence generating operators. Instead, we employ an informal query function `rcl` described below. For each $n \in \mathbb{N} \cup \{*\}$, let $\mathbf{MLF}_n$ be the sequence generating operator associated with allowing n *parallel* queries (feedback learning). Further, for all $m \in \mathbb{N} \cup \{*\}$, we let $\mathbf{MLR}_m$ be the sequence generating operator associated with allowing m *sequential* queries (recall learning).

As indicated, for specifying learners with feedback or recall queries, we introduce the use of `rcl` as follows.[5]

[4] Without loss of generality, the set of states is $\{0, \ldots, n-1\}$.

[5] The use of `rcl` is "syntactic sugar".

For $a, b, c \in \mathcal{P}$ and $d \in \mathbb{N}$ we will frequently make statements such as the following.

$$\forall x : \varphi_d(x) = \begin{cases} a(x), & \text{if } \texttt{rcl}(c(x)); \\ b(x), & \text{otherwise.} \end{cases} \tag{5}$$

Intuitively, this means that φ_d on input (new datum) x will first recall $c(x)$, and then, if $c(x)$ was seen previously, output $a(x)$, otherwise $b(x)$. Furthermore, for a finite set D, we use $\texttt{rcl}(D)$ to denote the set $\{x \in D \mid \texttt{rcl}(x)\}$.

Starred Learners

For a learner h, possibly learning with restricted access to past data, we write $h^*(\sigma)$ for what the current output of h is after being fed the sequence σ of data items.

3 Bounded Memory States Learning (BMS)

Definition 1. Let $f \in \mathcal{P}$. For this definition, we let $f^* \in \mathcal{P}$ be such that $f^*(\emptyset) = 0$ and $\forall \sigma, x : f^*(\sigma \diamond x) = f(x, f^*(\sigma))$.[6]

For all $g \in \mathcal{R}$, let Y_g be such that

$$Y_g = \{j \mid (\forall k \leq j+1 : f^*(g[k])\downarrow) \ \wedge \ (\forall k \leq j) f^*(g[k]) \neq f^*(g[j+1])\}. \tag{6}$$

Intuitively, Y_g is the set of all j such that f, when presented the text g, changes into a previously not visited state after seeing element $g(j)$.

Lemma 2. Let $f \in \mathcal{P}$. Let f^* be as in Definition 1 above. For $g \in \mathcal{P}$, we will below refer to the following statement.

$$\forall k : f^*(g[k])\downarrow. \tag{7}$$

(i) There is an effective operator $\Theta : \mathcal{P} \to \mathcal{P}$ [7] such that, for all $g \in \mathcal{R}$, if (7), then $\Theta(g)$ is total and decides Y_g.
(ii) If range(f) is finite, then, for all $g \in \mathcal{R}$, Y_g is finite.
(iii) For all $g \in \mathcal{R}$, if (7), then

$$\forall \tau \subset g \; \exists \sigma \in \mathbb{S}\mathrm{eq}(\{g(j) \mid j \in Y_g\}) : f^*(\tau) = f^*(\sigma). \tag{8}$$

Proof. Obviously, Θ as follows satisfies (i).

$$\forall x, j : \Theta(\varphi_x)(j) = \begin{cases} \uparrow, & \text{if } (\exists k \leq j+1 : f^*(\varphi_x[k])\uparrow); \\ 1, & \text{else if } (\forall k \leq j) f^*(\varphi_x[k]) \neq f^*(\varphi_x[j+1]); \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

(ii) is easy to see.

[6] Note that $f^*(\emptyset) = 0$ is the initial state of any given **BMS**-learner.

[7] I.e. there exists a computable $f \in \mathcal{R}$ such that, for all φ-programs q, $\Theta(\varphi_q) = \varphi_{f(q)}$ [Rog67].

(iii) can be seen by $\subseteq$-induction on τ as follows. Let $\tau \subset g$ be such that, for all $\hat{\tau} \subset \tau$, $\exists \sigma \in \mathbb{S}\mathrm{eq}(\{g(j) \mid j \in Y_g\}) : f^*(\hat{\tau}) = f^*(\sigma)$. Let $\tau_0 \subseteq \tau$ be the $\subseteq$-minimum such that $f^*(\tau_0) = f^*(\tau)$. The conclusion is trivial if $\tau_0 = \emptyset$. Else, $\#\mathrm{elets}(\tau_0) - 1 \in Y_s$. Let $\sigma \in \mathbb{S}\mathrm{eq}(\{s(j) \mid j \in Y_s\})$ such that $f^*(\tau_0^-) = f^*(\sigma)$. Therefore,

$$f^*(\tau) = f^*(\tau_0) = f(\mathrm{last}(\tau_0), f^*(\tau_0^-)) = f(\mathrm{last}(\tau_0), f^*(\sigma)) = f^*(\sigma \diamond \mathrm{last}(\tau_0)). \tag{10}$$

Hence, $(\sigma \diamond \mathrm{last}(\tau_0)) \in \mathbb{S}\mathrm{eq}(\{g(j) \mid j \in Y_g\})$ is the desired sequence witnessing (iii) for τ. □

Contrasting $\mathbf{BMS}_1 = \mathbf{NUBMS}_1$ and $\mathbf{BMS}_2 = \mathbf{NUBMS}_2$ from [CCJS07b], we have the following theorem, solving an open problem, Problem 40, from [CCJS07b].

Theorem 3. We have

$$\mathbf{BMS}_3 \setminus \bigcup_{n>0} \mathbf{NUBMS}_n \neq \emptyset.$$

Proof. Let $M \in \mathcal{P}$ be such that

$$\forall v, x : M(x, v) = \begin{cases} \langle ?, v\rangle, & \text{if } x = \#; \\ \varphi_x(v), & \text{otherwise.} \end{cases} \tag{11}$$

Let $\mathcal{L} = \mathbf{BMS}_3(M)$. Let $n > 0$. Suppose, by way of contradiction, $\mathcal{L} \in \mathbf{NUBMS}_n$, as witnessed by $\langle h, f\rangle$ (h returns the new conjecture, f the new state). Suppose, without loss of generality, range(f) is finite. Let f^* be as in Definition 1 above. Let h^* be such that $h^*(\emptyset) = ?$ and $\forall \sigma, x : h^*(\sigma \diamond x) = h(x, f^*(\sigma))$. Let, for all $s \in \mathcal{R}$, Y_s be as in Definition 1 above, and let Θ be as shown existent in Lemma 2 (i).

Intuitively, $h^*(\sigma)$ is the hypothesis of the learner after seeing σ as input.

We define ce sets P, Q in uniform dependence of $r, s \in \mathcal{P}$ (we abbreviate, for all i, $s_i = \lambda j \centerdot s(i, j)$) such that $\forall a, b, \sigma, \tau$:

$$Q(b, \tau) \Leftrightarrow \exists \xi \in \mathbb{S}\mathrm{eq}(\mathrm{range}(\tau)) : \emptyset \neq W_{h^*(\tau \diamond r(b) \diamond \xi)} \cap (\mathrm{range}(s_b) \setminus \mathrm{range}(\tau)); \tag{12}$$

$$P(a, b, \sigma, \tau) \Leftrightarrow \begin{cases} a \neq b & \wedge \\ \sigma \in \mathbb{S}\mathrm{eq}(\{s_a(j) \mid \Theta(s_a)(j) = 1\}) & \wedge \\ \tau \in \mathbb{S}\mathrm{eq}(\mathrm{range}(s_b)) & \wedge \\ f^*(\sigma) = f^*(\tau) & \wedge \\ f^*(\sigma \diamond r(a)) = f^*(\tau \diamond r(b)) & \wedge \\ Q(b, \tau). \end{cases} \tag{13}$$

Fix a ce-index for P. By 1-1 ORT, there are 1-1 $e, r, s, t, y, z \in \mathcal{R}$ with pairwise disjoint ranges and $p \in \mathbb{N}$ such that a number of restrictions are satisfied. The first group of restrictions is given by the following four equations. $\forall x, i$:

$$\varphi_p(x) = \mu\langle a, b, \sigma, \tau, d\rangle \,\centerdot\, [P(a, b, \sigma, \tau) \text{ in } \leq d \text{ steps}]; \tag{14}$$

$$W_{y(i)} = \text{range}(s_i) \ \cup \ \{r(i)\}; \tag{15}$$

$$W_{z(i)} = \text{range}(s_i); \tag{16}$$

$$W_{e(i)} = \{r(i)\} \ \cup \ \text{range}(t) \ \cup \ \begin{cases} \emptyset, & \text{if } \varphi_p(0)\uparrow; \\ \text{content}(\pi_3(\varphi_p(0))), & \text{else if } i = \pi_1(\varphi_p(0)); \\ \text{content}(\pi_4(\varphi_p(0))), & \text{otherwise.} \end{cases} \tag{17}$$

The second group of restrictions in our application of ORT is indicated by a labeled graph using vertices $\{0, 1, 2\}$. For all elements $x \in \text{range}(r) \cup \text{range}(s) \cup \text{range}(t)$ and $\ell \in \mathbb{N}$, an edge from vertex v to vertex w labeled $\frac{x}{\ell}$ (we use this kind of label for readability; $\frac{k}{\ell}$ is not to be confused with a fraction) in the graph just below adds the restriction $\varphi_x(v) = \langle \ell, w\rangle$ as part of our application of ORT.

$$\begin{tikzcd}
{} \arrow[r] & 0 \arrow[loop above, "\forall i, j : \frac{s_i(j)}{z(i)}"] \arrow[r, "\forall i : \frac{r(i)}{e(i)}"] & 1 \arrow[r, "(\forall i, j \mid \Theta(s_i)(j) = 0)\ \frac{s_i(j)}{y(i)}"] & 2 \arrow[loop above, "\forall i : \frac{t(i)}{e(\pi_2(\varphi_p(0)))}"]
\end{tikzcd}$$

The third and last group of restrictions is as follows.

$$(\forall i, j \mid \Theta(s_i)(j)\uparrow)\varphi_{s_i(j)}(1)\uparrow \tag{18}$$

and, for all $x \in \text{range}(r) \cup \text{range}(s) \cup \text{range}(t)$ and vertices v such that $\varphi_x(v)$ was not previously specified, we have the restriction $\varphi_x(v) = \langle ?, v\rangle$.

It is easy to verify that these three groups are not contradictory and embody a valid application of ORT.

The above graph now allows us to easily determine whether certain interesting subsets of $\text{range}(r) \cup \text{range}(s) \cup \text{range}(t)$ are in $\mathcal{L}$. For example,

$$\forall i \in \mathbb{N} : \text{range}(s_i) = W_{z(i)} \in \mathcal{L}. \tag{19}$$

Statement (19) can be derived from with the help of the graph, as the graph shows that, for all i, M, on any element from $\text{range}(s_i)$, stays in state 0 and outputs $z(i)$ as hypothesis.

From (19) we get, for all i and $\sigma \in \mathbb{S}\text{eq}(\text{range}(s_i))$, $f^*(\sigma)\downarrow$; in particular, for all i, we have (7) with s_i in the place of g. Therefore, for all i, using Lemma 2 (i), $\Theta(s_i)$ is total and (18) is vacuous. By Lemma 2 (ii), for each i, Y_{s_i} is finite, and, thus, the above graph easily shows

$$\forall i \in \mathbb{N} : \text{range}(s_i) \cup \{r(i)\} = W_{y(i)} \in \mathcal{L}. \tag{20}$$

Hence,

$$\forall i \in \mathbb{N}, \forall \rho \in \mathbb{S}\text{eq}(W_{y(i)}) : \langle h^*, f^*\rangle(\rho)\downarrow. \tag{21}$$

Claim 1. $\forall b \in \mathbb{N}\, \exists \tau \subset s_b : Q(b, \tau)$.

Proof of Claim 1. Let $b \in \mathbb{N}$. By the Pigeonhole Principle, as range(f) is finite, there is v such that $\exists^\infty k : f^*(s_b[k]) = v$. By (19), there is j_0 such that

$$W_{h^*(s_b[j_0])} = W_{z(b)}; \text{ and } \forall j \geq j_0 : h^*(s_b[j]) \in \{?, h^*(s_b[j_0])\}. \tag{22}$$

Let $j_1 > j_0$ be such that $f^*(s_b[j_1]) = v$. By (20), there is $k \in \mathbb{N}$ such that

$$W_{h^*(s_b[j_1] \diamond r(b) \diamond s_b[k])} = W_{y(b)}. \tag{23}$$

Let $j_2 > k$ be such that $f^*(s_b[j_2]) = v$. Then

$$W_{h^*(s_b[j_2] \diamond r(b) \diamond s_b[k])} = W_{y(b)}. \tag{24}$$

Hence, $\exists \tau \subset s_b : Q(b, \tau)$ as witnessed by $s_b[j_2]$ for τ and $s_b[k]$ for ξ.

$\square$ (for Claim 1)

Claim 2. $\varphi_p(0)\!\downarrow$.

Proof of Claim 2. For the proof of this claim only, for each $b \in \mathbb{N}$, we fix τ_b as shown existent by Claim 1.

There are only finitely many pairs of states (elements of range(f)), while there are infinitely many $b \in \mathbb{N}$. Hence, by the Pigeonhole Principle, there are $a, b \in \mathbb{N}$ such that $a \neq b$, $f^*(\tau_a) = f^*(\tau_b)$ and $f^*(\tau_a \diamond r(a)) = f^*(\tau_b \diamond r(b))$. To show the claim, we use Lemma 2 (iii) with s_a in place of g to replace τ_a by σ such that $\sigma \in \mathbb{S}\mathrm{eq}(\{s_a(j) \mid j \in Y_{s_a}\})$ and $f^*(\tau_a) = f^*(\sigma)$. Thus,

$$f^*(\sigma \diamond r(a)) = f^*(\tau_a \diamond r(a)) = f^*(\tau_b \diamond r(b)) \tag{25}$$

and

$$f^*(\sigma) = f^*(\tau_a) = f^*(\tau_b). \tag{26}$$

Now we see that $P(a, b, \sigma, \tau_b)$, as

$$\begin{array}{rl} a \neq b & \text{by choice of } a, b; \\ \sigma \in \mathbb{S}\mathrm{eq}(\{s_a(j) \mid \Theta(s_a)(j) = 1\}) & \text{by choice of } \sigma \text{ and } \Theta(s_a) \text{ decides } Y_{s_a}; \\ \tau_b \in \mathbb{S}\mathrm{eq}(\mathrm{range}(s_b)) & \text{as } \tau_b \subset s_b; \\ f^*(\sigma) = f^*(\tau_b) & \text{as (26)}; \\ f^*(\sigma \diamond r(a)) = f^*(\tau_b \diamond r(b)) & \text{because of (25)}; \\ Q(b, \tau) & \text{by choice of } \tau_b. \end{array}$$

$\square$ (for Claim 2)

Let

$$\langle a, b, \sigma, \tau, d \rangle = \varphi_p(0), \tag{27}$$

and let ξ be as stated existent by $Q(b, \tau)$. We have

$$W_{e(a)} = \{r(a)\} \cup \mathrm{range}(t) \cup \mathrm{content}(\sigma), \text{ and} \tag{28}$$
$$W_{e(b)} = \{r(b)\} \cup \mathrm{range}(t) \cup \mathrm{content}(\tau). \tag{29}$$

It is easy to see (from the graph above) that $W_{e(a)}, W_{e(b)} \in \mathcal{L}$.

Let

$$\begin{aligned} \rho_a &= \sigma \diamond r(a), && (30)\\ \rho_b &= \tau \diamond r(b), && (31)\\ T_a &= \rho_a \diamond t, && (32)\\ T_b &= \rho_b \diamond t, \text{ and} && (33)\\ T'_b &= \rho_b \diamond \xi \diamond t. && (34) \end{aligned}$$

Then T_a is a text for $W_{e(a)}$ and T_b, T'_b are texts for $W_{e(b)}$. As $f^*(\rho_a) = f^*(\rho_b)$, and, as h has to identify $W_{e(a)}$ from $\rho_a \diamond t = T_a$ and $W_{e(b)}$ from $\rho_b \diamond t = T_b$, we have, for all k, $h^*(\rho_a \diamond t[k]) =?$ and $h^*(\rho_b \diamond t[k]) =?$. Thus, there is ℓ such that $W_{h^*(\rho_b[\ell])} = W_{e(b)}$.

To see that we have a U-shape with text T'_b:

1. $\exists \ell : W_{h^*(\rho_b[\ell])} = W_{e(b)}$ (as stated just above);
2. $W_{h^*(\rho_b \diamond \xi)} \neq W_{e(b)}$ (by ξ witnessing $Q(b, \tau_b)$); and
3. $\exists \ell : W_{h^*(\rho_b \diamond \xi \diamond t[\ell])} = W_{e(b)}$ (as T'_b is a text for $W_{e(b)} \in \mathcal{L}$).

This is a contradiction to $\mathcal{L} \in \mathbf{NUBMS}_n(\langle h, f\rangle)$. □

4 Memoryless Feedback Learning (MLF)

The main theorem in this section is Theorem 5 just below. This theorem answers the open question mentioned in Section 1.3 above regarding memory-less feedback learning.

Theorem 6 shows that memoryless learning with arbitrarily many feedback queries is equivalent to **TxtEx**-learning *on classes of infinite languages.*

Memoryless feedback learning, as defined above, trivially allows a learner to query for whether the *current input element* has been seen *strictly previously.* In Definition 8 below we give a variant of memoryless feedback learning, called MLF′ learning, where all queries are answered based on *all* data seen so far, *including* the current datum. For MLF′ learning, it is no longer possible to query to see if the current datum has been seen previously. From Theorem 9 we have that the learning power of MLF′ learning is strictly lower.

Some results in this section make use of the following definition.

Definition 4. Let $h_0 \in \mathcal{P}$ be such that $\forall i, x, D$:

$$h_0(i, x, D) = \begin{cases} \emptyset, & \text{if } x = \# \text{ and } i = 0;\\ ?, & \text{if } x = \# \text{ and } i \neq 0;\\ \varphi_x(i, D), & \text{otherwise.} \end{cases} \qquad (35)$$

Let $\mathcal{L}_0 = \mathbf{MLF}_1(h_0)$.

Intuitively, h_0 is a learner which learns languages by interpreting each input datum as a program for the computations to make to get an appropriate output.

In fact, one can argue that not much information about how to identify $\mathcal{L}_0$ is actually in h_0; instead, one could say that "$\mathcal{L}_0$ learns itself." We call such classes of languages "self-learning classes of languages." Using such classes to show something is learnable with respect to one criterion and *not* with respect to some other is beneficial in two ways. We explain using $\mathcal{L}_0$ as an example. First, trivially, $\mathcal{L}_0$ is $\mathbf{MLF}_1$-learnable – $\mathcal{L}_0$ was defined to be the set of all languages so learnable by the explicitly given learner h_0. Secondly, one merely has to use recursion theorems, such as variants of ORT, to diagonalize out of $\mathcal{L}_0$. We so use the self-learning $\mathcal{L}_0$ in the proofs of Theorems 5 and 9 (but the latter proof is omitted herein) and we employ a variant in the (omitted) proof of Theorem 7.

Theorem 5

$$\mathbf{MLF}_1 \setminus \mathbf{NUMLR}_* \neq \emptyset.$$

Proof. We consider $\mathcal{L}_0$ from Definition 4. Suppose $h_1 \in \mathcal{P}$ $\mathbf{MLR}_*$-learns $\mathcal{L}_0$. We show that h_1 is *not* non-U-shaped. It is easy to see that we can assume, without loss of generality,

$$\forall \tau, x : h_1^*(\tau \diamond x \diamond x) \in \mathbb{N} \Rightarrow h_1^*(\tau \diamond x) = h_1^*(\tau \diamond x \diamond x). \tag{36}$$

Let $f \in \mathcal{P}$ be such that, on input x, f first computes $h_1(x)$ where all queries are answered with "false". If $h_1(x)\downarrow$, f outputs the maximum recalled element plus 1 (or 0, if no queries were asked). Then we have

$$\forall x : h_1^*(x)\downarrow \Rightarrow f(x)\downarrow \text{ and } h_1^* \text{ on } x \text{ does not recall any } y \geq f(x). \tag{37}$$

By padded ORT, there are $e_0, e_1, a \in \mathbb{N}$ and strictly monotonic increasing functions $\hat{b}, \hat{c} \in \mathcal{R}$ such that $\hat{b}$ and $\hat{c}$ have disjoint ranges, neither containing a, and, abbreviating

$$\begin{aligned} b &= \hat{b}(f(a)); \\ c &= \lambda i \centerdot \hat{c}(i + f(a)); \\ E &= \{c(i) \mid \exists j \geq i : h_1^*(a \diamond c[j])\downarrow \neq h_1^*(a \diamond c[j+1])\downarrow\} \end{aligned}$$

we have $\forall i, x :$

$$W_{e_0} = \{a\} \cup \mathrm{range}(c); \tag{38}$$

$$W_{e_1} = \begin{cases} \{a\}, & \text{if } h_1^*(a)\uparrow \text{ or } h_1^*(a) = ?; \\ \{a, b\} \cup E, & \text{otherwise}; \end{cases} \tag{39}$$

$$\varphi_a(x) = \begin{cases} ?, & \text{if } \mathtt{rcl}(a); \\ e_1, & \text{otherwise}; \end{cases} \tag{40}$$

$$\varphi_{\hat{b}(i)}(x) = \begin{cases} ?, & \text{if } \mathtt{rcl}(\hat{b}(i)); \\ e_1, & \text{otherwise}; \end{cases} \tag{41}$$

$$\varphi_{\hat{c}(i)}(x) = \begin{cases} e_1, & \text{if } \mathtt{rcl}(b); \\ e_0, & \text{otherwise}. \end{cases} \tag{42}$$

Claim 3. $h_1^*(a) \in \mathbb{N}$.

Proof of Claim 3. Suppose, by way of contradiction, otherwise. We have $\{a\} = W_{e_1} \in \mathcal{L}_0$. If $h_1^*(a)\uparrow$, then h_1 would not learn $W_{e_1} \in \mathcal{L}_0$, a contradiction. Hence, $h_1^*(a) =?$. Using (36), we get that h_1, on the text $\lambda i \centerdot a$, does not learn $\{a\} = W_{e_1} \in \mathcal{L}_0$, a contradiction. □ (for Claim 3)

Using (37) and Claim 1, we have $f(a)\downarrow$. Hence, b is defined and c is total. It is now easy to see that

$$W_{e_0} \in \mathcal{L}_0. \tag{43}$$

Therefore, h_1 $\mathbf{MLR}_*$-learns W_{e_0}. Let k be minimal such that

$$\forall i \geq k : h_1^*(a \diamond c[i]) \in \{?, h_1^*(a \diamond c[k])\}. \tag{44}$$

In particular,

$$W_{h_1^*(a \diamond c[k])} = W_{e_0}. \tag{45}$$

Hence, W_{e_1} is the finite set $\{a, b, c(0)\} \cup \text{content}(c[k])$. It is easily verified that $W_{e_1} \in \mathcal{L}_0$.

Note that, as $\hat{b}$ and $\hat{c}$ are strictly monotone increasing,

$$\forall x \in \{b\} \cup \text{range}(c) : x \geq f(a). \tag{46}$$

Hence, by choice of f (see (37)), h_1 on a cannot recall any $x \in \{b\} \cup \text{range}(c)$. Therefore,

$$h_1^*(a) = h_1^*(c[k] \diamond b \diamond a). \tag{47}$$

Claim 4. $W_{h_1^*(a)} = W_{e_1}$.

Proof of Claim 4. From Claim 1 we have that $h_1^*(a) \in \mathbb{N}$. As h_1 $\mathbf{MLR}_*$-identifies W_{e_1} from the text $c[k] \diamond b \diamond \lambda i \centerdot a$ and using (36),

$$W_{h_1^*(c[k] \diamond b \diamond a)} = W_{e_1}. \tag{48}$$

Using (47), we get the claim. □ (for Claim 4)

We consider the text $T = a \diamond c[k] \diamond b \diamond \lambda x \centerdot \#$ for W_{e_1}. The following shows that h_1 has a U-shape on T.

1. $W_{h_1^*(a)} = W_{e_1}$ by Claim 2;
2. $W_{h_1^*(a \diamond c[k])} = W_{e_0}$ by (45);
3. $\exists j \geq k+1 : W_{h_1^*(T[j])} = W_{e_1}$ as h_1 $\mathbf{MLR}_*$-identifies W_{e_1}.

Therefore, h_1 is not non-U-shaped on $\mathcal{L}_0$. □

With $\text{Pow}(\mathcal{E}_\infty)$ we denote the powerset of all infinite `ce` sets.

Theorem 6

$$\text{Pow}(\mathcal{E}_\infty) \cap \mathbf{NUMLF}_* = \text{Pow}(\mathcal{E}_\infty) \cap \mathbf{TxtEx}.$$

Theorem 7

$$(\mathbf{MLF}_1 \cap \mathrm{Pow}(\mathcal{E}_\infty)) \setminus \bigcup_{n \in \mathbb{N}} \mathbf{NUMLF}_n \neq \emptyset.$$

Note that **MLF** allows for a learner to determine (at the cost of a query) whether the current datum has been seen previously (i.e., is a repetition), and the previous proofs in this section sometimes made use of this ability. The next theorem states that this ability is important for learning power.

Definition 8. Call $\mathbf{MLF}'$ the sequence generating functional that one gets by modifying **MLF** to answer *true* to recalls for the current datum.

The sequence generating functional $\mathbf{MLF}'$ destroys a learners ability to determine whether the current datum has been seen previously (i.e., is a repetition).

Theorem 9

$$\mathbf{MLF}_1 \setminus \mathbf{MLF}'_* \neq \emptyset.$$

References

[BCM+08] Baliga, G., Case, J., Merkle, W., Stephan, F., Wiehagen, W.: When unlearning helps. Information and Computation 206, 694–709 (2008)

[Car82] Carey, S.: Face perception: Anomalies of development. In: Strauss, S., Stavy, R. (eds.) U-Shaped Behavioral Growth. Developmental Psychology Series. Academic Press, NY (1982)

[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)

[Cas99] Case, J.: The power of vacillation in language learning. SIAM Journal on Computing 28, 1941–1969 (1999)

[CCJS07a] Carlucci, L., Case, J., Jain, S., Stephan, F.: Non-U-shaped vacillatory and team learning. Journal of Computer and System Sciences, Special Issue in Memory of Carl Smith (2007)

[CCJS07b] Carlucci, L., Case, J., Jain, S., Stephan, F.: Results on memory-limited U-shaped learning. Information and Computation 205, 1551–1573 (2007)

[CJLZ99] Case, J., Jain, S., Lange, S., Zeugmann, T.: Incremental concept learning for bounded data mining. Information and Computation 152, 74–110 (1999)

[CL82] Case, J., Lynes, C.: Machine inductive inference and language identification. In: Nielsen, M., Schmidt, E.M. (eds.) ICALP 1982. LNCS, vol. 140, pp. 107–115. Springer, Heidelberg (1982)

[FJO94] Fulk, M., Jain, S., Osherson, D.: Open problems in Systems That Learn. Journal of Computer and System Sciences 49, 589–604 (1994)

[FKS95] Freivalds, R., Kinber, E., Smith, C.: On the impact of forgetting on learning machines. Journal of the ACM 42, 1146–1168 (1995)

[Gol67] Gold, E.: Language identification in the limit. Information and Control 10, 447–474 (1967)

[JK09] Jain, S., Kinber, E.: Iterative learning from texts and counterexamples using additional information. In: Gavaldà, R., Lugosi, G., Zeugmann, T., Zilles, S. (eds.) ALT 2009. LNCS, vol. 5809, pp. 308–322. Springer, Heidelberg (2009)

[JLMZ10] Jain, S., Lange, S., Moelius III, S.E., Zilles, S.: Incremental learning with temporary memory. Theoretical Computer Science 411, 2757–2772 (2010)

[JORS99] Jain, S., Osherson, D., Royer, J., Sharma, A.: Systems that Learn: An Introduction to Learning Theory, 2nd edn. MIT Press, Cambridge (1999)

[KS95] Kinber, E., Stephan, F.: Language learning from texts: Mind changes, limited memory and monotonicity. Information and Computation 123, 224–241 (1995)

[LZ96] Lange, S., Zeugmann, T.: Incremental learning from positive data. Journal of Computer and System Sciences 53, 88–103 (1996)

[MPU$^+$92] Marcus, G., Pinker, S., Ullman, M., Hollander, M., Rosen, T.J., Xu, F.: Overregularization in Language Acquisition. In: Monographs of the Society for Research in Child Development, vol. 57(4). University of Chicago Press, Chicago (1992); Includes commentary by H. Clahsen

[OSW86] Osherson, D., Stob, M., Weinstein, S.: Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists. MIT Press, Cambridge (1986)

[OW82] Osherson, D., Weinstein, S.: Criteria of language learning. Information and Control 52, 123–138 (1982)

[RC94] Royer, J., Case, J.: Subrecursive Programming Systems: Complexity and Succinctness. In: Research monograph in Progress in Theoretical Computer Science. Birkhäuser, Basel (1994)

[Rog67] Rogers, H.: Theory of Recursive Functions and Effective Computability. McGraw Hill, New York (1967); Reprinted by MIT Press, Cambridge, Massachusetts (1987)

[SS82] Strauss, S., Stavy, R. (eds.): U-Shaped Behavioral Growth. Developmental Psychology Series. Academic Press, NY (1982)

[TA02] Taatgen, N.A., Anderson, J.R.: Why do children learn to say broke? A model of learning the past tense without feedback. Cognition 86, 123–155 (2002)

[WC80] Wexler, K., Culicover, P.: Formal Principles of Language Acquisition. MIT Press, Cambridge (1980)

[Wie76] Wiehagen, R.: Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. Elektronische Informationverarbeitung und Kybernetik 12, 93–99 (1976)

Learning without Coding

Samuel E. Moelius III[1] and Sandra Zilles[2]

[1] IDA Center for Computing Sciences
17100 Science Drive, Bowie, MD 20715-4300
semoeli@super.org
[2] Department of Computer Science
University of Regina
Regina, Saskatchewan, Canada S4S 0A2
zilles@cs.uregina.ca

Abstract. Iterative learning is a model of language learning from positive data, due to Wiehagen. When compared to a learner in Gold's original model of language learning from positive data, an iterative learner can be thought of as *memory-limited*. However, an iterative learner can memorize *some* input elements by *coding* them into the syntax of its hypotheses. A main concern of this paper is: to what extent are such coding tricks *necessary*?

One means of preventing *some* such coding tricks is to require that the hypothesis space used be free of redundancy, i.e., that it be 1-1. By extending a result of Lange & Zeugmann, we show that many interesting and non-trivial classes of languages can be iteratively identified in this manner. On the other hand, we show that there exists a class of languages that can*not* be iteratively identified using any 1-1 effective numbering as the hypothesis space.

We also consider an iterative-like learning model in which the computational component of the learner is modeled as an *enumeration operator*, as opposed to a partial computable function. In this new model, there are no hypotheses, and, thus, no syntax in which the learner can encode what elements it has or has not yet seen. We show that there exists a class of languages that *can* be identified under this new model, but that can*not* be iteratively identified. On the other hand, we show that there exists a class of languages that can*not* be identified under this new model, but that *can* be iteratively identified using a Friedberg numbering as the hypothesis space.

Keywords: Coding tricks, inductive inference, iterative learning.

1 Introduction

Iterative learning (**It**-learning, Definition 1(a)) is a model of language learning from positive data, due to Wiehagen [Wie76]. Like many models based on positive data, the **It**-learning model involves a learner that is repeatedly fed elements drawn from $\{\#\}$ and from some unknown target language $L \subseteq \mathbb{N}$, where $\mathbb{N}$ is

© Springer-Verlag Berlin Heidelberg 2010

the set of natural numbers, $\{0, 1, 2, ...\}$.[1] After being fed each such element, the learner outputs a hypothesis (provided that the learner does not diverge). The learner is said to *identify* the target language L iff there is some point from whence on the learner outputs only one hypothesis, and that hypothesis corresponds to L. Furthermore, the learner is said to identify a *class* of languages $\mathcal{L}$ iff the learner identifies *each* $L \in \mathcal{L}$ when fed the elements of L (and possibly #).

In the **It**-learning model, the learner itself is modeled as a triple.

– The first element of the triple is a two-place partial computable function, whose arguments are, respectively, the learner's most recently output hypothesis, and the next input element.
– The second element of the triple is a preliminary hypothesis, i.e., the hypothesis output by the learner before being fed any input.
– The third element of the triple is a *hypothesis space*. The *hypothesis space* determines the language that corresponds to each of the learner's hypotheses. Formally, a hypothesis space is a numbering $(X_j)_{j\in\mathbb{N}}$ of some collection of subsets of $\mathbb{N}$, and that is *effective* in the sense that the two-place predicate $\lambda j, x \centerdot [x \in X_j]$ is partial computable.[2]

It-learning is a special case of Gold's original model of language learning from positive data [Gol67]. In Gold's original model, the learner is provided access to all previously seen input elements, in addition to the next input element. In this sense, a learner in Gold's model can be thought of as *memorizing* all previously seen input elements. When compared to learners in Gold's model, iterative learners are restricted in terms of the classes of languages that they can identify.[3] In this sense, the *memory-limited* aspect of iterative learners is a *true* restriction, and *not* a mere superficial difference in definitions.

This does not however mean that iterative learners are *memory-less*. In particular, an iterative learner can memorize *some* input elements by employing *coding tricks*, which we define (informally) as follows.

– A *coding trick* is any use by an iterative learner of the syntax of a hypothesis to determine what elements that learner has or has not yet seen.

The following is an example. Suppose that an iterative learner $(M, p, (X_j)_{j\in\mathbb{N}})$ identifies a class of languages $\mathcal{L}$. Further suppose that one desires a learner that identifies the class $\mathcal{L}'$, where

$$\mathcal{L}' = \mathcal{L} \cup \{L \cup \{0\} \mid L \in \mathcal{L}\}. \tag{1}$$

[1] The symbol '#' is pronounced "pause". The inclusion of # in the model allows the target language L to be empty, i.e., in such a case, the learner is repeatedly fed #.

[2] *Not-necessarily-effective* hypothesis spaces have also been considered [dBY10]. However, such hypothesis spaces are not needed herein. For the remainder, we use the terms *hypothesis space* and *effective numbering* interchangeably.

[3] Many variants of the **It**-learning model have been considered, and have also been shown to be restricted in this sense [LZ96, CCJS07, JLMZ10].

Such a learner $(M', p', (Y_k)_{k\in\mathbb{N}})$ may be obtained as follows. Let $(Y_k)_{k\in\mathbb{N}}$ be such that, for each j:

$$Y_{2j} = X_j; \qquad\qquad Y_{2j+1} = X_j \cup \{0\}.$$

Then, let M' be such that, for each $x \in (\mathbb{N} \cup \{\#\}) - \{0\}$:

$$\begin{aligned} M'(2j, \quad x) &= 2M(j,x); & M'(2j, \quad 0) &= 2M(j,0)+1;\\ M'(2j+1,x) &= 2M(j,x)+1; & M'(2j+1,0) &= 2M(j,0)+1. \end{aligned}$$

It is easily seen that $(M', 2p, (Y_k)_{k\in\mathbb{N}})$ iteratively identifies $\mathcal{L}'$. Intuitively, M' simulates M, while using the least-significant bit of each hypothesis to *encode* whether or not M' has seen a 0 (e.g., M' switches from an even to an odd hypothesis when it sees a 0). Further note that, if $\mathcal{L}$ already contains languages for which 0 is a member, then there is *redundancy* in the hypothesis space $(Y_k)_{k\in\mathbb{N}}$. In particular, if $0 \in X_j$, then $Y_{2j} = Y_{2j+1}$. For such hypotheses, the least-significant bit affects *only* their syntax, and *not* their semantics.

This example demonstrates how coding tricks can at least *facilitate* the identification of a class of languages. A main concern of this paper is: to what extent are such coding tricks *necessary*?

One approach to preventing *some* such coding tricks is to require that the hypothesis space be free of redundancy, i.e., that it be 1-1. One means of doing this is to require that the hypothesis space be a *Friedberg numbering* [Fri58, Kum90]. A *Friedberg numbering* is a 1-1 effective numbering of all computably enumerable (ce) subsets of $\mathbb{N}$. The use of such numberings as hypothesis spaces was considered by Jain & Stephan [JS08].[4] They observed, for example, that $\mathcal{Fin}$, the collection of all finite subsets of $\mathbb{N}$, can*not* be iteratively identified using a Friedberg numbering as the hypothesis space [JS08, Remark 28]. For the remainder, to **FrIt**-*identify* a class of languages $\mathcal{L}$ shall mean to iteratively identify $\mathcal{L}$ using a Friedberg numbering as the hypothesis space (see Definition 1(b)).

Our first main result is to show that, despite this observation of Jain & Stephan, many interesting and non-trivial classes can be **FrIt**-identified. More specifically, we extend a result of Lange & Zeugmann [LZ96, Theorem 12] by showing that, for each class $\mathcal{L}$, if there exists a single hypothesis space witnessing that $\mathcal{L}$ is both uniformly decidable and computably finitely thick (see Definition 3 below), then $\mathcal{L}$ can be **FrIt**-identified (Theorem 6). By comparison, Lange & Zeugmann showed that such a class can be **It**-identified. One significant application of this result is the following. A *pattern language* [Ang80] is a type of language with applications to molecular biology (see, e.g., [SSS$^+$94]). Furthermore, the pattern languages naturally form classes that are **It**-identifiable by Lange & Zeugmann's result,[5] and, thus, are **FrIt**-identifiable, by ours.

As the reader may have already noticed, if one's intent is simply to eliminate redundancy in the hypothesis space, then to require that the hypothesis space be a Friedberg numbering is really overkill. That is because to require that

[4] Freivalds, et al. [FKW82] considered the use of Friedberg numberings as hypothesis spaces in the context of *function* learning.

[5] The pattern languages were first shown to be **It**-identifiable by Lange & Wiehagen [LW91].

the hypothesis space be a Friedberg numbering is to require that it be free of redundancy *and* that it represent all of the ce sets.

Thus, we consider a milder variant of **FrIt**-learning, which we call *injective iterative learning* (**InjIt**-learning, Definition 1(c)). In this variant, the hypothesis space is required to be free of redundancy (i.e., be 1-1), but need not represent all of the ce sets.[6] Clearly, for each class $\mathcal{L}$, if $\mathcal{L}$ can be **FrIt**-identified, then $\mathcal{L}$ can be **InjIt**-identified. On the other hand, $\mathcal{Fin}$ can be **InjIt**-identified, but, as per Jain & Stephan's observation mentioned above, $\mathcal{Fin}$ can*not* be **FrIt**-identified.

Going further, if one's intent is to *prevent coding tricks*, then to require that the hypothesis space be free of redundancy may still be overkill. In particular, one might allow that there be redundancy in the hypothesis space, but require that the learner not benefit from this redundancy. This idea is captured in our next model, which is called *extensional iterative learning* (**ExtIt**-learning, Definition 1(d)).

For a learner to **ExtIt**-identify a class of languages, it is required that, when presented with equivalent hypotheses and identical input elements, the learner must produce equivalent hypotheses. More formally: suppose that $\mathcal{L}$ is a class of languages, that σ_0 and σ_1 are two non-empty sequences of elements drawn from $\{\#\}$ and from two (possibly distinct) languages in $\mathcal{L}$, and that the following conditions are satisfied.

- When fed all but the last elements of σ_0 and σ_1, the learner outputs hypotheses for the same language (though those hypotheses may differ syntactically).
- The last elements of σ_0 and σ_1 are identical.

Then, for the learner to **ExtIt**-identify $\mathcal{L}$, it is required that:

- When fed *all* of σ_0 and σ_1, the learner outputs hypotheses for the same language (though those hypotheses may differ syntactically).

Clearly, if a learner identifies a class of languages using a 1-1 hypothesis space, then that learner satisfies the just above requirement. Thus, every class of languages that can be **InjIt**-identified can be **ExtIt**-identified. On the other hand, we show that there exists a class of languages that *can* be **ExtIt**-identified, but that can*not* be **InjIt**-identified (Theorem 9).

Before introducing our final model, let us recall the definition of an *enumeration operator* [Rog67, §9.7].[7] Let $\mathcal{P}(\mathbb{N})$ be the powerset of $\mathbb{N}$, i.e., the collection of all subsets of $\mathbb{N}$. Let $\langle \cdot, \cdot \rangle$ be any pairing function, i.e., a computable, 1-1, onto function of type $\mathbb{N}^2 \to \mathbb{N}$ [Rog67, page 64]. Let $\hat{\#} = 0$, and, for each $x \in \mathbb{N}$, let $\hat{x} = x + 1$. Let $(D_j)_{j \in \mathbb{N}}$ be any canonical enumeration of $\mathcal{Fin}$.

An *enumeration operator* of type $\mathcal{P}(\mathbb{N}) \times (\mathbb{N} \cup \{\#\}) \to \mathcal{P}(\mathbb{N})$ is a mapping that is algorithmic in the following precise sense. To each enumeration operator

[6] The use of 1-1 hypothesis spaces was also considered in [BBCJS10] in the context of learning certain *specific* classes of languages.

[7] Herein, we focus on the enumeration operators of a particular type. A more general definition can be found in [MZ10].

Θ (of the given type), there corresponds a ce set H, such that, for each $X \subseteq \mathbb{N}$ and $x \in \mathbb{N} \cup \{\#\}$,

$$\Theta(X, x) = \{y \mid \langle j, \langle \hat{x}, y\rangle\rangle \in H \ \wedge \ D_j \subseteq X\}. \tag{2}$$

Thus, given an *enumeration of* X, and given x, one can enumerate $\Theta(X, x)$ in the following manner.

– Enumerate H. For each element of the form $\langle j, \langle \hat{x}, y\rangle\rangle \in H$, if ever the finite set D_j appears in the enumeration of X, then list y into $\Theta(X, x)$.

Enumeration operators exhibit certain notable properties, including *monotonicity* [Rog67, Theorem 9-XXI]: for each enumeration operator $\mathcal{M}$ of type $\mathcal{P}(\mathbb{N}) \times (\mathbb{N} \cup \{\#\}) \to \mathcal{P}(\mathbb{N})$, each $X, Y \subseteq \mathbb{N}$, and each $x \in \mathbb{N} \cup \{\#\}$,

$$X \subseteq Y \ \Rightarrow \ \mathcal{M}(X, x) \subseteq \mathcal{M}(Y, x). \tag{3}$$

Intuitively, this means that an enumeration operator can tell from its set argument X what elements are in X, but it cannot tell from X what elements are in the *complement of* X.

The final model that we consider is called *iterative learning by enumeration operator* (**EOIt**-learning, Definition 1(e)). As the name suggests, the computational component of the learner is modeled as an enumeration operator, as opposed to a partial computable function. Specifically, the learner is modeled as a *pair*, where:

– The first element of the pair is an enumeration operator of type $\mathcal{P}(\mathbb{N}) \times (\mathbb{N} \cup \{\#\}) \to \mathcal{P}(\mathbb{N})$, whose arguments are, respectively, the learner's most recently output *language*, and the next input element.
– The second element of the pair is the learner's preliminarily output *language*, i.e., the language output by the learner before being fed any input. (We require that this preliminary language be ce.)

Thus, there are no hypotheses in this model. Since there are no hypotheses, there is no *syntax* in which the learner can encode what elements it has or has not yet seen.

The expulsion of hypotheses from the model has an additional consequence, and that is that the success criterion has to be adjusted. Specifically, we say that a learner in this model *identifies* a language L iff when fed the elements of L (and possibly #), there is some point from whence on the learner outputs only the *language* L. The success criterion for identifying a *class* of languages is adjusted similarly. This more liberal approach to language identification, in some sense, gives an advantage to learners in this model. In particular, there exists a class of languages that *can* be **EOIt**-identified, but that can*not* be **It**-identified (Corollary 13).

Interestingly, there also exists a class of languages that can*not* be **EOIt**-identified, but that *can* be **FrIt**-identified (Theorem 14). To help to see why, consider the following two scenarios. First, suppose that $(\mathcal{M}, X)$ is a learner in

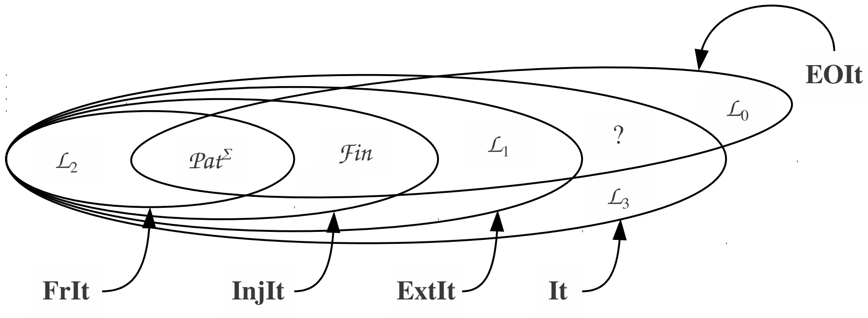


Fig. 1. A summary of main results and open problems. $\mathcal{P}at^{\Sigma}$ is the collection of all pattern languages over Σ, where Σ is an arbitrary alphabet. $\mathcal{F}in$ is the collection of all finite subsets of $\mathbb{N}$. The classes $\mathcal{L}_0$, $\mathcal{L}_1$, and $\mathcal{L}_2$ are defined in the proofs of Theorems 7, 9, and 14, respectively. The existence of the class $\mathcal{L}_3$ was shown by Jain (see Theorem 10).

the enumeration operator model, and that Y is its most recently output language. Then, since $\mathcal{M}$ is an enumeration operator, $\mathcal{M}$ can tell from Y what elements are in Y, but it cannot tell from Y what elements are in the *complement of* Y. Next, consider the analogous situation for a conventional iterative learner. That is, suppose that $(M, p, (X_j)_{j\in\mathbb{N}})$ is such a learner, and that j is its most recently output hypothesis. Then, in many cases, M *can* tell from j what elements are in the complement of X_j. In this sense, one could say that a hypothesis implicitly encodes *negative* information about the language that it represents. (In fact, this phenomenon can clearly be seen in the full proof of Theorem 14. See [MZ10, Theorem 20].)

A question to then ask is: is this a *coding trick*, i.e., is it the case that *every* learner that operates on hypotheses (as opposed to languages) is employing coding tricks? At present, we do not see a clear answer to this question. Thus, we leave it as a subject for further study.

The main points of the preceding paragraphs are summarized in Figure 1. The remainder of this paper is organized as follows. Section 2 covers preliminaries. Section 3 presents our results concerning uniformly decidable and computably finitely thick classes of languages. Section 4 presents our results concerning Friedberg, injective, and extensional iterative learning (**FrIt**, **InjIt**, and **ExtIt**-learning, respectively). Section 5 presents our results concerning iterative learning by enumeration operator (**EOIt**-learning).

Due to space constraints, only partial proofs are given. Complete proofs of all of our results can be found in the associated tech-report [MZ10].

2 Preliminaries

Computability-theoretic concepts not covered below are treated in [Rog67].

Lowercase math-italic letters (e.g., a, j, x), with or without decorations, range over elements of $\mathbb{N}$, unless stated otherwise. Uppercase italicized letters (e.g.,

A, J, X), with or without decorations, range over subsets of $\mathbb{N}$, unless stated otherwise. For each non-empty X, $\min X$ denotes the minimum element of X. $\min\emptyset \stackrel{\text{def}}{=} \infty$. For each non-empty, finite X, $\max X$ denotes the maximum element of X. $\max\emptyset \stackrel{\text{def}}{=} -1$. The symbol $\mathcal{L}$, with or without decorations, ranges over subsets of $\mathcal{P}(\mathbb{N})$, unless stated otherwise.

For each x, $\langle x\rangle \stackrel{\text{def}}{=} x$. For each $x_0, ..., x_{n-1}$, where $n > 2$, $\langle x_0, ..., x_{n-1}\rangle \stackrel{\text{def}}{=} \big\langle x_0, \langle x_1, ..., x_{n-1}\rangle\big\rangle$.

$\mathbb{N}_\# \stackrel{\text{def}}{=} \mathbb{N} \cup \{\#\}$. A *text* is a total function of type $\mathbb{N} \to \mathbb{N}_\#$. For each text t and $i \in \mathbb{N}$, $t[i]$ denotes the initial segment of t of length i. For each text t, $\text{content}(t) \stackrel{\text{def}}{=} \{t(i) \mid i \in \mathbb{N}\} - \{\#\}$. For each text t and $L \subseteq \mathbb{N}$, t is a text *for* L $\stackrel{\text{def}}{\Leftrightarrow} \text{content}(t) = L$.

Seq denotes the set of all initial segments of texts. Lowercase Greek letters (e.g., ρ, σ, τ), with or without decorations, range over elements of Seq, unless stated otherwise. λ denotes the empty initial segment (equivalently, the everywhere divergent function). For each σ, $|\sigma|$ denotes the length of σ (equivalently, the size of the domain of σ). For each σ and $i \le |\sigma|$, $\sigma[i]$ denotes the initial segment of σ of length i. For each σ, $\text{content}(\sigma) \stackrel{\text{def}}{=} \{\sigma(i) \mid i < |\sigma|\} - \{\#\}$. For each σ and τ, $\sigma \cdot \tau$ denotes the concatenation of σ and τ. For each $\sigma \in \text{Seq} - \{\lambda\}$:

$$\sigma^- \stackrel{\text{def}}{=} \sigma[|\sigma| - 1]; \qquad \text{last}(\sigma) \stackrel{\text{def}}{=} \sigma(|\sigma| - 1).$$

For each L and $\mathcal{L}$, $\text{Txt}(L)$, $\text{Txt}(\mathcal{L})$, $\text{Seq}(L)$, and $\text{Seq}(\mathcal{L})$ are defined as follows.

$$\begin{array}{ll} \text{Txt}(L) = \{\, t \mid t \text{ is a text for } L\}. & \text{Seq}(L) = \{\sigma \mid \text{content}(\sigma) \subseteq L\}. \\ \text{Txt}(\mathcal{L}) = \{\, t \mid (\exists L \in \mathcal{L})[t \in \text{Txt}(L)]\}. & \text{Seq}(\mathcal{L}) = \{\sigma \mid (\exists L \in \mathcal{L})[\sigma \in \text{Seq}(L)]\}. \end{array}$$

For each one-argument partial function ψ and $x \in \mathbb{N}$, $\psi(x){\downarrow}$ denotes that $\psi(x)$ converges; $\psi(x){\uparrow}$ denotes that $\psi(x)$ diverges. We use $\uparrow$ to denote the value of a divergent computation.

$\mathcal{EN}$ denotes the collection of all effective numberings. $\mathcal{CE}$ denotes the collection of all computably enumerable (ce) subsets of $\mathbb{N}$. For each m and n, $\mathcal{PC}_{m,n}$ denotes the collection of partial computable functions mapping $\mathbb{N}^m \times \mathbb{N}_\#^n$ to $\mathbb{N}$. We shall be concerned primarily with $\mathcal{PC}_{1,0}$ and $\mathcal{PC}_{1,1}$. $(\varphi_p)_{p\in\mathbb{N}}$ denotes any fixed, acceptable numbering of $\mathcal{PC}_{1,0}$. For each i, $W_i \stackrel{\text{def}}{=} \{x \mid \varphi_i(x){\downarrow}\}$. Thus, $(W_i)_{i\in\mathbb{N}}$ is an effective numbering of $\mathcal{CE}$.

Iter $\stackrel{\text{def}}{=} \mathcal{PC}_{1,1} \times \mathbb{N} \times \mathcal{EN}$. For each $M \in \mathcal{PC}_{1,1}$ and p, the partial function M_p^* is such that, for each $\sigma \in \text{Seq}$ and $x \in \mathbb{N}_\#$:

$$M_p^*(\lambda) = p; \qquad M_p^*(\sigma \cdot x) = \begin{cases} M\big(M_p^*(\sigma), x\big), & \text{if } M_p^*(\sigma){\downarrow}; \\ \uparrow, & \text{otherwise.} \end{cases}$$

$\mathcal{EO}_{1,1}$ denotes the collection of all enumeration operators of type $\mathcal{P}(\mathbb{N}) \times \mathbb{N}_\# \to \mathcal{P}(\mathbb{N})$. For each $\mathcal{M} \in \mathcal{EO}_{1,1}$ and X, the function $\mathcal{M}_X^* : \text{Seq} \to \mathcal{P}(\mathbb{N})$ is such that, for each $\sigma \in \text{Seq}$ and $x \in \mathbb{N}_\#$:

$$\mathcal{M}_X^*(\lambda) = X; \qquad \mathcal{M}_X^*(\sigma \cdot x) = \mathcal{M}\big(\mathcal{M}_X^*(\sigma), x\big).$$

The following are the formal definitions of the learning models described in Section 1. The symbols **Fr**, **Inj**, **Ext**, and **EO** are mnemonic for *Friedberg*, *injective*, *extensional*, and *enumeration operator*, respectively.

Definition 1. For each $\mathcal{L}$, (a)-(e) below. In parts (a)-(d), $(M, p, (X_j)_{j\in\mathbb{N}}) \in \mathcal{I}ter$. In part (e), $(\mathcal{M}, X) \in \mathcal{EO}_{1,1} \times \mathcal{CE}$.

(a) **(Wiehagen [Wie76])** $(M, p, (X_j)_{j\in\mathbb{N}})$ **It**-*identifies* $\mathcal{L} \Leftrightarrow$ for each $t \in \mathrm{Txt}(\mathcal{L})$, there exists $i_0 \in \mathbb{N}$ such that $X_{M_p^*(t[i_0])} = \mathrm{content}(t)$, and, for each $i \geq i_0$, $\big[M_p^*(t[i]) = M_p^*(t[i_0])\big]$.
(b) **(Jain & Stephan [JS08])** $(M, p, (X_j)_{j\in\mathbb{N}})$ **FrIt**-*identifies* $\mathcal{L} \Leftrightarrow (M, p, (X_j)_{j\in\mathbb{N}})$ **It**-identifies $\mathcal{L}$, and $(X_j)_{j\in\mathbb{N}}$ is a Friedberg numbering.
(c) $(M, p, (X_j)_{j\in\mathbb{N}})$ **InjIt**-*identifies* $\mathcal{L} \Leftrightarrow (M, p, (X_j)_{j\in\mathbb{N}})$ **It**-identifies $\mathcal{L}$, and $(X_j)_{j\in\mathbb{N}}$ is 1-1.
(d) $(M, p, (X_j)_{j\in\mathbb{N}})$ **ExtIt**-*identifies* $\mathcal{L} \Leftrightarrow (M, p, (X_j)_{j\in\mathbb{N}})$ **It**-identifies $\mathcal{L}$, and, for each $\sigma_0, \sigma_1 \in \mathrm{Seq}(\mathcal{L}) - \{\lambda\}$,

$$[X_{M_p^*(\sigma_0^-)} = X_{M_p^*(\sigma_1^-)} \ \wedge\ \mathrm{last}(\sigma_0) = \mathrm{last}(\sigma_1)] \ \Rightarrow\ X_{M_p^*(\sigma_0)} = X_{M_p^*(\sigma_1)}. \quad (4)$$

(e) $(\mathcal{M}, X)$ **EOIt**-*identifies* $\mathcal{L} \Leftrightarrow$ for each $t \in \mathrm{Txt}(\mathcal{L})$, there exists $i_0 \in \mathbb{N}$ such that $(\forall i \geq i_0)\big[\mathcal{M}_X^*(t[i]) = \mathrm{content}(t)\big]$.

Definition 2. Let **It** be as follows.

$$\mathbf{It} = \big\{\mathcal{L} \mid \big(\exists (M, p, (X_j)_{j\in\mathbb{N}}) \in \mathcal{I}ter\big)[(M, p, (X_j)_{j\in\mathbb{N}}) \ \mathbf{It}\text{-identifies } \mathcal{L}]\big\}. \quad (5)$$

Let **FrIt**, **InjIt**, **ExtIt**, and **EOIt** be defined similarly.

3 Uniform Decidability and Computable Finite Thickness

In this section, we extend a result of Lange & Zeugmann by showing that, for each class $\mathcal{L}$, if there exists a single hypothesis space witnessing that $\mathcal{L}$ is both uniformly decidable and computably finitely thick, then $\mathcal{L}$ can be **FrIt**-identified (Theorem 6). We also show that there exists a class of languages that *is* uniformly decidable and computably finitely thick, but that is *not* in **It**, let alone **FrIt** (Theorem 7). Thus, one could not arrive at the conclusion of the just mentioned Theorem 6 if one were to merely require that: there exists a uniformly decidable effective numbering of $\mathcal{L}$, and a possibly *distinct* computably finitely thick effective numbering of $\mathcal{L}$.

The following are the formal definitions of the terms *uniformly decidable* and *computably finitely thick*. For additional background, see [LZZ08].

Definition 3

(a) An effective numbering $(X_j)_{j\in\mathbb{N}}$ is *uniformly decidable* $\Leftrightarrow$ the predicate $\lambda j, x \centerdot [x \in X_j]$ is decidable.
(b) A class of languages $\mathcal{L}$ is *uniformly decidable* $\Leftrightarrow$ there exists a uniformly decidable effective numbering of $\mathcal{L}$.
(c) An effective numbering $(X_j)_{j\in\mathbb{N}}$ is *computably finitely thick* $\Leftrightarrow$ there exists a computable function $f : \mathbb{N} \to \mathbb{N}$ such that, for each x,

$$\{X_j \mid j \in D_{f(x)}\} = \{L \mid x \in L \ \wedge\ (\exists j)[X_j = L]\}. \quad (6)$$

(d) **(Lange & Zeugmann [LZ96, Definition 9])** A class of languages $\mathcal{L}$ is *computably finitely thick* $\Leftrightarrow$ there exists a computably finitely thick effective numbering of $\mathcal{L}$.

N.B. In part (c) just above, the function f need *not* satisfy $D_{f(x)} = \{j \mid x \in X_j\}$. However, see the proof of Theorem 6 below.

Example 4

(a) $\mathcal{F}in$ *is* uniformly decidable, but is *not* computably finitely thick.
(b) $\mathcal{CE}$ is *neither* uniformly decidable *nor* computably finitely thick.
(c) The class $\big\{\{e\}, \{e\} \cup (W_e + e + 1) \mid e \in \mathbb{N}\big\}$ is *not* uniformly decidable, but *is* computably finitely thick.[8]
(d) The class $\{\mathbb{N} + e \mid e \in \mathbb{N}\}$ is both uniformly decidable and computably finitely thick. Moreover, there exists a single effective numbering witnessing both properties simultaneously.
(e) Let $\mathcal{L}$ be as follows.

$$\mathcal{L} = \big\{\{e\} \mid e \in \mathbb{N}\big\} \cup \big\{\{e, \varphi_e(0) + e + 1\} \mid e \in \mathbb{N} \ \wedge \ \varphi_e(0)\downarrow\big\}. \tag{7}$$

Then, $\mathcal{L}$ is both uniformly decidable and computably finitely thick,[9] but there is *no* effective numbering of $\mathcal{L}$ witnessing both properties simultaneously. In fact, no such numbering exists for any class containing $\mathcal{L}$.

The following result, due to Lange & Zeugmann, gives a sufficient condition for a class of languages to be **It**-identifiable.

Theorem 5 (Lange & Zeugmann [LZ96, Theorem 12]). For each $\mathcal{L}$, if there exists an effective numbering of $\mathcal{L}$ that is both uniformly decidable and computably finitely thick, then $\mathcal{L} \in \mathbf{It}$.[10]

The following result strengthens Theorem 5 (Lange & Zeugmann) just above.

Theorem 6. For each $\mathcal{L}$, if there exists an effective numbering of $\mathcal{L}$ that is both uniformly decidable and computably finitely thick, then $\mathcal{L} \in \mathbf{FrIt}$.

Proof (Sketch). Suppose that $\mathcal{L}$ satisfies the conditions of the theorem. Then it can be shown that there exists an effective numbering $(X_j)_{j\in\mathbb{N}}$ of $\mathcal{L}$ and a computable function $f : \mathbb{N} \to \mathbb{N}$ such that, for each x,

$$D_{f(x)} = \{j \mid x \in X_j\}. \tag{8}$$

For each x and J, say that x *narrows* J $\Leftrightarrow$ by letting $J' = \{j \in J \mid x \in X_j\}$,

$$\emptyset \neq J' \subset J \ \wedge \ \{w \in (\textstyle\bigcap_{j\in J'} X_j) \mid w < x\} = \{w \in (\textstyle\bigcap_{j\in J} X_j) \mid w < x\}. \tag{9}$$

[8] One can construct computably finitely thick effective numberings of the classes given in parts (c) and (e) of Example 4 using a technique similar to that used in Figure 3(b) below.

[9] See footnote 8.

[10] In [LZ96], Theorem 12 is not stated exactly as Theorem 5 is stated here. However, based on the proof of this result, we believe that what is stated here is what is meant.

List $\emptyset$ into $(Z_\ell)_{\ell\in\mathbb{N}}$ exactly once. Then, for each x, run survey(x).

survey(x): Act according the following conditions.

- Cond. (a) $[D_{f(x)} = \emptyset \ \vee \ \min(\bigcap_{j\in D_{f(x)}} X_j) \neq x]$. For each k, list $\{x\}\cup(Y_k+x+1)$ into $(Z_\ell)_{\ell\in\mathbb{N}}$ exactly once.
- Cond. (b) $[D_{f(x)} \neq \emptyset \ \wedge \ \min(\bigcap_{j\in D_{f(x)}} X_j) = x]$. List $(\bigcap_{j\in D_{f(x)}} X_j)$ into $(Z_\ell)_{\ell\in\mathbb{N}}$ exactly once, set $\zeta(D_{f(x)})$ to the index used to list this set, and run descend$(D_{f(x)}, x+1)$.

descend(J,x): Let J', x_0, and A be such that:

$$J' = \{j \in J \mid x \in X_j\}; \qquad x_0 = \min(\textstyle\bigcap_{j\in J} X_j); \qquad A = \{w \in (\textstyle\bigcap_{j\in J} X_j) \mid w < x\}.$$

Act according to the following conditions.

- Cond. (i) $\big[J' = J\big]$. For each k, list $A\cup(Y_k+x+1)$ into $(Z_\ell)_{\ell\in\mathbb{N}}$ exactly once, and run descend$(J, x+1)$.
- Cond. (ii) $\big[x \leq x_0 \ \vee \ [J' \subset J \ \wedge \ x$ does *not* narrow $J]\big]$. For each k, list $A\cup\{x\}\cup(Y_k+x+1)$ into $(Z_\ell)_{\ell\in\mathbb{N}}$ exactly once, and run descend$(J, x+1)$.
- Cond. (iii) $\big[x > x_0 \ \wedge \ x$ narrows $J\big]$. List $(\bigcap_{j\in J'} X_j)$ into $(Z_\ell)_{\ell\in\mathbb{N}}$ exactly once, set $\zeta(J')$ to the index used to list this set, and run both descend$(J, x+1)$ and descend$(J', x+1)$.

Fig. 2. The construction of $(Z_\ell)_{\ell\in\mathbb{N}}$ in the proof of Theorem 6

Let $(Y_k)_{k\in\mathbb{N}}$ be any Friedberg numbering.

An effective numbering $(Z_\ell)_{\ell\in\mathbb{N}}$ is constructed in Figure 2. The construction makes use of two procedures: survey and descend. The procedure survey takes one argument: an element of $\mathbb{N}$. The procedure descend takes two arguments: a finite subset of $\mathbb{N}$, and an element of $\mathbb{N}$.

In conjunction with $(Z_\ell)_{\ell\in\mathbb{N}}$, a partial computable function ζ from $\mathcal{Fin}$ to $\mathbb{N}$ is constructed. It is clear from the construction that ζ is 1-1, i.e., for each $J, J' \in \mathcal{Fin}$, $[\zeta(J){\downarrow} = \zeta(J') \ \Rightarrow \ J = J']$.

It can be shown that $(Z_\ell)_{\ell\in\mathbb{N}}$ is a Friedberg numbering. For ease of presentation, suppose that $Z_0 = \emptyset$. Let M be such that, for each $\ell > 0$ and x:

$$M(0,\#) = 0; \quad M(0,x) = \begin{cases} \zeta(D_{f(x)}), & \text{if } \zeta(D_{f(x)}){\downarrow}; \\ \uparrow, & \text{otherwise;} \end{cases}$$

$$M(\ell,\#) = \ell; \quad M(\ell,x) = \begin{cases} \zeta(J'), & \text{where } J \text{ and } J' \text{ are such that } \zeta(J) = \ell \text{ and} \\ & J' = \{j \in J \mid x \in X_j\}, \text{ if such a } J \text{ exists} \\ & \text{and } \zeta(J'){\downarrow}; \\ \uparrow, & \text{otherwise.} \end{cases}$$

It can be shown that $(M, 0, (Z_\ell)_{\ell\in\mathbb{N}})$ **FrIt**-identifies $\mathcal{L}$. $\approx\Box$ (**Theorem 6**)

(a) For each i, execute stage 0 below.

- Stage 0. For each i, include $(\mathbb{N}+i)$ and $\{i\}$ in $\mathcal{L}_0$. Go to stage 1.
- Stage 1. Let (M,p) be the ith pair in $((M,p)_i)_{i\in\mathbb{N}}$. Search for a $k \geq i$ such that

$$M_p^*\big((i\cdot\dots\cdot k)\cdot(k+1)\big)\!\downarrow \;=\; M_p^*\big((i\cdot\dots\cdot k)\cdot k\big) \;=\; M_p^*(i\cdot\dots\cdot k).$$

  If such a k is found, then include $\{i,...,k\}$ and $\{i,...,k+1\}$ in $\mathcal{L}_0$, and terminate the construction (for i). If *no* such k is found, then search indefinitely.

(b) For each i, execute stage 0 below.

- Stage 0. Set $X_{\mathrm{start}(i)} = \mathbb{N}+i$, and, for each $j \in \{\mathrm{start}(i)+1,...,\mathrm{start}(i+1)-1\}$, set $X_j = \{i\}$. Go to stage 1.
- Stage 1. In a dovetailing manner, monitor and act according to the following conditions.
  - Cond. [in the construction of $\mathcal{L}_0$ above, a k is found for i]. Set $X_{\mathrm{start}(i)+2} = \{i,...,k\}$ and $X_{\mathrm{start}(i)+3} = \{i,...,k+1\}$.
  - Cond. [$i \in X_j$, where $j < \mathrm{start}(i)$]. Set $X_{\mathrm{start}(i)+j+4} = X_j$.

Fig. 3. (a) The construction of $\mathcal{L}_0$ in the proof of Theorem 7. **(b)** The construction of $(X_j)_{j\in\mathbb{N}}$ in the proof of Theorem 7. The function start is defined in (10).

The proof of Theorem 7 below exhibits a class of languages $\mathcal{L}_0$ that is uniformly decidable and computably finitely thick, but $\mathcal{L}_0 \notin \mathbf{It}$. Thus, one could not arrive at the conclusion of Theorem 6 if one were to merely require that: there exists a uniformly decidable effective numbering of $\mathcal{L}$, and a possibly *distinct* computably finitely thick effective numbering of $\mathcal{L}$.

Theorem 7. There exists a class of languages $\mathcal{L}_0$ that is uniformly decidable and computably finitely thick, but $\mathcal{L}_0 \notin \mathbf{It}$.

Proof (Sketch). Let $((M,p)_i)_{i\in\mathbb{N}}$ be an algorithmic enumeration of all pairs of type $\mathcal{PC}_{1,1} \times \mathbb{N}$. Let start : $\mathbb{N} \to \mathbb{N}$ be such that, for each i,

$$\mathrm{start}(i) = 2^{i+1} - 4. \tag{10}$$

Note that, for each i, $\mathrm{start}(i+1) - \mathrm{start}(i) = \mathrm{start}(i) + 4$. The class $\mathcal{L}_0$ is constructed in Figure 3(a). An effective numbering $(X_j)_{j\in\mathbb{N}}$, which is easily seen to be of $\mathcal{L}_0$, is constructed in Figure 3(b). Let $f : \mathbb{N} \to \mathbb{N}$ be such that, for each i,

$$D_{f(i)} = \{\mathrm{start}(i), ..., \mathrm{start}(i+1) - 1\}, \tag{11}$$

It can be shown that $(X_j)_{j\in\mathbb{N}}$ and f witness that $\mathcal{L}_0$ is computably finitely thick. It is straightforward to construct an effective numbering witnessing that $\mathcal{L}_0$ is uniformly decidable. Finally, it can be shown that $\mathcal{L}_0 \notin \mathbf{It}$. $\approx\Box$ (**Theorem 7**)

4 Friedberg, Injective, and Extensional Iterative Learning

This section examines the Friedberg, injective, and extensional iterative learning models (**FrIt**, **InjIt**, and **ExtIt**, respectively). In terms of the classes of languages learnable by these models and by **It**, they are clearly related as follows.

$$\mathbf{FrIt} \subseteq \mathbf{InjIt} \subseteq \mathbf{ExtIt} \subseteq \mathbf{It}. \tag{12}$$

In this section we establish that **InjIt** $\not\subseteq$ **FrIt** (Proposition 8), and that **ExtIt** $\not\subseteq$ **InjIt** (Theorem 9). The fact that **It** $\not\subseteq$ **ExtIt** is due to Jain (Theorem 10).

Proposition 8 just below establishes that **InjIt** $\not\subseteq$ **FrIt**.

Proposition 8. InjIt $\not\subseteq$ FrIt.

Proof. Recall that $\mathcal{F}in$ is the collection of all finite subsets of $\mathbb{N}$. Jain & Stephan observed that $\mathcal{F}in \notin$ **FrIt** [JS08, Remark 28]. However, it is easily seen that $\mathcal{F}in \in$ **InjIt**. □ (**Proposition 8**)

Theorem 9 just below establishes that **ExtIt** $\not\subseteq$ **InjIt**.

Theorem 9. ExtIt $\not\subseteq$ InjIt.

Proof (Sketch). Let $\mathcal{L}_1$ be as follows.

$$\begin{aligned}\mathcal{L}_1 = \big\{2\mathbb{N}\big\} \cup \ \big\{&\{0,2,...,2e\} \cup \{2e+1\} \cup 2W_e,\\ &\{0,2,...,2e\} \cup \{2e+1\} \cup 2X \quad \mid e \in \mathbb{N} \ \wedge \ \text{(a)-(c) below}\big\}.\\ &\text{(a) } (\forall e' \in X)[W_{e'} = X].\\ &\text{(b) If } W_e = X, \text{ then } W_e \text{ and } X \text{ are finite.}\\ &\text{(c) If } W_e \neq X, \text{ then } 2W_e \subseteq \{0,2,...,2e\} \text{ and } X \text{ is infinite.}\end{aligned}$$

It can be shown that $\mathcal{L}_1 \in$ **ExtIt**. To complete the sketch of the proof: by way of contradiction, suppose that $\mathcal{L}_1 \in$ **InjIt**, as witnessed by $(M, p, (X_j)_{j\in\mathbb{N}}) \in \mathit{Iter}$. Then, there exists a k_0 such that

$$(\forall e \geq k_0)[M_p^*(0 \cdot 2 \cdot \dots \cdot 2k_0 \cdot 2e){\downarrow} = M_p^*(0 \cdot 2 \cdot \dots \cdot 2k_0)]. \tag{13}$$

By Case's 1-1 Operator Recursion Theorem [Cas74, Cas94],[11] there exists a computably enumerable sequence of pairwise-distinct φ-programs $(e_i)_{i\in\mathbb{N}}$ such that $(\forall i)[e_i \geq k_0]$, and such that the behavior of $(e_i)_{i\in\mathbb{N}}$ is as in Figure 4. It can be shown that $\{W_{e_0}, W_{e_1}\} \subseteq \mathcal{L}_1$, but that $(M, p, (X_j)_{j\in\mathbb{N}})$ does *not* **InjIt**-identify at least one of W_{e_0} and W_{e_1} (a contradiction). $\approx$□ (**Theorem 9**)

As mentioned above, the fact that **It** $\not\subseteq$ **ExtIt** is due to Jain.

Theorem 10 (Jain [Jai10]). It $\not\subseteq$ ExtIt.

We conclude this section with the following remark.

[11] Intuitively, the 1-1 Operator Recursion Theorem allows one to construct a computably enumerable sequence of pairwise-distinct φ-programs $(e_i)_{i\in\mathbb{N}}$ such that each program e_i *knows* all programs in the sequence and its own index i.

- Stage 0. Search for an $m \geq 1$ such that

$$M_p^*\big((0 \cdot 2 \cdot \dots \cdot 2e_0) \cdot (2e_0+1)^{m+1}\big)\downarrow = M_p^*\big((0 \cdot 2 \cdot \dots \cdot 2e_0) \cdot (2e_0+1)^{m}\big).$$

  If such an m is found, then set $\sigma_1 = (0 \cdot 2 \cdot \dots \cdot 2e_0) \cdot (2e_0+1)^m$, and go to stage 1. If *no* such m is found, then search indefinitely.

- Stage 1. For larger and larger values of n, make it the case that $W_{e_1} = \dots = W_{e_n} = \{e_1, ..., e_n\}$. Simultaneously, search for an $i \in \{1, ..., n\}$ such that

$$M_p^*(\sigma_1 \cdot 2e_i)\downarrow \neq M_p^*(\sigma_1).$$

  If such i and n are found, then make it the case that $W_{e_0} = \{e_1, ..., e_n\}$, and terminate the construction. If *no* such i and n are found, then search indefinitely, while making it the case that $(\forall i \geq 1)[W_{e_i} = \{e_i \mid i \geq 1\}]$.

Fig. 4. The construction of $(e_i)_{i \in \mathbb{N}}$ in the proof of Theorem 9

Remark 11. The fact **It** $\not\subseteq$ **InjIt** (as opposed to **It** $\not\subseteq$ **ExtIt** or **ExtIt** $\not\subseteq$ **InjIt**) can be shown directly using either of the next two pre-existing results.

- There exists a class of languages that *can* be **It**-identified, but that can*not* be so identified *order-independently* (in the sense of [BB75, Ful90]).[12]
- There exists a class of languages that *can* be **It**-identified, but that can*not* be so identified *strongly non-U-shapedly* [CK10, Theorem 5.4] (see also [Bei84, Wie91, CM08]).

5 Iterative Learning by Enumeration Operator

This section examines the iterative learning by enumeration operator model (**EOIt**). Recall that **EOIt** is similar to **It**, except that the computational component of the learner is modeled as an enumeration operator, as opposed to a partial computable function. Our main results of this section are the following.

- Every computably finitely thick class of languages (see Definition 3) can be **EOIt**-identified (Theorem 12).
- **EOIt** $\not\subseteq$ **It** (Corollary 13).
- **FrIt** $\not\subseteq$ **EOIt** (Theorem 14).

An open problem that remains is whether **It** $\cap$ **EOIt** $\subseteq$ **ExtIt**, i.e., whether every class of languages that can be **It**-identified *and* **EOIt**-identified can be **ExtIt**-identified (Problem 15).

Recall that $\mathcal{F}in \in$ **InjIt** $-$ **FrIt** (Proposition 8). Further recall that the class $\mathcal{L}_1$ from the proof Theorem 9 satisfies: $\mathcal{L}_1 \in$ **ExtIt** $-$ **InjIt**. It is straightforward to show that $\{\mathcal{F}in, \mathcal{L}_1\} \subseteq$ **EOIt**.

Theorem 12 just below is our first main result of this section.

[12] An anonymous referee attributes this result to Liepe & Wiehagen.

Theorem 12. Suppose that $\mathcal{L}$ is computably finitely thick. Then, $\mathcal{L} \in \mathbf{EOIt}$.

Proof (Sketch). Suppose that $\mathcal{L}$ is computably finitely thick. Let $\psi : \mathcal{F}in \to \mathcal{CE}$ be such that, for each $A \in \mathcal{F}in - \{\emptyset\}$:

$$\psi(\emptyset) = \emptyset; \qquad \psi(A) = \bigcap\{L \in \mathcal{L} \mid A \subseteq L\}.$$

Let $\mathcal{M} : \mathcal{P}(\mathbb{N}) \times \mathbb{N}_{\#} \to \mathcal{P}(\mathbb{N})$ be such that, for each $X \subseteq \mathbb{N}$ and x:

$$\mathcal{M}(X, \#) = X; \qquad \mathcal{M}(X, x) = \bigcup\big\{\psi(A) \mid A \text{ is finite } \wedge\ A \subseteq X \cup \{x\}\big\}.$$

It can be shown that $(\mathcal{M}, \emptyset)$ **EOIt**-identifies $\mathcal{L}$. $\approx\square$ (**Theorem 12**)

Recall that the proof of Theorem 7 exhibited a computably finitely thick class of languages $\mathcal{L}_0 \notin \mathbf{It}$. By Theorem 12, $\mathcal{L}_0 \in \mathbf{EOIt}$. Thus, one has the following.

Corollary 13 (of Theorems 7 and 12). EOIt $\not\subseteq$ **It**.

The proof of Theorem 14 below exhibits a class $\mathcal{L}_2 \in \mathbf{FrIt} - \mathbf{EOIt}$.

Theorem 14. FrIt $\not\subseteq$ **EOIt**.

Proof (Sketch). It is straightforward to construct a Friedberg numbering $(X_j)_{j\in\mathbb{N}}$ satisfying:

$$(\forall j)[1 \leq |D_j| \leq 3 \ \Rightarrow\ X_j = D_j]. \tag{14}$$

Let $(Y_k)_{k\in\mathbb{N}}$ be any Friedberg numbering satisfying: $Y_0 = \emptyset$. Let $(Z_\ell)_{\ell\in\mathbb{N}}$ be such that, for each j and k, $Z_{\langle j,k\rangle} = (2X_j) \cup (2Y_k + 1)$. It is straightforward to show that $(Z_\ell)_{\ell\in\mathbb{N}}$ is a Friedberg numbering. Let $\mathcal{L}_2$ be the following class of languages.

$$\mathcal{L}_2 = \{Z_{\langle j, \max D_j\rangle} \mid 1 \leq |D_j| \leq 3\}. \tag{15}$$

Note that, for each j such that $1 \leq |D_j| \leq 3$,

$$Z_{\langle j,\max D_j\rangle} \ = \ (2X_j) \cup (2Y_{\max D_j} + 1) \ = \ (2D_j) \cup (2Y_{\max D_j} + 1). \tag{16}$$

It is straightforward to show that $\mathcal{L}_2 \in \mathbf{FrIt}$ (e.g., using $(Z_\ell)_{\ell\in\mathbb{N}}$ as the hypothesis space). On the other hand, it can be shown that $\mathcal{L}_2 \notin \mathbf{EOIt}$. $\approx\square$ (**Theorem 14**)

As mentioned above, the following problem remains open.

Problem 15. Is it the case that $\mathbf{It} \cap \mathbf{EOIt} \subseteq \mathbf{ExtIt}$?

References

[Ang80] Angluin, D.: Finding patterns common to a set of strings. J. Comput. Syst. Sci. 21(1), 46–62 (1980)

[BB75] Blum, L., Blum, M.: Toward a mathematical theory of inductive inference. Inform. Control 28(2), 125–155 (1975)

[BBCJS10] Becerra-Bonache, L., Case, J., Jain, S., Stephan, F.: Iterative learning of simple external contextual languages. Theor. Comput. Sci. 411(29-30), 2741–2756 (2010)

[Bei84] Beick, H.-R.: Induktive Inferenz mit höchster Konvergenzgeschwindigkeit. PhD thesis, Sektion Mathematik, Humboldt-Universität Berlin (1984)

[Cas74] Case, J.: Periodicity in generations of automata. Math. Syst. Theory 8(1), 15–32 (1974)

[Cas94] Case, J.: Infinitary self-reference in learning theory. J. Exp. Theor. Artif. In. 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)

[CK10] Case, J., Kötzing, T.: Strongly non-U-shaped learning results by general techniques. In: Proc. of COLT 2010, pp. 181–193 (2010)

[CM08] Case, J., Moelius, S.: Optimal language learning. In: Freund, Y., Györfi, L., Turán, G., Zeugmann, T. (eds.) ALT 2008. LNCS (LNAI), vol. 5254, pp. 419–433. Springer, Heidelberg (2008)

[dBY10] de Brecht, M., Yamamoto, A.: Topological properties of concept spaces (full version). Inform. Comput. 208(4), 327–340 (2010)

[FKW82] Freivalds, R., Kinber, E., Wiehagen, R.: Inductive inference and computable one-one numberings. Z. Math. Logik 28(27), 463–479 (1982)

[Fri58] Friedberg, R.: Three theorems on recursive enumeration. I. Decomposition. II. Maximal set. III. Enumeration without duplication. J. Symbolic Logic 23(3), 309–316 (1958)

[Ful90] Fulk, M.: Prudence and other conditions on formal language learning. Inform. Comput. 85(1), 1–11 (1990)

[Gol67] Mark Gold, E.: Language identification in the limit. Inform. Control 10(5), 447–474 (1967)

[Jai10] Jain, S.: Private communcation (2010)

[JLMZ10] Jain, S., Lange, S., Moelius, S., Zilles, S.: Incremental learning with temporary memory. Theor. Comput. Sci. 411(29-30), 2757–2772 (2010)

[JS08] Jain, S., Stephan, F.: Learning in Friedberg numberings. Inform. Comput. 206(6), 776–790 (2008)

[Kum90] Kummer, M.: An easy priority-free proof of a theorem of Friedberg. Theor. Comput. Sci. 74(2), 249–251 (1990)

[LW91] Lange, S., Wiehagen, R.: Polynomial time inference of arbitrary pattern languages. New Generat. Comput. 8(4), 361–370 (1991)

[LZ96] Lange, S., Zeugmann, T.: Incremental learning from positive data. J. Comput. Syst. Sci. 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)

[MZ10] Moelius, S., Zilles, S.: Learning without coding (2010),(unpublished manuscript), `http://www2.cs.uregina.ca/~zilles/moeliusZ10TR.pdf`

[Rog67] Rogers, H.: Theory of Recursive Functions and Effective Computability. McGraw Hill, New York (1967); Reprinted, MIT Press (1987)

[SSS$^+$94] Shimozono, S., Shinohara, A., Shinohara, T., Miyano, S., Kuhara, S., Arikawa, S.: Knowledge acquisition from amino acid sequences by machine learning system BONSAI. Trans. Inform. Process. Soc. Jpn. 35(10), 2009–2018 (1994)

[Wie76] Wiehagen, R.: Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. J. Inform. Process. Cybern. (EIK) 12(1/2), 93–99 (1976)

[Wie91] Wiehagen, R.: A thesis in inductive inference. In: Dix, J., Schmitt, P.H., Jantke, K.P. (eds.) NIL 1990. LNCS (LNAI), vol. 543, pp. 184–207. Springer, Heidelberg (1991)

Learning Figures with the Hausdorff Metric by Fractals

Mahito Sugiyama[1,2], Eiju Hirowatari[3], Hideki Tsuiki[4], and Akihiro Yamamoto[1]

[1] Graduate School of Informatics, Kyoto University
Yoshida Honmachi, Sakyo-ku, Kyoto, 606-8501, Japan
mahito@iip.ist.i.kyoto-u.ac.jp,
akihiro@i.kyoto-u.ac.jp

[2] Research Fellow of the Japan Society for the Promotion of Science

[3] Center for Fundamental Education, The University of Kitakyushu

[4] Graduate School of Human and Environmental Studies, Kyoto University

Abstract. *Discretization* is a fundamental process for machine learning from analog data such as continuous signals. For example, the discrete Fourier analysis is one of the most essential signal processing methods for learning or recognition from continuous signals. However, only the direction of the time axis is discretized in the method, meaning that each datum is not purely discretized. To give a completely computational theoretical basis for machine learning from analog data, we construct a learning framework based on the Gold-style learning model. Using a modern mathematical computability theory in the field of Computable Analysis, we show that scalable sampling of analog data can be formulated as *effective* Gold-style learning. On the other hand, recursive algorithms are a key expression for models or rules explaining analog data. For example, FFT (Fast Fourier Transformation) is a fundamental recursive algorithm for discrete Fourier analysis. In this paper we adopt *fractals*, since they are general geometric concepts of recursive algorithms, and set learning objects as nonempty compact sets in the Euclidean space, called *figures*, in order to introduce fractals into Gold-style learning model, where the Hausdorff metric can be used to measure generalization errors. We analyze learnable classes of figures from informants (positive and negative examples) and from texts (positive examples), and reveal the hierarchy of learnabilities under various learning criteria. Furthermore, we measure the number of positive examples, one of complexities of learning, by using the Hausdorff dimension, which is the central concept of Fractal Geometry, and the VC dimension, which is used to measure the complexity of classes of hypotheses in the Valiant-style learning model. This work provides theoretical support for machine learning from analog data.

1 Introduction

The aim of this paper is giving a theoretical foundation for machine learning integrated with *discretization* of analog data. We construct a learning framework based on the Gold-style learning model [10], which is a traditional and basic

© Springer-Verlag Berlin Heidelberg 2010

$$\begin{tikzcd}[column sep=small]
 & \textsc{FigEx-Inf} = \textsc{FigCons-Inf} = \textsc{FigRelEx-Inf} \arrow[dl, dash] \arrow[dr, dash] & \\
\textsc{FigEfEx-Inf} \arrow[d, dash] & & \begin{array}{l}\textsc{FigEx-Txt} = \textsc{FigCons-Txt} \\ = \textsc{FigEx}^{\varepsilon}\textsc{-Inf} = \textsc{FigEx}^{\varepsilon}\textsc{-Txt}\end{array} \arrow[dll, dash] \arrow[d, dash] \\
\textsc{FigRefEx-Inf} \arrow[dr, dash] & & \textsc{FigRelEx-Txt} \arrow[dl, dash] \\
 & \textsc{FigRefEx-Txt} \arrow[d, dash] & \\
 & \textsc{FigEfEx-Txt} = \varnothing &
\end{tikzcd}$$

Fig. 1. The hierarchy of learnabilities. In each line, the lower set is a proper subset of the upper set.

model of computational learning theory [16]. Learners use *fractals* to represent models that explain given discretized analog data. We set learning targets as nonempty compact sets in the Euclidean space, called *figures*, in order to introduce the notion of fractals into the Gold-style learning model. We construct the hierarchy of learnabilities under various criteria, summarized in Fig. 1.

1.1 Background and Problems

Discretization is a fundamental process in machine learning from analog data. For example, Fourier analysis is one of the most essential signal processing methods, and the discrete version of the method, the discrete Fourier analysis, is used for learning or recognition on a computer from continuous signals. However, in the method, only the direction of the time axis is discretized, hence each datum is not purely discretized. This means that the gap between analog and digital data still remains.

In contrast, computational learning of languages or recursive functions from discrete data has been studied in detail based on the Gold-style learning model. However, computational learning of continuous objects, such as real-valued functions, from analog data is difficult and still under development [11, 12], since every datum has an intrinsic numerical error when it discretized.

1.2 Solutions

Mathematically, discretization is realized as a *partition* of rational intervals, which is a general model of sampling of continuous signals. We generate *positive examples* and *negative examples* of a target by partitioning intervals recursively, where a positive example is a rational interval intersecting the target, and a negative example is the one that does not intersect the target.

Recursive algorithms are key discrete expressions for models or rules explaining discretized analog data. For example, FFT (Fast Fourier Transformation) is a fundamental recursive algorithm for the discrete Fourier analysis. By generalizing such algorithms, we can extract the concept of fractals, which is an essential geometric concept of such algorithms. Moreover, lots of natural objects are known to be in the form of fractals [2]. Thus setting fractals as a basis of representation is one of straightforward ways in the learning from analog data. Here, the notion "learning of figures" is derived from abstraction and generalization of the concept of the discrete Fourier analysis, where fractals, which learners use to represent and compute figures, correspond to FFT intuitively.

In this paper, we use only *self-similar sets*, which are in a major family of fractals, since it is known that we can approximate every figure by some self-similar set arbitrarily closely, and can compute by a simple recursive algorithm (sometimes such an algorithm is called an Iterated Function System) [8].

In the process of sampling from analog data in the discrete Fourier analysis, scalability is a desirable property, meaning that when sample resolution increases, the accuracy of the result is refined. We formulate this property as *effective* learning of figures, which is inspired by effective computing in the framework of Type-2 Theory of Effectivity (TTE) in Computable Analysis [23]. This model guarantees that while a computer reads more and more precise information of the input, it produces more and more accurate approximations of the result. Here we interpret this model as effective learning from an infinite sequence of rational intervals to an infinite sequence of self-similar sets.

For effective learning of figures, we introduce the concept of *generalization errors* to evaluate "goodness" of each hypothesis. We use the Hausdorff metric to measure generalization errors, since the metric induces the standard topology on the set of figures [3].

1.3 Related Works

Lots of statistical methods are used in machine learning of continuous objects such as real-valued functions. The statistical approach fits to data mining and knowledge discovery from experimental data, and now it is achieving a great success in the fields [4]. However, they do not pay attention to treat continuous data themselves by discretization. For example, multi-layer perceptrons are used to learn real-valued functions, since they can approximate every continuous function arbitrarily closely. However, a perceptron is based on the idea of regulating analog wiring [20], hence such learning is not purely computable; *i.e.*, it ignores the gap between analog raw data and digital discretized data.

1.4 Organization of This Paper

The rest of the paper is organized as follows: Section 2 gives some notation about analysis and computability theory, and an overview of fractals, dimensions, and self-similar sets. We prepare methods for learning figures, partitions and self-similar programs, in Section 3, and formulate learning of figures in Section 4 (some methods have been introduced in the previous paper [22]). Section 5 gives learning of self-similar sets, and Section 6 gives learning of figures. The number of positive examples is measured with the Hausdorff and VC dimensions in Section 7. Section 8 gives conclusion.

2 Preliminaries

2.1 Notation

We assume that readers are familiar with the basic concepts of analysis and the ordinary computability theory [7, 13]. In the following let $\mathbb{N}$ be the set of natural numbers including 0, $\mathbb{Q}$ the set of rational numbers, and $\mathbb{R}$ the set of real numbers. The set $\mathbb{N}^+$ (resp. $\mathbb{R}^+$) is the set of positive natural (resp. real) numbers. The n product of $\mathbb{R}$ is denoted by $\mathbb{R}^n$. The set of nonempty compact sets of $\mathbb{R}^n$ is denoted by $\mathcal{K}^*$. The set of finite sequences over an alphabet Σ is denoted by Σ^*, and the *length* of a string w is denoted by $|w|$. The *empty string* λ is the string whose length is 0.

A *diameter* of a nonempty subset X of $\mathbb{R}^n$ is defined by $|X| := \sup\{d_{\mathrm{E}}(x, y) \mid x, y \in X\}$ for all $x, y \in \mathbb{R}^n$ with $x = (x_1, \dots, x_n)$ and $y = (y_1, \dots, y_n)$, where d_{E} is the *Euclidean metric* defined by $d_{\mathrm{E}}(x, y) := \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$. A set X is *countable* if there is a bijection from $\mathbb{N}$ to X. The cardinality of X is denoted by $\|X\|$. A set $\mathcal{U}$ is a *cover* of $X \subseteq \mathbb{R}^n$ if $\mathcal{U}$ is countable and $X \subseteq \bigcup_{U \in \mathcal{U}} U$, and $\mathcal{U}$ is a δ-*cover* of X if $\mathcal{U}$ is a cover of X and $|U| \leqslant \delta$ for all $U \in \mathcal{U}$.

2.2 Fractals and Dimensions

A *fractal* usually means a set with the following properties: It has a fine structure; it is too irregular to be described in the traditional geometrical language; it has some form of self-similarity; it is defined in a simple recursive manner [8].

For $X \subseteq \mathbb{R}^n$ and $s \in \mathbb{R}$ with $s > 0$, define $\mathcal{H}^s_\delta(X) := \inf\{\sum_{U \in \mathcal{U}} |U|^s \mid \mathcal{U}$ is a δ-cover of $X\}$. The *s-dimensional Hausdorff measure* of X is $\lim_{\delta \to 0} \mathcal{H}^s_\delta(X)$, denoted by $\mathcal{H}^s(X)$. When we fix a set X, a graph of $\mathcal{H}^s(X)$ with respect to s shows that there is at most one critical value at which $\mathcal{H}^s(X)$ jumps from ∞ to 0 [9]. This value is called the *Hausdorff dimension* of X. Formally, the Hausdorff dimension of a set X, written by $\dim_{\mathrm{H}} X$, is defined by $\dim_{\mathrm{H}} X := \sup\{\, s \mid \mathcal{H}^s(X) = \infty \,\} = \inf\{\, s \geqslant 0 \mid \mathcal{H}^s(X) = 0 \,\}$.

We have $\dim_{\mathrm{T}} X \leqslant \dim_{\mathrm{H}} X$ for all $X \subseteq \mathbb{R}^n$, and the term "fine structure" in the above properties of a fractal X usually means $\dim_{\mathrm{T}} X < \dim_{\mathrm{H}} X$. Here, $\dim_{\mathrm{T}} X$ denotes the *topological dimension* of X (see literature [14] for detail).

2.3 Self-similar Sets

A self-similar set is a fractal that is defined as the fixed point of a finite set of contractions [8].

For a nonempty compact set K, *δ-neighborhood* of K is defined by $K_\delta := \{ x \in \mathbb{R}^n \mid d_{\mathrm{E}}(x, a) \leqslant \delta \text{ for some } a \in K \}$. The *Hausdorff metric* d_{H} is defined by $d_{\mathrm{H}}(K, L) := \inf \{ \delta \mid K \subseteq L_\delta \text{ and } L \subseteq K_\delta \}$ for every pair of nonempty compact sets K and L. It is known that the metric space $(\mathcal{K}^*, d_{\mathrm{H}})$ is complete.

Let (X, d) be a metric space. A mapping $\varphi : X \to X$ is a *contraction* if there is a real number c with $0 < c < 1$ such that $d(\varphi(x), \varphi(y)) \leqslant cd(x, y)$ for all $x, y \in X$. The infimum of such a real number c is called *contractivity factor* and denoted by $L(\varphi)$. If $d(\varphi(x), \varphi(y)) = cd(x, y)$ for all $x, y \in X$, then the contraction φ is called *contracting similarity.*

Let C be a finite set of contractions on $(\mathbb{R}^n, d_{\mathrm{E}})$. A *self-similar set* of C is a nonempty compact set F such that $F = \bigcup_{\varphi \in C} \varphi(F)$. Moreover, if we define a mapping $\Phi : \mathcal{K}^* \to \mathcal{K}^*$ by $\Phi(K) := \bigcup_{\varphi \in C} \varphi(K)$ for all $K \in \mathcal{K}^*$, then Φ is a contraction on $(\mathcal{K}^*, d_{\mathrm{H}})$ with the contractivity factor $L(\Phi) = \max\{L(\varphi) \mid \varphi \in C\}$, and $F = \mathrm{Fix}(\Phi)$, where $\mathrm{Fix}(\Phi)$ is the fixed point of Φ. It is unique since the space $(\mathcal{K}^*, d_{\mathrm{H}})$ is complete. We say that $L(\Phi)$ is the contractivity factor of C. In addition, if we define $\Phi^0(K) := K$ and $\Phi^{k+1}(K) := \Phi(\Phi^k(K))$ for each $k \in \mathbb{N}$ recursively, then $F = \bigcap_{k=0}^{\infty} \Phi^k(K)$ for every set $K \in \mathcal{K}^*$ such that $\varphi(K) \subset K$ for every $\varphi \in C$. This means that we have a level-wise constructing method with Φ to obtain the self-similar set F.

Example 1. Let contractions $\varphi_1, \varphi_2, \varphi_3 : \mathbb{R}^2 \to \mathbb{R}^2$ be as follows:

$$\varphi_1\begin{bmatrix}x_1\\x_2\end{bmatrix} = \frac{1}{2}\begin{bmatrix}x_1\\x_2\end{bmatrix} + \begin{bmatrix}0\\0\end{bmatrix},\ \varphi_2\begin{bmatrix}x_1\\x_2\end{bmatrix} = \frac{1}{2}\begin{bmatrix}x_1\\x_2\end{bmatrix} + \begin{bmatrix}0\\1/2\end{bmatrix},\ \varphi_3\begin{bmatrix}x_1\\x_2\end{bmatrix} = \frac{1}{2}\begin{bmatrix}x_1\\x_2\end{bmatrix} + \begin{bmatrix}1/2\\1/2\end{bmatrix}.$$

A self-similar set of the set $\{ \varphi_1, \varphi_2, \varphi_3 \}$ is called the *Sierpiński triangle.*

We can bound the Hausdorff distance between a compact set and a self-similar set using the following theorem known as Collage Theorem [2].

Theorem 1. *Let C be a finite set of contractions and $K \in \mathcal{K}^*$. We have $d_{\mathrm{H}}(K, F) \leqslant d_{\mathrm{H}}(K, \bigcup_{\varphi \in C} \varphi(K))/(1 - c)$, where F is the self-similar set of C and c is the contractivity factor of C.*

This theorem shows that every compact set can be approximated arbitrarily closely by a self-similar set in the sense of the Hausdorff metric.

Corollary 1 (Falconer [8]). *Let $K \in \mathcal{K}^*$. Given $\delta > 0$, then there exists a finite set of contractions with its self-similar set F satisfying $d_{\mathrm{H}}(K, F) < \delta$.*

It is usually difficult to know the Hausdorff dimension of a given set. However, we can obtain the dimension of a certain class of self-similar sets in the following manner. Let C be a finite set of contractions, and F be the self-similar set of C. The *similarity dimension* of F, denoted by $\dim_{\mathrm{S}} F$, is defined by the equation $\sum_{\varphi \in C} L(\varphi)^{\dim_{\mathrm{S}} F} = 1$. We have $\dim_{\mathrm{H}} F \leqslant \dim_{\mathrm{S}} F$, and if C satisfies the open set condition, $\dim_{\mathrm{H}} F = \dim_{\mathrm{S}} F$ [8]. Here, a finite set of contractions C satisfies the *open set condition* if there exists a nonempty bounded open set $O \subset \mathbb{R}^n$ such that $\varphi(O) \subset O$ for all $\varphi \in C$ and $\varphi(O) \cap \varphi'(O) = \emptyset$ for all $\varphi, \varphi' \in C$ with $\varphi \neq \varphi'$.

3 Methods for Learning

3.1 Partitions

We realize discretization by partition of rational closed intervals (closed intervals whose end points are rational numbers). First of all, we define a partition for the *unit interval*, which is the closed interval $[0,1] \times [0,1] \times \cdots \times [0,1]$ on $\mathbb{R}^n$, denoted by I_{U}. We denote the set of rational closed intervals by $\mathcal{CI}$, and the interior of X by $\operatorname{int} X$.

Definition 1. Let Σ be the set $\{1, 2, \ldots, k\}$ for some $k \in \mathbb{N}$. A mapping ψ from Σ to $\mathcal{CI}$ is a *partition* if $\bigcup_{i \in \Sigma} \psi(i) = I_{\mathrm{U}}$. A partition ψ is a *net* if for all $I, J \in \operatorname{range}(\psi)$ with $I = I_1 \times \cdots \times I_n$ and $J = J_1 \times \cdots \times J_n$, $\operatorname{int} I \cap \operatorname{int} J = \emptyset$ and $|I_i| = |J_j|$ for all $i, j \in \{1, \ldots, n\}$.

We identify an alphabet Σ with the domain of a partition $\operatorname{dom}(\psi)$. If ψ is a net, there exists $b \in \mathbb{N}$ such that $\|\Sigma\| = b^n$ and $|I| = |I_{\mathrm{U}}|/b$ for all $I \in \operatorname{range}(\psi)$.

We define a way for dividing arbitrary intervals using a partition as a template. Let a pair of intervals $I = I_1 \times \cdots \times I_n$ and $J = J_1 \times \cdots \times J_n$. Define $\varphi_{I \to J}(x) := [a_{ij}]_{n \times n} x + [b_{ij}]_{n \times 1}$ for all $x \in \mathbb{R}^n$, where $[a_{ij}]_{n \times n}$ is a $n \times n$ diagonal matrix and $[b_{ij}]_{n \times 1}$ is a $n \times 1$ matrix, such that $a_{ii} = |J_i| \,/\, |I_i|$ and $b_{i1} = \min J_i - a_{ii} \min I_i$ for all $i \in \{1, \ldots, n\}$.

Here we define $\mathcal{S}_{\psi,I} := \{\varphi_{I_{\mathrm{U}} \to I}(\psi(a)) \mid a \in \Sigma\}$ for all $I \in \mathcal{CI}$. It is a set of intervals obtained by partitioning I using the template partition ψ. By partitioning a fixed interval recursively, we can obtain a unique set of intervals.

Definition 2. Define $\mathcal{S}^0_{\psi,I} := \{\, I \,\}$ and $\mathcal{S}^{k+1}_{\psi,I} := \bigcup_{J \in \mathcal{S}^k_{\psi,I}} \mathcal{S}_{\psi,J}$ for all $k \in \mathbb{N}$ recursively, and define $\mathcal{S}^*_{\psi,I} := \bigcup_{k \in \mathbb{N}} \mathcal{S}^k_{\psi,I}$. We say that $\mathcal{S}^*_{\psi,I}$ is the *partitioning space* from I, $\mathcal{S}^k_{\psi,I}$ is the *level k partitioning space* from I, and I is an *initial interval* of $\mathcal{S}^*_{\psi,I}$.

Example 2. Let $n = 1$, $\Sigma = \{0, 1\}$, $\psi(0) = [0, 1/2]$, and $\psi(1) = [1/2, 1]$. Then $\mathcal{S}^0_{\psi,I_{\mathrm{U}}} = \{[0,1]\}, \mathcal{S}^1_{\psi,I_{\mathrm{U}}} = \{[0,1/2],[1/2,1]\}, \ldots$. This corresponds to the standard binary representation of real numbers.

Definition 3. Let $I \in \mathcal{CI}$ be an initial interval and $\psi : \Sigma \to \mathcal{CI}$ be a partition. We define a *representation* $\rho_{\psi,I} : \Sigma^* \to \mathcal{S}^*_{\psi,I}$ by $\rho_{\psi,I}(\lambda) := I$ and $\rho_{\psi,I}(wa) := \varphi_{I_{\mathrm{U}} \to \rho_{\psi,I}(w)}(\psi(a))$, where $w \in \Sigma^*$ and $a \in \Sigma$.

Example 3. Let $n = 1$ and ψ be the partition defined in Example 2. Then $\rho_{\psi,I}(010) = [1/4, 3/8]$ if $I = [0,1]$, and $\rho_{\psi,J}(10) = [1/4, 3/8]$ if $J = [0, 1/2]$.

3.2 Self-similar Programs

We introduce a class of logic programs corresponding to a class of finite sets of contractions by using the predicate symbol $\mathrm{Path}(\cdot)$. A *logic program* is a finite set of definite clauses, which are sentences of the form $A \leftarrow A_1, \ldots, A_m$ $(m \geqslant 0)$, where all of A, A_1, $\ldots$, and A_m are atoms [18]. We assume that every term is a list, which is used in Prolog, and identify lists $[]$, $[a_1, \ldots, a_m]$, and $[a_1, \ldots, a_m \mid x]$ with λ, $a_1 \ldots a_m$, and $a_1 \ldots a_m x$, respectively, where x is a variable.

Definition 4. We define $\mathrm{SP}(W) := \{\mathrm{Path}(\lambda)\} \cup \{\mathrm{Path}(wx) \leftarrow \mathrm{Path}(x) \mid w \in W\}$ for every nonempty finite set $W \subset \Sigma^*$, where x is a variable. We say that $\mathrm{SP}(W)$ is a *self-similar program.*

The set of all self-similar programs is denoted by $\mathcal{P}$, and for $N \subseteq \mathbb{N}$, the set $\{\mathrm{SP}(W) \in \mathcal{P} \mid \|W\| \in N\}$ by $\mathcal{P}_N$. For simplicity, we denote $\mathcal{P}_{\{m\}}$ ($m \in \mathbb{N}$) by $\mathcal{P}_m$. Trivially, the set $\mathcal{P}$ is recursively enumerable.

Each self-similar program corresponds to a finite set of contractions, and its representation is defined as follows.

Definition 5. Let a self-similar program $P = \mathrm{SP}(W)$. We define $\kappa_{\psi,I}(P) := \bigcap_{k=0}^{\infty} \Gamma_{\psi,I}^{k}(P)$, where for all $k \in \mathbb{N}$, $\Gamma_{\psi,I}^{k}(P) := \bigcup\{\rho_{\psi,I}(w_0 \dots w_k) \mid w_i \in W \text{ for all } i \in \{0, \dots, k\}\}$.

We say that a program P represents the self-similar set $\kappa_{\psi,I}(P)$. We have shown that the process of building a self-similar set from a self-similar program is guaranteed to be unique and constructive, where we interpret self-similar programs as their least Herbrand models [22].

We can easily obtain the similarity dimension for every set represented by a self-similar program. We have $\dim_{\mathrm{S}} \kappa_{\psi,I}(\mathrm{SP}(W)) = d$, where $\sum_{\varphi \in C} L(\varphi)^d = 1$ with $C = \{\, \varphi_{I \to \rho_{\psi,I}(w)} \mid w \in W \,\}$. Moreover, if ψ is a net, such a set C always holds the open set condition. Thus $\dim_{\mathrm{H}} \kappa_{\psi,I}(P) = \dim_{\mathrm{S}} \kappa_{\psi,I}(P)$ for all $P \in \mathcal{P}$.

Example 4. Let ψ be a partition defined by $\Sigma := \{0, 1, 2, 3\}$, $\psi(0) := [0, 1/2] \times [0, 1/2]$, $\psi(1) := [0, 1/2] \times [1/2, 1]$, $\psi(2) := [1/2, 1] \times [0, 1/2]$, and $\psi(3) := [1/2, 1] \times [1/2, 1]$. Assume that $P = \mathrm{SP}(\{\, 0, 1, 3 \,\})$. Then ψ is a net and the set $\kappa_{\psi,I}(P)$ is the Sierpiński triangle shown in Example 1. Its Hausdorff dimension is equal to its similarity dimension, and $\dim_{\mathrm{H}} \kappa_{\psi,I}(P) = \log 3 / \log 2$.

4 Learning Framework

Using the above methods, we formulate learning of figures. In the following in this paper, we fix a partition ψ and an initial interval for the partitioning space, and assume that ψ is a net and every figure is a subset of the initial interval. We omit the indexes ψ and I; *e.g.*, write ρ and κ for $\rho_{\psi,I}$ and $\kappa_{\psi,I}$, respectively.

Define $\mathcal{Q}(K) := \{\, I \in \mathcal{S}^* \mid K \cap I \neq \emptyset \,\}$ for a figure $K \in \mathcal{K}^*$. Each element of $\mathcal{Q}(K)$ corresponds to a discretized positive datum of the figure K.

Definition 6. An *example* of a figure K is a signed finite sequence $\langle l, w \rangle$ with $l \in \{\, +, - \,\}$, where $\rho(w) \in \mathcal{Q}(K)$ if $l = +$, and $\rho(w) \notin \mathcal{Q}(K)$ if $l = -$.

An example $\langle l, w \rangle$ is *positive* if $l = +$, and *negative* if $l = -$. Here, if $\langle +, w \rangle$ is an example of K, then $\langle +, v \rangle$ is an example of K for every prefix v of w, and $\langle +, wa \rangle$ is an example of K for some $a \in \Sigma$. If $\langle -, w \rangle$ is an example of K, then $\langle -, wv \rangle$ is an example of K for all $v \in \Sigma^*$.

We say that an infinite sequence of examples of a figure K, denoted by σ_K, is a *presentation* of K. The set of all examples occurring in σ_K is denoted by $\mathrm{range}(\sigma_K)$, and the initial segment of σ_K of length m is denoted by $\sigma_K[m-1]$. A *text* is a presentation σ_K such that $\{\, \rho(w) \mid \langle +, w\rangle \in \mathrm{range}(\sigma_K)\,\} = \mathcal{Q}(K)$, and an *informant* is a presentation σ_K such that $\{\, \rho(w) \mid \langle +, w\rangle \in \mathrm{range}(\sigma_K)\,\} = \mathcal{Q}(K)$ and $\{\, \rho(w) \mid \langle -, w\rangle \in \mathrm{range}(\sigma_K)\,\} = \mathcal{S}^* \setminus \mathcal{Q}(K)$.

A *learner* is a procedure that reads a presentation of a target figure from time to time, and outputs self-similar programs from time to time. A *hypothesis* is a program produced by a learner. In the following, let $\mathbf{M}$ denote a learner, and $\mathbf{M}(\sigma_K)$ denote an infinite sequence of hypotheses produced by $\mathbf{M}$ on the input σ_K, thereby $\mathbf{M}(\sigma_K)(i-1)$ denotes the ith hypothesis produced by $\mathbf{M}$. We say that an infinite sequence of hypotheses $\mathbf{M}(\sigma_K)$ *converges* to a hypothesis P if there exists $m \in \mathbb{N}$ such that $\mathbf{M}(\sigma_K)(i) = P$ for all $i \geqslant m$.

5 Learning Self-similar Sets in the Limit

We consider the learning criterion corresponding to **Ex**-learning (learning in the limit), called **FigEx-Inf-** and **FigEx-Txt**-learning, and analyze the learnabilities. A *generalization error* is a distance between a target figure K and a hypothesis P, written by $\mathrm{GE}(K, P)$, and measured by the Hausdorff metric, that is, $\mathrm{GE}(K, P) := d_{\mathrm{H}}(K, \kappa(P))$. Trivially, $\mathrm{GE}(K, P) = 0$ if and only if $K = \kappa(P)$.

Definition 7. A learner **M FigEx-Inf**-*learns* (resp. **FigEx-Txt**-*learns*) a set of figures $\mathcal{F} \subseteq \mathcal{K}^*$ if for all $K \in \mathcal{F}$ and for every informant (resp. text) of K, $\mathbf{M}(\sigma_K)$ converges to a hypothesis P such that $\mathrm{GE}(K, P) = 0$.

For every learning criterion **CR** introduced in the following, we say that a set of figures $\mathcal{F}$ is **CR**-*learnable* if there is some learner that **CR**-learns $\mathcal{F}$, and denote **CR** by the collection of sets of figures that are **CR**-learnable.

Let us consider the learnability of the set $\kappa(\mathcal{P}) = \{\, \kappa(P) \mid P \in \mathcal{P}\,\}$, where every figure can be represented by some self-similar program.

Lemma 1. *For every $K, L \in \kappa(\mathcal{P})$, $K \neq L$ if and only if $\mathcal{Q}(K) \neq \mathcal{Q}(L)$.*

A figure $K \in \kappa(\mathcal{P})$ and the set of intervals $\mathcal{Q}(K)$ are therefore identifiable. Furthermore, we can easily check that $\mathcal{Q}(K) \subseteq \mathcal{Q}(L)$ if and only if $K \subseteq L$. Thus a figure corresponds to a *concept* in the Gold-style learning.

An infinite sequence $P_0, P_1, \ldots$ is a *normal enumeration* if $\{\, P_i \mid i \in \mathbb{N}\,\} = \mathcal{P}$ and for all $i, j \in \mathbb{N}$, $i < j$ implies $|P_i| \leqslant |P_j|$, where $|\mathrm{SP}(W)|$ denotes $\max_{w \in W} |w|$. Trivially, some procedure can enumerate $\mathcal{P}$ through a normal enumeration.

We say that a hypothesis P is *consistent* with an example $\langle l, w\rangle$ if $l = +$ implies $\rho(w) \in \mathcal{Q}(\kappa(P))$ and $l = -$ implies $\rho(w) \notin \mathcal{Q}(\kappa(P))$, and consistent with a set of examples E if P is consistent with all examples in E. Here we show that this notion of consistency is effective, since $\kappa(\mathcal{P})$ corresponds to the well known notion of *indexed family of recursive concepts* [1].

Lemma 2. *Let P be a self-similar program. For all $I \in \mathcal{S}^*$, $I \in \mathcal{Q}(\kappa(P))$ if and only if $I \cap J \neq \emptyset$ for some $J \in \Gamma^k(P)$ such that $|J| < |I|$.*

This means that there is an algorithm that can judge consistency of P w.r.t. $\langle l, w\rangle$, since it is enough to check intervals in $\Gamma^{|w|+1}(P)$. In the following in this paper, we assume that learners can judge the consistency of hypotheses.

First, we analyze **FigEx-Inf**-learning. A learner can **FigEx-Inf**-learn a set of figures if it has an ability to enumerate all hypotheses and judge whether or not each hypothesis is consistent with received examples [10].

Theorem 2. *The set $\kappa(\mathcal{P}) \in$ **FigEx-Inf**.*

Next, we consider **FigEx-Txt**-learning. The necessary and sufficient conditions to achieve learning of languages from texts have been studied in detail [1], and characterization with finite tell-tale sets is well known. We use these results.

Definition 8. Let $\mathcal{F}$ be a subset of $\kappa(\mathcal{P})$. For a figure $K \in \mathcal{F}$, a finite subset $\mathcal{T}$ of $\mathcal{Q}(K)$ is a *finite tell-tale set* of K with respect to $\mathcal{F}$ if for all $L \in \mathcal{F}$, $\mathcal{T} \subseteq \mathcal{Q}(L)$ implies $\mathcal{Q}(L) \not\subset \mathcal{Q}(K)$ (*i.e.*, $L \not\subset K$).

Theorem 3. *Let $\mathcal{F}$ be a subset of $\kappa(\mathcal{P})$. Then $\mathcal{F}$ is **FigEx-Txt**-learnable if and only if for every figure $K \in \mathcal{F}$, there is a procedure that enumerates a set $W \subset \Sigma^*$ such that $\rho(W)$ is a finite tell-tale set of K with respect to $\mathcal{F}$.*

Here we show that the set $\kappa(\mathcal{P})$ is not **FigEx-Txt**-learnable. However, if we fix the number of contractions, that is, fix the cardinality of a parameter W of a self-similar program $\mathrm{SP}(W)$ *a priori*, such a set becomes **FigEx-Txt**-learnable.

Theorem 4. *The set $\kappa(\mathcal{P}) \notin$ **FigEx-Txt**, and for every finite set $N \subset \mathbb{N}$, $\kappa(\mathcal{P}_N) \in$ **FigEx-Txt**.*

It is natural that every hypothesis generated by a learner is consistent with examples received by it so far. Here we introduce **FigCons-Inf**- and **FigCons-Txt**-learning (CONS means CONSistent) that correspond to **Cons**-learning studied in the learning of languages and recursive functions [16].

Definition 9. A learner $\mathbf{M}$ **FigCons-Inf**-*learns* (resp. **FigCons-Txt**-*learns*) a set of figures $\mathcal{F} \subseteq \mathcal{K}^*$ if $\mathbf{M}$ **FigEx-Inf**-learns (resp. **FigEx-Txt**-learns) $\mathcal{F}$, and for all $K \in \mathcal{F}$ and all informants (resp. texts) σ_K, $\mathbf{M}(\sigma_K)(i)$ is consistent with E_i for all $i \in \mathbb{N}$, where E_i is the set of examples that $\mathbf{M}$ received until just before generating the hypothesis $\mathbf{M}(\sigma_K)(i)$.

Corollary 2. **FigEx-Inf** $=$ **FigCons-Inf**, **FigEx-Txt** $=$ **FigCons-Txt**.

6 Towards Learning Figures out of $\kappa(\mathcal{P})$

Here we consider learning of figures that might not be represented by any hypothesis, since generally there is no guarantee of existence of a correct hypothesis that represents a target figure exactly. Moreover, the set of figures $\mathcal{K}^*$ has the cardinality of the continuum, thereby no learner is able to learn $\mathcal{K}^*$ in the limit. To deal with learning of an arbitrary figures, we have to introduce other learning criteria extending **FigEx-Inf**- and **FigEx-Txt**-learning.

In the learning of languages and recursive functions, two learning criteria with the concepts of *reliability* and *refutability* have been proposed and studied to deal with targets out of a hypothesis space [19, 21]. First, we introduce these concepts into learning of figures. Next, we investigate our original criterion, where a target is learned *effectively*. Finally, we introduce generalization error bounds into Gold-style learning model. We compare learnability under each criterion.

6.1 Reliable Learning

Let us consider *reliable* learning of figures. Intuitively, reliability requires that an infinite sequence of hypotheses converges to only a correct hypothesis.

Definition 10. A learner **M** **FigRelEx-Inf**-*learns* (resp. **FigRelEx-Txt**-*learns*) a set of figures $\mathcal{F} \subseteq \mathcal{K}^*$ if **M** satisfies the following conditions:

1. The learner **M** **FigEx-Inf**-learns (resp. **FigEx-Txt**-learns) $\mathcal{F}$.
2. If a target figure $K \in \mathcal{K}^* \setminus \mathcal{F}$, then for all informants (resp. texts) σ_K, the infinite sequence of hypotheses $\mathbf{M}(\sigma_K)$ does not converge to any hypothesis.

Let a target figure $K \in \mathcal{K}^* \setminus \mathcal{F}$. Intuitively, for any hypothesis P, if **M** can judge $\kappa(P) \neq K$ in finite time, then a set of figures $\mathcal{F}$ can be learned reliably.

Theorem 5. **FigEx-Inf** = **FigRelEx-Inf**.

Next, we analyze **FigRelEx-Txt**-learnability of the set $\kappa(\mathcal{P}_N)$. Note that for all $K \in \kappa(\mathcal{P}_N)$, $N = \{1\}$ implies that K is a singleton.

Theorem 6. *The set* $\kappa(\mathcal{P}_N)$ *is* **FigRelEx-Txt**-*learnable iff* $N = \{1\}$.

Corollary 3. **FigRelEx-Txt** $\subset$ **FigEx-Txt**.

It is known that a set of concepts $\mathcal{C}$ is reliably **Ex**-learnable from texts if and only if $\mathcal{C}$ contains no infinite concept [19]. However, we have shown that the set $\kappa(\mathcal{P}_1)$ is **FigRelEx-Txt**-learnable, where $\mathcal{Q}(K)$ is infinite for all $K \in \kappa(\mathcal{P}_1)$. The monotonicity of the set $\mathcal{Q}(K)$ causes this remarkable difference.

6.2 Refutable Learning

We extend **FigEx-Inf**- and **FigEx-Txt**-learning by paying our attention to *refutability*. In refutable learning, a learner tries to learn figures in the limit, and it refutes the given space if there is no correct hypothesis in the space.

Definition 11. A learner **M** **FigRefEx-Inf**-*learns* (resp. **FigRefEx-Txt**-*learns*) a set of figures $\mathcal{F} \subseteq \mathcal{K}^*$ if **M** satisfies the following conditions. Here, $\perp$ is the *refutation symbol*.

1. The learner **M** **FigEx-Inf**-learns (resp. **FigEx-Txt**-learns) $\mathcal{F}$.
2. If a target figure $K \in \mathcal{F}$, then for all informants (resp. texts) σ_K, $\mathbf{M}(\sigma_K)(i) \neq \perp$ for all $i \in \mathbb{N}$.
3. If $K \in \mathcal{K}^* \setminus \mathcal{F}$, then for all informants (resp. texts) σ_K, there exists $m \in \mathbb{N}$ such that $\mathbf{M}(\sigma_K)(i) \neq \perp$ for all $i < m$, and $\mathbf{M}(\sigma_K)(i) = \perp$ for all $i \geqslant m$.

Conditions 2 and 3 in the above definition means that a learner **M** refutes the set $\mathcal{F}$ in finite time only if a target figure $K \in \mathcal{K}^* \setminus \mathcal{F}$.

Lemma 3. *Let a learner* **M** **FigRefEx-Inf**-*learns* (*resp.* **FigRefEx-Txt**-*learns*) *a set of figures* $\mathcal{F}$, *and let a target figure* $K \in \mathcal{K}^* \setminus \mathcal{F}$. *For every informant* (*resp. text*) σ_K, *if* **M** *outputs* $\perp$ *after receiving* $\sigma_K[m]$, *then for any* $L \in \mathcal{F}$, *the set of examples* $\mathrm{range}(\sigma_K[m])$ *is not consistent with* L.

We show that the set $\kappa(\mathcal{P}_m)$ is not **FigRefEx-Inf**-learnable. Intuitively, the reason is that for some figure $K \in \mathcal{K}^* \setminus \kappa(\mathcal{P}_m)$, there exists an arbitrarily close figure $L \in \kappa(\mathcal{P}_m)$, and $\mathbf{M}$ cannot refute $\kappa(\mathcal{P}_m)$ in finite time.

Theorem 7. *For any $m \in \mathbb{N}$, $\kappa(\mathcal{P}_m) \notin$* **FigRefEx-Inf**.

We can easily find a set $\mathcal{F} \subset \kappa(\mathcal{P})$ such that $\mathcal{F} \in$ **FigRefEx-Inf** and $\mathcal{F} \notin$ **FigRelEx-Txt**. Thus both **FigRefEx-Inf** $\not\subseteq$ **FigRelEx-Txt** and **FigRelEx-Txt** $\not\subseteq$ **FigRefEx-Inf** hold.

Theorem 8. **FigRefEx-Txt** $\subset$ **FigRefEx-Inf**.

Corollary 4. **FigRefEx-Txt** $\subset$ **FigRelEx-Txt**.

6.3 Effective Learning

In reliable and refutable learning, we cannot know how far it is from the recent hypothesis to the correct hypothesis. It is therefore more useful if we can measure the amount of an error of each hypothesis.

Define diam(k) by the diameter of an interval in $\mathcal{S}^k$. If a hypothesis P is consistent with all level k examples, we can bound the Hausdorff distance between a target figure and the figure $\kappa(P)$ with diam(k).

Lemma 4. *Let σ_K be an informant of a figure K and P be a self-similar program that is consistent with the set $\{\langle l, w\rangle \in \mathrm{range}(\sigma_K) \mid |w| = k\}$. We have $d_{\mathrm{H}}(K, \kappa(P)) \leqslant \mathrm{diam}(k)$.*

We now define a novel effective learning criteria, **FigEfEx-Inf**- and **FigEf Ex-Txt**-learning (EF means EFfective). Intuitively, these criteria guarantee that for any target figure, a generalization error becomes smaller and smaller monotonically, and converges to zero. Thus we can know when the learner learns the target figure "well enough". Furthermore, if a target figure is learnable in the limit, then a generalization error goes to zero in finite time.

Definition 12. A learner $\mathbf{M}$ **FigEfEx-Inf**-*learns* (resp. **FigEfEx-Txt**-*learns*) a set of figures $\mathcal{F} \subseteq \mathcal{K}^*$ if $\mathbf{M}$ satisfies the following conditions:

1. The learner $\mathbf{M}$ **FigEx-Inf**-learns (resp. **FigEx-Txt**-learns) $\mathcal{F}$.
2. For an arbitrary target figure $K \in \mathcal{K}^*$ and all informants (resp. texts) σ_K, there is a monotone decreasing function $\varepsilon : \mathbb{N} \to \mathbb{R}^+$ such that $\lim_{i\to\infty} \varepsilon(i) = 0$ and $\mathrm{GE}(K, \mathbf{M}(\sigma_K)(i)) \leqslant \varepsilon(i)$ for all $i \in \mathbb{N}$.

This effective learning is different from well-known learning criterion of **Bc**-learning [16], since **Bc**-learning only guarantees that generalization errors go to zero in finite time. Thus **Bc**-learning is not effective.

Theorem 9. *The set $\kappa(\mathcal{P}) \in$* **FigEfEx-Inf** (see Fig. 2).

However, the set $\kappa(\mathcal{P}_N)$ with finite $N \subset \mathbb{N}$ is not **FigEfEx-Inf**-learnable. Informally, the reason is that $\kappa(\mathcal{P}_N)$ does not have enough ability to approximate an arbitrary figure.

```
Input: an informant $\sigma_K = \langle l_0, w_0 \rangle, \langle l_1, w_1 \rangle, \dots$
Output: an infinite sequence of hypotheses $\mathbf{M}(\sigma_K)(0), \mathbf{M}(\sigma_K)(1), \dots$

$i := 0$ ;   $k := 0$ ;   $E := \emptyset$ ;   /* $E$ is a set of received examples */
repeat
  read $\sigma_K(i)$ and add to $E$ ;   /* $\sigma_K(i) = \langle l_i, w_i \rangle$ */
  if $\{\langle l, w \rangle \in \mathrm{range}(\sigma_K) \mid |w| = k\} \subseteq E$ then
    search the first $P$ that is consistent with $E$ through a normal enumeration ;
    output $P$ ;   /* $\mathbf{M}(\sigma_K)(i) = P$ */
    $k := k + 1$ ;
  $i := i + 1$ ;
until forever
```

Fig. 2. A learning procedure that **FigEfEx-Inf**-learns $\kappa(\mathcal{P})$

Theorem 10. *For any finite set* $N \subset \mathbb{N}$, $\kappa(\mathcal{P}_N) \notin$ **FigEfEx-Inf**.

We analyze the hierarchy of learnabilities under reliable, refutable, and effective learning. From the above results, **FigEfEx-Inf** $\not\subseteq$ **FigEx-Txt** and **FigEx-Txt** $\not\subseteq$ **FigEfEx-Inf** hold. In the learning from informants, if $\mathcal{F} \in$ **FigRefEx-Inf**, then $\mathcal{F} \in$ **FigEfEx-Inf**. Thus the following holds.

Corollary 5. **FigRefEx-Inf** $\subset$ **FigEfEx-Inf** $\subset$ **FigRelEx-Inf**.

Theorem 11. **FigEfEx-Txt** $= \emptyset$.

6.4 Learning with Generalization Error Bounds

In the effective learning, we used generalization errors to measure the difference between a hypothesis and a target figure. However, achieving the learning is difficult, since we have proved that $\kappa(\mathcal{P}_N)$ is not **FigEfEx-Inf**-learnable and any set of figures is not **FigEfEx-Txt**-learnable. Here we propose an another new learning criterion with generalization error bounds, and show the interesting result: The learnabilities from informants and from texts become same.

Definition 13. For $\varepsilon \in \mathbb{R}^+$, a learner **M** **FigEx**$^\varepsilon$**-Inf**-*learns* (resp. **FigEx**$^\varepsilon$-**Txt**-*learns*) a set of figures $\mathcal{F} \subseteq \mathcal{K}^*$ if for all target figures K and all informants (resp. texts) σ_K, $K \in \mathcal{F}$ implies that $\mathbf{M}(\sigma_K)$ converges to a hypothesis P such that $\mathrm{GE}(K, P) = 0$, and $K \in \mathcal{K}^* \setminus \mathcal{F}$ implies that $\mathbf{M}(\sigma_K)$ converges to a hypothesis P such that $\mathrm{GE}(K, P) \leqslant \varepsilon$.

First, we compare **FigEx**$^\varepsilon$**-Inf**- and **FigEfEx-Inf**-learning.

Theorem 12. *For any* $\varepsilon \in \mathbb{R}^+$, $\kappa(\mathcal{P}) \notin$ **FigEx**$^\varepsilon$**-Inf**.

Intuitively, the reason is that **FigEx**$^\varepsilon$**-Inf**-learning demands convergence of hypotheses for every target figure, but **FigEfEx-Inf**-learning does not. Here we show that $\mathcal{F}$ is **FigEx-Txt**-learnable iff it is **FigEx**$^\varepsilon$**-Txt**-learnable.

Theorem 13. *For all* $\varepsilon \in \mathbb{R}^+$*,* $\textbf{FigEx-Txt} = \textbf{FigEx}^{\varepsilon}\textbf{-Txt}$*.*

Next, we compare $\textbf{FigEx}^{\varepsilon}\textbf{-Inf}$- and $\textbf{FigEx}^{\varepsilon}\textbf{-Txt}$-learning. The learnability from informants is usually properly larger than that from texts. However, they become same in this case.

Theorem 14. *For all* $\varepsilon \in \mathbb{R}^+$*,* $\textbf{FigEx}^{\varepsilon}\textbf{-Inf} = \textbf{FigEx}^{\varepsilon}\textbf{-Txt}$*.*

Corollary 6. *For all* $\varepsilon, \varepsilon' \in \mathbb{R}^+$*,* $\textbf{FigEx}^{\varepsilon}\textbf{-Inf} = \textbf{FigEx}^{\varepsilon'}\textbf{-Inf}$*.*

Moreover, we have the following hierarchy.

Theorem 15. *For all* $\varepsilon \in \mathbb{R}^+$*,* $\textbf{FigRefEx-Inf} \subset \textbf{FigEx}^{\varepsilon}\textbf{-Inf}$*.*

7 Measuring the Complexity of Learning with the Hausdorff Dimension and the VC Dimension

Here we measure the number of positive examples, one of the complexity of learning, by using the Hausdorff dimension and the VC dimension. First, we show that the Hausdorff dimension of a target figure gives the lower bound of the number of positive examples.

In the following, assume that the initial interval is I_{U} for simplicity, and fix a partition that is a net, where $\|\Sigma\| = b^n$. Let $N^k(K)$ denote the number of positive examples of a target figure K at level k, that is, $N^k(K) = \|\mathcal{Q}(K) \cap \mathcal{S}^k\|$. Note that for all $k \in \mathbb{N}$, $\|\mathcal{S}^k\| = b^{kn}$ and the diameter $\operatorname{diam}(k) = \sqrt{n} b^{-k}$.

Theorem 16. *For every figure* $K \in \mathcal{K}^*$ *and for any* $s < \dim_{\mathrm{H}} K$*, if we take* k *large enough, the number* $N^k(K) \geqslant b^{ks}$*.*

If a target figure $K \in \kappa(\mathcal{P})$, we can bound the number $N^k(K)$ more precisely. In addition, we can decide when k becomes large enough.

Theorem 17. *For every figure* $K \in \kappa(\mathcal{P})$ *with* $K = \kappa(\mathrm{SP}(W))$ *and for every* $k \geqslant \max\{|w| \mid w \in W\}$*, the number* $N^k(K) \geqslant b^{k \dim_{\mathrm{H}} K}$*.*

Example 5. Let $n = 2$, a target figure K be the Sierpiński triangle, and ψ be the partition defined in Example 4. Note that $\|\Sigma\| = 2^n$ and $\dim_{\mathrm{H}} K = \log 3 / \log 2 = 1.584\dots$. From Theorem 17, $2^{\dim_{\mathrm{H}} K} = 3 \leqslant N^1(K)$ at level 1 and $4^{\dim_{\mathrm{H}} K} = 9 \leqslant N^2(K)$ at level 2. Actually, $N^1(K) = 4$ and $N^2(K) = 13$. Note that K is already covered by 3 and 9 intervals at level 1 and 2, respectively (Fig. 3).

Next, we introduce the Vapnik-Chervonenkis (VC) dimension and show that we can also use the VC dimension to characterize the number of positive examples. Intuitively, the VC dimension is a parameter of the separability and well known that we can bound example sizes in the Valiant-style learning model (also called PAC learning model) with the VC dimension [17].

For all $\mathcal{R} \subseteq \mathcal{P}$ and $\mathcal{I} \subseteq \mathcal{S}^*$, define $\Pi_{\mathcal{R}}(\mathcal{I}) := \{\mathcal{Q}(\kappa(P)) \cap \mathcal{I} \mid P \in \mathcal{R}\}$. If $\Pi_{\mathcal{R}}(\mathcal{I}) = 2^{\mathcal{I}}$, we say that $\mathcal{I}$ is *shattered* by $\mathcal{R}$. Here the *VC dimension* of $\mathcal{R}$, denoted by $\dim_{\mathrm{VC}} \mathcal{R}$, is the cardinality of the largest set $\mathcal{I}$ shattered by $\mathcal{R}$.

We define $\mathcal{P}^k := \{\, \mathrm{SP}(W) \in \mathcal{P} \mid |w| = k \text{ for all } w \in W \,\} \cup \{\mathrm{SP}(\emptyset)\}$, where $\kappa(\mathrm{SP}(\emptyset)) = \emptyset$ for technical reasons. The VC dimension of the set of level k programs $\mathcal{P}^k$ is equal to the cardinality of $\mathcal{S}^k$.

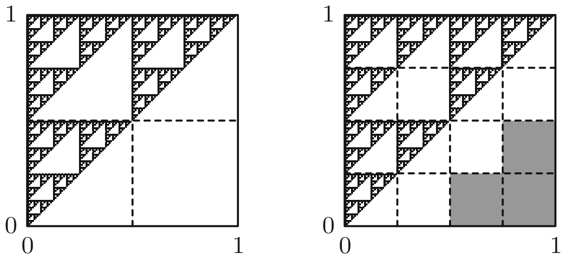


Fig. 3. Positive and negative examples for the Sierpiński triangle at level 1 and 2. White (resp. gray) squares mean positive (resp. negative) examples.

Lemma 5. *At each level k, we have* $\dim_{\mathrm{VC}} \mathcal{P}^k = \|\mathcal{S}^k\|$.

Therefore we can rewrite Theorems 16 and 17 as follows.

Theorem 18. *For every figure $K \in \mathcal{K}^*$ and for any $s < \dim_{\mathrm{H}} K$, if we take k large enough, $N^k(K) \geqslant (\dim_{\mathrm{VC}} \mathcal{P}^k)^{ks}$. Moreover, if $K \in \kappa(\mathcal{P})$ with $K = \kappa(\mathrm{SP}(W))$, for every $k \geqslant \max\{|w| \mid w \in W\}$, $N^k(K) \geqslant (\dim_{\mathrm{VC}} \mathcal{P}^k)^{k \dim_{\mathrm{H}} K}$.*

These results demonstrate a relationship among the complexities of learning figures (numbers of positive examples), classes of hypotheses (VC dimension), and target figures (Hausdorff dimension).

8 Conclusion

We have formulated learning of figures using self-similar sets based on the Gold-style learning model, and demonstrated the hierarchy of learning criteria (Fig. 1). This learning model can be viewed as the constructive interpretation of Collage Theorem, which is well known theorem in Fractal Geometry. Furthermore, we have measured the lower bound of the number of positive examples by using the Hausdorff dimension of a target figure and the VC dimension of a set of self-similar programs. Our results indicate that Gold-style learning model can be a basis of machine learning for continuous objects (*i.e.*, uncountable sets).

Self-similar sets can be viewed as a geometric interpretation of languages recognized by ω-automata (Büchi-automata), and learning of such languages has been already investigated [6, 15]. Thus comparison of the study and our study is a important future work.

In the field of Computable Analysis, representations of compact sets have been studied [5], and *admissible representations* are known as a key concept in the field [23]. It is a future work to study the relationship between such representations and the representations of figures in our learning model.

Acknowledgment

This work was partly supported by Grant-in-Aid for Scientific Research from the Japan Society for the Promotion of Science.

References

[1] Angluin, D.: Inductive inference of formal languages from positive data. Information and Control 45(2), 117–135 (1980)

[2] Barnsley, M.F.: Fractals Everywhere, 2nd edn. Morgan Kaufmann, San Francisco (1993)

[3] Beer, G.A.: Topologies on Closed and Closed Convex Sets, 1st edn. Mathematics and Its Applications, vol. 268. Kluwer Academic Publishers, Dordrecht (1993)

[4] Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer, Heidelberg (2007)

[5] Brattka, V., Presser, G.: Computability on subsets of metric spaces. Theoretical Computer Science 305(1-3), 43–76 (2003)

[6] de la Higuera, C., Janodet, J.-C.: Inference of ω-languages from prefixes. In: Abe, N., Khardon, R., Zeugmann, T. (eds.) ALT 2001. LNCS (LNAI), vol. 2225, pp. 364–377. Springer, Heidelberg (2001)

[7] Dieudonné, J.: Foundations of Modern Analysis. Academic Press, London (1960)

[8] Falconer, K.: Fractal Geometry: Mathematical Foundations and Applications. Wiley, Chichester (2003)

[9] Federer, H.: Geometric Measure Theory. Springer, New York (1996)

[10] Gold, E.: Language identification in the limit. Information and Control 10(5), 447–474 (1967)

[11] Hirowatari, E., Arikawa, S.: A comparison of identification criteria for inductive inference of recursive real-valued functions. Theoretical Computer Science 268(2), 351–366 (2001)

[12] Hirowatari, E., Arikawa, S.: Inferability of recursive real-valued functions. In: Algorithmic Learning Theory, pp. 18–31. Springer, Heidelberg (1997)

[13] Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation. Addison Wesley Publishing Company, Reading (1979)

[14] Hurewicz, W., Wallman, H.: Dimension Theory. Princeton University Press, Princeton (1948)

[15] Jain, S., Luo, Q., Semukhin, P., Stephan, F.: Uncountable automatic classes and learning. In: Gavaldà, R., Lugosi, G., Zeugmann, T., Zilles, S. (eds.) ALT 2009. LNCS (LNAI), vol. 5809, pp. 293–307. Springer, Heidelberg (2009)

[16] Jain, S., Osherson, D., Royer, J.S., Sharma, A.: Systems That Learn, 2nd edn. The MIT Press, Cambridge (1999)

[17] Kearns, M.J., Vazirani, U.V.: An Introduction to Computational Learning Theory. The MIT Press, Cambridge (1994)

[18] Lloyd, J.W.: Foundations of Logic Programming, 2nd edn. Springer, Heidelberg (1993)

[19] Mukouchi, Y., Arikawa, S.: Towards a mathematical theory of machine discovery from facts. Theoretical computer science 137(1), 53–84 (1995)

[20] Rosenblatt, F.: The perceptron: A probabilistic model for information storage and organization in the brain. Psychological review 65(6), 386–408 (1958)

[21] Sakurai, A.: Inductive inference of formal languages from positive data enumerated primitive-recursively. In: Algorithmic Learning Theory, JSAI, 73–83 (1991)

[22] Sugiyama, M., Hirowatari, E., Tsuiki, H., Yamamoto, A.: Learning figures with the Hausdorff metric by self-similar sets. In: Proc. of LLLL 2009, pp. 27–34 (2009)

[23] Weihrauch, K.: Computable Analysis: An Introduction. Springer, Heidelberg (2000)

Inductive Inference of Languages from Samplings

Sanjay Jain[1,⋆] and Efim Kinber[2]

[1] School of Computing, National University of Singapore, Singapore 117417, Republic of Singapore
sanjay@comp.nus.edu.sg

[2] Department of Computer Science, Sacred Heart University, Fairfield, CT 06825-1000, U.S.A.
kinbere@sacredheart.edu

Abstract. We introduce, discuss, and study a model for inductive inference from samplings, formalizing an idea of learning different "projections" of languages. One set of our results addresses the problem of finding a uniform learner for all samplings of a language from a certain set when learners for particular samplings are available. Another set of results deals with extending learnability from a large natural set of samplings to larger sets. A number of open problems is formulated.

Keywords: Inductive inference, samplings, sublanguages.

1 Introduction

Consider the following model of learning. A learner gets data about a target concept, one piece at a time. As the learner is receiving the data, it outputs its conjectures on what the target might be. If the sequence of conjectures of the learner converges to a correct hypothesis about the target concept, then one might say that the learner has successfully learnt the concept. This is essentially the model of **TxtEx**-learning considered by Gold [Gol67].

In our paper, the target concept is a language L from a class $\mathcal{L}$ of possible languages. The learner gets as input, one element at a time, in arbitrary order with repetitions allowed, members of the target language (such a presentation is called a *text* of the language; note that negative data is not presented to the learner in this model). The conjectures made by the learner take the form of a grammar or acceptor in some acceptable programming system [Rog67]. The learner is then successful (that is, the learner **TxtEx**-identifies the target language) if the sequence of conjectures converges to a grammar which generates/accepts the target language L.

Expecting that the learner gets all elements of the target language is unrealistic. Often it is difficult to obtain full data, and a learner gets actually elements of only a subset X of the target language L. Of course, then it may be unrealistic to expect the learner to learn the full language. Thus, in such a situation one

⋆ Supported in part by NUS grant number R252-000-308-112 and C252-000-087-001.

© Springer-Verlag Berlin Heidelberg 2010

says that the learner is successful if it converges to a grammar for a language L' such that $X \subseteq L' \subseteq L$. In [JK08], the authors considered this model of learning. Note that here, the learner is supposed to be successful in the above sense for all subsets X of the target language L.

In this paper, we consider a different variant of the model explored in [JK08]: every language from a target class is being learnt from inputs defined by the same *sampling*. As in the model in [JK08], the learner must produce in the limit a grammar covering the input sampling of the target language, and the final grammar must represent a subset of the target language containing all data from the input sampling. The difference of our variant from the model in [JK08] is that, in [JK08], any *arbitrary* subset of the target language may appear on the input, whereas in our model the input is defined by a certain (fixed) sampling.

Since positive data in the process of inductive inference can be viewed as being supplied by a teacher, it is natural to assume that, in some situations, the teacher may have difficulty providing full positive data — because, for example, it is too time-consuming, or it is too lengthy. In such a situation, the teacher may provide just a part of the target language using some natural representation of the language known to the teacher. For example, the teacher may provide, say, the 2nd, 4th, 6th, etc. elements of this representation, or some other sequence of positive data that is representative of the target language as a whole and/or represents one or another salient aspect of it. Say, if the target language consists of all primes, the teacher may provide the sampling containing just every other prime (or another sampling of a similar sort) to give the learner a good idea of what the language is about. Or, if all languages in the target class are infinite, the teacher may omit some elements of the target language as being of lesser importance for the overall understanding of the concept. (Our idea of sampling can be traced to Trakhtenbrot's paper [Tra73]).

The learner is required to be able to produce in the limit a grammar that covers the input part (representing a given sampling) of the target language and does not exceed the target language (in set containment sense). Now, obviously, there are many different issues of interest: for example, does learnability of every specific sampling (from some class of samplings) imply overall (uniform) learnability for all samplings, how does knowledge of the sampling choice (provided by the teacher) affect learning capabilities, etc.

The idea of considering inductive inference from inputs defined by different samplings was first suggested by R. Freivalds in [Fre74], where he studied how learnability on different specific orderings of the input function graph (provided by the teacher) can affect overall learnability of the function regardless of the order of input. When one assumes that languages are sets of positive integers (as, following the tradition of classic inductive inference, it is done in our paper), and considers learning languages from positive data only, the teacher may use samplings based on many different natural representations of all positive data. For example, one can fix some standard way of enumerating all recursively enumerable sets, and then, for a given enumeration $a_0, a_1, a_2, a_3, \ldots$ of a (recursively enumerable) language L, the sampling, say, $A = \{1, 3, 5, \ldots\}$ will define the sublanguage $a_1, a_3, a_5, \ldots$. However, it turns out that this approach makes learnability dependent on the choice of enumerating mechanism, and, thus, rather unnatural.

We have chosen a formalization of the concept of sampling based on the representation of target languages in the increasing order. Namely, for any language L, consider the increasing order of all elements $a_0, a_1, a_2, a_3, \ldots$. Then a *sampling* A is a set of positive integers, say, $i_1, i_2, \ldots$ and the corresponding A-*sampling* (sublanguage) of L is the language $\{a_{i_1}, a_{i_2}, a_{i_3}, \ldots\}$. For example, for the sampling $A = \{5, 6, 7, \ldots\}$, one gets the A-sampling of L containing all the elements of L except the first five smallest ones.

Given the aforementioned formalization, we study several natural questions. The first question is if, given a class of languages $\mathcal{L}$ and a set of samplings $\mathcal{A}$, learnability of A-samplings of languages from $\mathcal{L}$ for every specific $A \in \mathcal{A}$ implies uniform learnability (that is, by one learner) of all A-samplings of languages from $\mathcal{L}$ for all $A \in \mathcal{A}$. We answer this question in the negative in Theorem 7, using the set of all possible samplings. For some special set of samplings, we also show that a class witnessing separation of non-uniform and uniform learnability for this set of samplings can be uniformly learned with just one error in the final conjecture (Theorem 8). A related result addresses the question whether uniform learnability on each of the two different sets of samplings implies uniform learnability on all samplings from the union of these two sets: as we show in Theorem 10, the answer is negative even if each set contains just one recursive sampling. On the other hand, we suggest a simple sufficient condition for learnability on the union of two sets of samplings when the learner gets access to the sampling A (from the oracle, or as a separate input), see Proposition 12.

We also studied the following problem: what are the circumstances when learnability of a class from some natural set of samplings ensures learnability of the class from a larger natural set of samplings? For example, is learnability of a class on the set of A-samplings for all infinite recursively enumerable A-s powerful enough to ensure learnability of the class from all infinite samplings? We were able to get only some negative results so far. In particular, we have shown that learnability of a class from all A-samplings for all infinite recursively enumerable A-s does not imply the learnability of the class from all infinite samplings (Theorem 16). Similarly, it turns out that learnability of a class from all simple recursively enumerable samplings does not imply learnability of the class from all infinite recursively enumerable samplings (Theorem 17). Moreover, it turns out that the learnability of a class from all samplings but some recursive sampling A (and all its subsets), does not imply that the class is learnable from all samplings (Theorem 18).

2 Preliminaries

2.1 Notations

Any unexplained recursion theoretic notation is from [Rog67]. Let N denote the set of natural numbers, $\{0, 1, 2, 3, \ldots\}$. Symbols $\emptyset$, $\subseteq$, $\subset$, $\supseteq$, and $\supset$ denote the empty set, subset, proper subset, superset, and proper superset, respectively. Symmetric difference of A and B is denoted by $A\mathbf{\Delta}B$. That is, $A\mathbf{\Delta}B = (A - B) \cup (B - A)$. $\mathcal{P}(A)$ denotes the power set of A, that is, $\mathcal{P}(A) = \{B \mid B \subseteq A\}$. The maximum and minimum of a set are respectively denoted by $\max(\cdot), \min(\cdot)$, where we take $\max(\emptyset) = 0$ and $\min(\emptyset) = \infty$. For a set $S = \{x_0, x_1, \ldots\}$, where $x_0 < x_1 < \ldots$, we call x_i the i-th minimal element (or just the i-th element) of S (thus, the 0-th (minimal) element is the minimal element of a set). The cardinality of a set S is denoted by $\mathrm{card}(S)$. We use $\mathrm{card}(S) \leq *$ to denote that the cardinality of S is finite. For $a \in N \cup \{*\}$, $A =^a B$ denotes that $\mathrm{card}(A\mathbf{\Delta}B) \leq a$. Quantifiers $\forall^\infty$ and $\exists^\infty$ respectively denote 'for all but finitely many' and 'there exist infinitely many'.

We let $\langle \cdot, \cdot \rangle$ denote a computable 1–1 and onto mapping from $N \times N$ to N (see [Rog67]). We assume without loss of generality that $\langle \cdot, \cdot \rangle$ is increasing in both its arguments. Let $\pi_1^2(\langle x, y \rangle) = x$ and $\pi_2^2(\langle x, y \rangle) = y$. The pairing function can be extended to coding of n-tuples in a natural way by taking $\langle x_1, x_2, \ldots, x_n \rangle = \langle x_1, \langle x_2, x_3, \ldots, x_n \rangle \rangle$, for $n > 2$. The corresponding projection functions are $\pi_i^n(\langle x_1, x_2, \ldots, x_n \rangle) = x_i$.

Let $\mathcal{R}$ denote the set of all recursive functions. For a partial function η, $\eta(x)\downarrow$ denotes that $\eta(x)$ is defined. $\eta(x)\uparrow$ denotes that $\eta(x)$ is undefined. We let $\eta[n]$ denote the partial function, $\{(x, \eta(x)) \mid x < n\}$. By φ we denote a fixed *acceptable* programming system for the partial computable functions from N to N [Rog67, HU79]. Then, φ_i denotes the i-th partial computable function in this programming system, and i is called a program for the partial function φ_i. By Φ we denote a fixed Blum complexity measure [Blu67, HU79] for the φ-system. Intuitively, $\Phi_i(x)$ denotes the resources (say time or space) needed to compute $\varphi_i(x)$.

Languages are subsets of N. By W_i we denote $\mathrm{domain}(\varphi_i)$. Thus, W_i is the recursively enumerable (r.e.) set/language accepted by φ_i. We also say that i is a grammar for W_i. Symbol $\mathcal{E}$ denotes the set of all r.e. languages. By $W_{i,s}$ we denote the set $\{x < s \mid \Phi_i(x) < s\}$. L, with or without decorations, ranges over $\mathcal{E}$. We let χ_L denote the characteristic function of L. We let $\overline{L} = N - L$, that is the complement of L. Symbol $\mathcal{L}$, with or without decorations, ranges over subsets of $\mathcal{E}$.

A set S is called *immune* [Rog67] iff S is infinite, and for all infinite r.e. sets X, $X \not\subseteq S$. A set S is called *simple* [Rog67] iff S is recursively enumerable and $\overline{S}$ is immune.

For a total function f, let $L_f = \{\langle x, f(x) \rangle \mid x \in N\}$. For any, possibly partial, function g, let Zext_g be the function defined as follows: $\mathrm{Zext}_g(x) = g(x)$, if $x \in \mathrm{domain}(g)$; $\mathrm{Zext}_g(x) = 0$, otherwise.

We will consider the following classes of languages and functions:

- INF is the class of all infinite sets.
- REinf is the class of all infinite recursively enumerable sets.
- $\mathrm{INIT} = \{L \mid (\exists n)[L = \{x \mid x < n\}]\}$, the class of initial segments of N.
- $\mathrm{COINIT} = \{L \mid (\exists n)[L = \{x \mid x \geq n\}]\}$, the class of coinitial segments of N.
- $\mathrm{SD} = \{f \in \mathcal{R} \mid \varphi_{f(0)} = f\}$.
- $\mathrm{AZext} = \{f \in \mathcal{R} \mid \mathrm{domain}(\varphi_{f(0)}) \in \mathrm{INIT}, \text{ and } f = \mathrm{Zext}_{\varphi_{f(0)}}\}$.

2.2 Preliminaries for Learning

A *text* T is a mapping from N into $(N \cup \{\#\})$. Thus, $T(i)$ represents the $(i+1)$-st element in the text. Intuitively, a text denotes the presentation of elements of a language, with #s representing pauses in the presentation. We let T, with or without decorations, range over texts. Content of a text T, denoted content(T), is the set of natural numbers in the range of T. A text T is for a language L iff content$(T) = L$. $T[n]$ denotes the initial sequence of T of length n, that is $T[n] = T(0)T(1) \ldots T(n-1)$.

A finite sequence is a mapping from an initial segment of N into $(N \cup \{\#\})$. The empty sequence is denoted by Λ. Content of σ, denoted content(σ), is the set of natural numbers in the range of σ. The *length* of σ, denoted by $|\sigma|$, is the number of elements in σ. For $i < |\sigma|$, $\sigma(i)$ denotes the $(i+1)$-th element in σ. For $n \leq |\sigma|$, $\sigma[n]$ denotes the initial sequence of σ of length n. SEQ denotes the set of all finite sequences. Thus, SEQ $= \{T[n] \mid n \in N, T \text{ is a text}\}$. We let σ and τ, with or without decorations, range over SEQ. We denote the sequence formed by the concatenation of τ at the end of σ by $\sigma\tau$.

An *inductive inference machine* (IIM) [Gol67] is an algorithmic mapping from SEQ to N. We also use the term *learner* or *learning machine* for IIM. We let $\mathbf{M}$, with or without decorations, range over IIMs. We say that $\mathbf{M}$ converges on T to i, (written: $\mathbf{M}(T)\downarrow = i$) iff $(\forall^\infty n)[\mathbf{M}(T[n]) = i]$.

The following define some of the notions of learning.

Definition 1. [Gol67, CL82]

(a) $\mathbf{M}$ **TxtEx**-*identifies an r.e. language* L (written: $L \in \mathbf{TxtEx}(\mathbf{M})$) just in case, for all texts T for L, $\mathbf{M}(T[n])$ is defined for all n and $(\exists i \mid W_i = L)(\forall^\infty n)[\mathbf{M}(T[n]) = i]$.

(b) $\mathbf{M}$ **TxtEx**-*identifies a class* $\mathcal{L}$ *of r.e. languages* (written: $\mathcal{L} \subseteq \mathbf{TxtEx}(\mathbf{M})$) just in case $\mathbf{M}$ **TxtEx**-identifies each language from $\mathcal{L}$.

(c) $\mathbf{TxtEx} = \{\mathcal{L} \subseteq \mathcal{E} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{TxtEx}(\mathbf{M})]\}$.

Definition 2. [OW82, Gol67, CL82]

(a) $\mathbf{M}$ **TxtBc**-*identifies an r.e. language* L (written: $L \in \mathbf{TxtBc}(\mathbf{M})$) just in case, for all texts T for L, for all but finitely many n, $W_{\mathbf{M}(T[n])} = L$.

(b) $\mathbf{M}$ **TxtBc**-*identifies a class* $\mathcal{L}$ *of r.e. languages* (written: $\mathcal{L} \subseteq \mathbf{TxtBc}(\mathbf{M})$) just in case $\mathbf{M}$ **TxtBc**-identifies each language from $\mathcal{L}$.

(c) $\mathbf{TxtBc} = \{\mathcal{L} \subseteq \mathcal{E} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{TxtBc}(\mathbf{M})]\}$.

There exists a recursive sequence of total IIMs, $\mathbf{M}_0, \mathbf{M}_1, \ldots$ such that, for the criteria of learning $\mathbf{I}$ discussed in this paper, for each $\mathcal{L} \in \mathbf{I}$, some $\mathbf{M}_i$ witnesses that $\mathcal{L} \in \mathbf{I}$. This can be shown essentially along the same lines as done for **TxtEx**-learning in [OSW86]. Thus, any learner $\mathbf{M}$ can be considered to be equivalent to some $\mathbf{M}_j$ from such an enumeration (with respect to being able to learn a class of languages under a criterion of inference considered in this paper). We fix one such enumeration $\mathbf{M}_0, \mathbf{M}_1, \ldots$, and will from now on consider only learners from this list.

Definition 3. (a) [Ful90] σ is said to be a **TxtEx**-*stabilizing sequence* for $\mathbf{M}$ on L, iff (i) content$(\sigma) \subseteq L$, and (ii) for all τ such that content$(\tau) \subseteq L$, $\mathbf{M}(\sigma\tau) = \mathbf{M}(\sigma)$.

(b) [BB75, Ful90] σ is said to be a **TxtEx**-*locking sequence* for $\mathbf{M}$ on L, iff (i) σ is a **TxtEx**-stabilizing sequence for $\mathbf{M}$ on L and (ii) $W_{\mathbf{M}(\sigma)} = L$.

If $\mathbf{M}$ **TxtEx**-identifies L, then every **TxtEx**-stabilizing sequence for $\mathbf{M}$ on L is a **TxtEx**-locking sequence for $\mathbf{M}$ on L. Furthermore, one can show that if $\mathbf{M}$ **TxtEx**-identifies L, then for every σ such that content$(\sigma) \subseteq L$, there exists a **TxtEx**-locking sequence, which extends σ, for $\mathbf{M}$ on L (see [BB75, Ful90]).

Similar locking sequence results can be proved for other criteria of inference considered in this paper.

3 Definitions for Learning from Samplings

Suppose $S = \{y_0, y_1, y_2, \ldots\}$, where $y_0 < y_1 < \ldots$, and $R \subseteq S$. Then, define Order$(R, S) = \{i \mid y_i \in R\}$. We call any subset of N a *sampling*. For a sampling A and a set S, we call X an A-sampling of S iff Order$(X, S) = A$.

Note that, if A is infinite and S is finite, then there is no A-sampling of S. On the other hand, for every infinite set S, and every sampling A, there is a (unique) A-sampling of S. Thus, for ease of notation, when learning from samplings, we will only consider infinite languages, without always explicitly mentioning so. This does not effect our results, as all the diagonalizations in this paper can be achieved using classes of infinite languages. Note that the samplings themselves may or may not be infinite.

The following definition now formalizes our notion of learning from samplings. Note that the model for learning sublanguages of a target language in [JK08] is different from the one formalized in the definition below, as the model in [JK08] requires learners to learn arbitrary input sublanguages, rather than the ones defined by specific samplings as considered in this paper. In other words, the model in [JK08] is $\mathbf{UniSublang}_{\mathcal{P}(N)}$, a special case of the models considered in this paper.

Definition 4. (a) Suppose a sampling $A \subseteq N$ is given. $\mathbf{M}$ $\mathbf{SublangEx}_A$-identifies an infinite language L iff, for A-sampling X of L, for any text T for X, there exists an n such that, (i) for all $m \geq n$, $\mathbf{M}(T[m]) = \mathbf{M}(T[n])$, and (ii) $X \subseteq W_{\mathbf{M}(T[n])} \subseteq L$.

$\mathbf{M}$ $\mathbf{SublangEx}_A$-identifies a class $\mathcal{L}$ of infinite languages iff $\mathbf{M}$ $\mathbf{SublangEx}_A$-identifies each language in $\mathcal{L}$.

$\mathbf{SublangEx}_A$ denotes the collection of all $\mathcal{L}$ which can be $\mathbf{SublangEx}_A$-identified by some learner $\mathbf{M}$.

(b) Let $\mathcal{A} \subseteq \mathcal{P}(N)$ be a set of samplings.

(b.1) $\mathbf{SublangEx}_{\mathcal{A}}$ denotes the collection of all $\mathcal{L}$ such that, for each $A \in \mathcal{A}$, $\mathcal{L} \in \mathbf{SublangEx}_A$ (this is the non-uniform version).

(b.2) $\mathbf{UniSublangEx}_{\mathcal{A}}$ denotes the collection of all $\mathcal{L}$ for which there exists a learner $\mathbf{M}$ such that, for each $A \in \mathcal{A}$, $\mathbf{M}$ $\mathbf{SublangEx}_A$-identifies $\mathcal{L}$ (this is the uniform version).

(b.3) **PUniSublangEx**$_{\mathcal{A}}$ denotes the collection of all $\mathcal{L}$ for which there exists a learner **M** such that, for each $A \in \mathcal{A}$, **M** using an oracle for A, **SublangEx**$_A$-identifies $\mathcal{L}$ (this is the pseudo-uniform version, where the learner has access to the sampling A in oracle form).

One can similarly define the criteria for **SublangBc**-learning. We say that **M** **SublangBc**$_A$-identifies $\mathcal{L}$ iff for all $L \in \mathcal{L}$, for $X \subseteq L$ such that $\text{Order}(X, L) = A$, for any text T for X, for all but finitely many n, $X \subseteq W_{\mathbf{M}(T[n])} \subseteq L$.

In this paper, we will be mainly concentrating on **Sublang** and **UniSublang**-learning paradigms. The criterion **PUniSublangEx** is used more for emphasizing what happens if a uniform learner "knows", in some way, the sampling A which it is getting. The usage of an oracle here is more for convenience, and the results presented in the paper will hold even if one gives the set A to the learner in the form of a separate text containing exactly the elements of A.

4 Results

Our first goal is to show that there are classes of languages non-uniformly learnable on all samplings, but not uniformly learnable, even just on samplings from COINIT. We begin with the following useful proposition and a corollary from it.

Proposition 5. $\mathcal{L} = \{L_f \mid f \in SD \cup AZext\} \notin$ **TxtBc**, *even when the texts given to the learner are increasing texts.*

Proof. This proposition can be proved essentially along the same lines as the proof of the non-union theorem [BB75]. For ease of presentation, we will give the proof for learning from arbitrary texts. As the class $\mathcal{L}$ used consists only of L_f such that $f \in \mathcal{R}$, such texts can be effectively converted to increasing texts. Suppose, by way of contradiction, that **M** **TxtBc**-identifies $\mathcal{L}$. Then, by implicit use of Kleene's recursion theorem [Rog67], there exists an e such that φ_e may be defined as follows. Let $\varphi_e(0) = e$. φ_e^s denotes φ_e defined before stage s, and x_s denotes the largest x such that $\varphi_e(x)$ is defined before stage s. Let σ_0 contain just one element: $\langle 0, e\rangle$. It will be the case that $\text{content}(\sigma_s) = \{\langle x, \varphi_e(x)\rangle \mid x \leq x_s\}$. Go to stage 0.

```
Stage $s$:
1. Search for a $\sigma \supseteq \sigma_s$ and $y \in N$ such that: $\text{content}(\sigma) \subseteq L_{\text{Zext}_{\varphi_e^s}}$, $y > x_s$,
      $\langle y, 0\rangle \notin \text{content}(\sigma)$ and $\langle y, 0\rangle \in W_{\mathbf{M}(\sigma)}$.
2. If and when such a $\sigma$ and $y$ are found,
         let $z$ be maximum such that $\langle z, 0\rangle \in \text{content}(\sigma)$,
         let $\varphi_e(y) = 1$,
         let $\varphi_e(x) = 0$, for $x \neq y$ such that $x_s < x \leq y + z + 1$.
         let $x_{s+1} = y + z + 1$.
         let $\sigma_{s+1}$ be an extension of $\sigma$ such that $\text{content}(\sigma_{s+1}) = \{\langle x, \varphi_e(x)\rangle \mid$
            $x \leq x_{s+1}\}$.
   Go to stage $s + 1$
End stage $s$
```

It is easy to verify that, if all stages terminate, then φ_e is total, $\varphi_e \in$ SD, $T = \bigcup_{s\in N} \sigma_s$ is a text for L_{φ_e}, and $W_{\mathbf{M}(T[n])} \neq L_{\varphi_e}$, for infinitely many n (as $W_{\mathbf{M}(\sigma)} \neq L_{\varphi_e}$, for each of σ found in the stages).

On the other hand, if stage s starts but does not finish, then for $g = \text{Zext}_{\varphi_e} \in$ AZext, we have that for any text T for L_g which extends σ_s, $\mathbf{M}(T[n])$ is not a grammar for L_g for any $n > |\sigma_s|$ (otherwise, the search in step 1 would succeed).

Thus, we have that $\mathbf{M}$ cannot **TxtBc**-identify $\mathcal{L}$. ∎

Corollary 6. *Let* $SD_j = \{f \mid (\exists g \in SD)[(\forall x < j)[f(x) = \langle j, g(0), 1\rangle]$ *and* $(\forall x \geq j)[f(x) = \langle j, g(0), g(x-j)+2\rangle]]\}$.

Let $AZext_j = \{f \mid (\exists g \in AZext)[(\forall x < j)[f(x) = \langle j, g(0), 0\rangle]$ *and* $(\forall x \geq j)[f(x) = \langle j, g(0), g(x-j)+2\rangle]]\}$.

Let $\mathcal{L}_j = \{L_f \mid f \in SD_j \cup AZext_j\}$.

Then $\mathcal{L}_j \notin \mathbf{UniSublangBc}_A$, *where* $A = \{j, j+1, j+2, \ldots\}$. *Thus,* $\mathcal{L}_j \notin \mathbf{UniSublangBc}_{COINIT}$.

Note: We used +2 above just to make sure that $\langle j, e, f(x)\rangle$ for $x < j$ are smaller than $\langle j, e, f(x)\rangle$, for $x \geq j$.

Now we can prove the separation result for the uniform and non-uniform learnability from samplings. The following Theorem holds even if we replace **Uni** by **PUni**.

Theorem 7. $\mathbf{SublangEx}_{\mathcal{P}(N)} - \mathbf{UniSublangBc}_{COINIT} \neq \emptyset$.

Proof. Let $\mathcal{L}_j$ be as in the Corollary 6. Fix $L_j \in \mathcal{L}_j$ that witnesses that $\mathbf{M}_j$ does not $\mathbf{UniSublangBc}_{\text{COINIT}}$-identify $\mathcal{L}_j$. Let $\mathcal{L} = \{L_j \mid j \in N\}$. Then, clearly $\mathcal{L} \notin \mathbf{UniSublangBc}_{\text{COINIT}}$.

To see that $\mathcal{L} \in \mathbf{SublangEx}_{\mathcal{P}(N)}$, let $A \in \mathcal{P}(N)$ be given. If $A = \emptyset$, then $\mathcal{L}$ is trivially in $\mathbf{SublangEx}_A$, as the learner can just output $\emptyset$.

If $A \neq \emptyset$, then let $k \in A$. Now, the learner can $\mathbf{SublangEx}_A$-learn $\mathcal{L}$ as follows. If the input text contains an element $\langle x, \langle j, e, y\rangle\rangle$, for some $j \leq k$, then output a grammar for L_j (as there are only finitely many such j, the learner can "code" these finitely many cases).

Otherwise, the input text will contain $\langle k, \langle j, e, y\rangle\rangle$ for some $j > k$, for some $y \in \{0,1\}$ (as the k-th least element in L_j is $\langle k, \langle j, e, y\rangle\rangle$, for some $y \in \{0,1\}$). If $y = 1$, then output a grammar for L_f, where $f(x) = \langle j, e, 1\rangle$, if $x < j$, and $f(x) = \langle j, e, \varphi_e(x-j)+2\rangle$, if $x \geq j$. If $y = 0$, then, in the limit, search for the least x_0 such that $\varphi_e(x_0)$ is not defined. Then, output a grammar for L_f, where $f(x) = \langle j, e, 0\rangle$, if $x < j$, and $f(x) = \langle j, e, g(x-j)+2\rangle$, if $x \geq j$, where $g = \text{Zext}_{\varphi_e[x_0]}$. It is easy to verify that the above learner will $\mathbf{SublangEx}_A$-identify $\mathcal{L}$. ∎

Now we exhibit a different result on separation of non-uniform and uniform learnability from samplings: for a certain set $\mathcal{A}$ of samplings, there exists a class of languages, which is non-uniformly learnable on all samplings from the given set $\mathcal{A}$, uniformly learnable with just one error in the final conjecture, but not uniformly learnable without errors in conjectures even in **Bc** style.

Here a learner $\mathbf{M}$ $\mathbf{SublangEx}^a_A$-identifies $\mathcal{L}$ iff, for all $L \in \mathcal{L}$, if $X \subseteq L$ is an A-sampling of L, then for any text T for X, there exists an n and a set Z such that, (i) for all $m \geq n$, $\mathbf{M}(T[m]) = \mathbf{M}(T[n])$, (ii) $X \subseteq Z \subseteq L$, and (iii) $Z =^a W_{\mathbf{M}(T[n])}$.

Theorem 8. *There exists a class $\mathcal{L}$ such that for $\mathcal{A} = \{N\} \cup \{N - \{k\} \mid k \in N\}$, $\mathcal{L} \in \mathbf{SublangEx}_{\mathcal{A}}$, $\mathbf{UniSublangEx}^1_{\mathcal{A}}$, but not in $\mathbf{UniSublangBc}_{\mathcal{A}}$.*

Proof. Let $\text{cyle}_j = \{\langle j, 2x\rangle \mid x \in N\}$.

Let $\text{cyle}^k_j = \{\langle j, 2x\rangle \mid x \in N, x \neq k\} \cup \{\langle j, 2k+1\rangle\}$

Let $\mathcal{L}_j = \{\text{cyle}_j\} \cup \{\text{cyle}^k_j \mid k \in N\}$.

Let $L_j = \text{cyle}_j$, if $\mathbf{M}_j$ does not $\mathbf{TxtBc}$-learn cyle_j. Otherwise, let $L_j = \text{cyle}^{k_j}_j$, where k_j is the least number such that $\langle j, 2k_j\rangle$ does not belong to the least $\mathbf{TxtBc}$-locking sequence (say τ_j) for $\mathbf{M}_j$ on cyle_j. Note that on any text extending τ_j for the set $\text{cyle}^{k_j}_j - \{\langle j, 2k_j+1\rangle\}$, $\mathbf{M}_j$ almost always outputs a grammar for cyle_j.

Let $\mathcal{L} = \{L_j \mid j \in N\}$. It follows immediately by definition of L_j above that $\mathcal{L} \notin \mathbf{UniSublangBc}_{\mathcal{A}}$.

On the other hand, $\mathcal{L}$ is easily seen to be in $\mathbf{SublangEx}_{\mathcal{A}}$. To see, this suppose $A \in \mathcal{A}$ is given. A learner can obtain the unique j such that the input text contains only elements of the form $\langle j, \cdot\rangle$. Now, if the input text contains $\langle j, 2k+1\rangle$, for some k, then the target language must be cyle^k_j and the learner can appropriately converge to a grammar for cyle^k_j; otherwise the target language is either cyle_j or cyle^k_j, for the unique k, if any, which is missing from A, and the learner can converge to a grammar for cyle_j or $\text{cyle}_j - \{\langle j, 2k\rangle\}$ depending on whether $\langle j, 2k\rangle$ belongs to the given input text or not.

Also, $\mathcal{L} \in \mathbf{UniSublangEx}^1_{\mathcal{A}}$, as witnessed by a learner which converges to a grammar for cyle^k_j, if it sees an element of the form $\langle j, 2k+1\rangle$ in the input text; otherwise, the learner converges to a grammar for cyle_j, where the input text only contains elements from cyle_j. ∎

Next, we establish a non-union result: learnability of a class of languages on each of two recursive samplings does not imply uniform learnability of the class on the set consisting of the two given samplings. First we establish a useful proposition.

Proposition 9. *Fix $j, m \in N$ and an IIM $\mathbf{M}$. Let S_j be a subset of $\{\langle j, 1, x\rangle \mid x \leq m\}$.*

Let $Z^f = S_j \cup \{\langle j, 2, \langle x, f(0), f(x)\rangle + m\rangle \mid x \in N\}$.

Then one of the following holds:

(a) there exists an $f \in SD$ such that for an increasing text T for $X_j = Z^f$, for infinitely many n, $W_{\mathbf{M}(T[n])} \cap \{\langle j, 2, y\rangle \mid y \in N\} \neq X_j - S_j$;

(b) not part (a) and there exists an $f \in AZext$ such that for an increasing text T for $X_j = Z^f$, for infinitely many n, $W_{\mathbf{M}(T[n])} \cap \{\langle j, 2, y\rangle \mid y \in N\} \neq X_j - S_j$.

Proof. Follows from Proposition 5. ∎

Now we establish the desired non-union result.

Theorem 10. *Suppose $\mathcal{A}_1 = \{A_1\}$ and $\mathcal{A}_2 = \{A_2\}$, where A_1 and A_2 are two distinct infinite recursive sets. Then,* $\mathbf{SublangEx}_{\mathcal{A}_1} \cap \mathbf{SublangEx}_{\mathcal{A}_2} - \mathbf{UniSublangBc}_{\mathcal{A}_1 \cup \mathcal{A}_2} \neq \emptyset$.

Proof. Without loss of generality we assume that the pairing function is increasing in all its arguments and that, for each j, if $\{w_0, w_1, \ldots\} = \{\langle j, r, x\rangle \mid r \in \{1, 2\}\}$, where for all i, $w_i < w_{i+1}$, then $\mathrm{card}(\{\langle j, 0, x\rangle \mid x \in N\} \cap \{x \mid w_i < x < w_{i+1}\}) \geq (i+2)$-th minimal elements of both A_1 and A_2. This ensures that there are enough gaps between elements of the form $\langle j, 1, \cdot\rangle$ and $\langle j, 2, \cdot\rangle$ to insert several elements of the form $\langle j, 0, \cdot\rangle$ as needed in the construction of L_j from X_j below.

Let $i_0 = \min(A_1 \mathbf{\Delta} A_2)$. Without loss of generality assume that $i_0 \in A_1$. Let i_1 be the smallest element in $A_2 - \{x \mid x \leq i_0\}$.

Let $S_j = \{\langle j, 1, x\rangle \mid x < i_0, x \in A_1\} \cup \{\langle j, 1, 2i_0 + 2\rangle\} \cup \{\langle j, 1, 2i_0 + 2 + 2y + 1\rangle \mid i_0 < y \leq i_1, y \in A_1\}$.

Let $m_j = 2i_0 + 2 + 2i_1 + 1$.

Now, if in Proposition 9, (a) holds for $S = S_j$, $m = m_j$ and $\mathbf{M} = \mathbf{M}_j$, then let X_j be as in Proposition 9 and let L_j be an r.e. set formed by adding elements of the form $\langle j, 0, \cdot\rangle$ to $X_j \cup \{\langle j, 1, 2i_0 + 2 + 2i_1 + 1\rangle\}$ such that $\mathrm{Order}(X_j \cup \{\langle j, 1, 2i_0 + 2 + 2i_1 + 1\rangle\}, L_j) = A_1 \cup \{i_1\}$ (here we assume that the elements of the form $\langle j, 0, \cdot\rangle$ which are added are the least ones possible so that one can, effectively from X_j, determine the elements which are added). We added $\langle j, 1, 2i_0 + 2 + 2i_1 + 1\rangle$ above just to make sure that the i_1-th element of L_j is $\langle j, 1, 2i_0 + 2 + 2i_1 + 1\rangle$ in case $i_1 \notin A_1$; in case $i_1 \in A_1$, $\langle j, 1, 2i_0 + 2 + 2i_1 + 1\rangle$ would already be in X_j. Note that the i_0-th element of L_j is $\langle j, 1, 2i_0 + 2\rangle$ and the i_1-th element of L_j is $\langle j, 1, 2i_0 + 2 + 2i_1 + 1\rangle$ (that is, the i_0-th element of L_j is of the form $\langle j, 1, 2x\rangle$ for some x, and the i_1-th element of L_j is of the form $\langle j, 1, 2x + 1\rangle$ for some x).

Otherwise, if in Proposition 9, (b) holds for $S = S_j$, $m = m_j$, and $\mathbf{M} = \mathbf{M}_j$, then let X_j be as in Proposition 9 and let L_j be an r.e. set formed by adding elements of the form $\langle j, 0, \cdot\rangle$ to $X_j \cup \{\langle j, 1, 2i_0 + 1\rangle\}$ such that $\mathrm{Order}(X_j \cup \{\langle j, 1, 2i_0 + 1\rangle\}, L_j) = A_2 \cup \{i_0\}$ (here we assume that the elements of the form $\langle j, 0, \cdot\rangle$ which are added are the least ones possible so that one can, effectively from X_j, determine the elements which are added). Note that the i_0-th element of L_j is $\langle j, 1, 2i_0 + 1\rangle$ and the i_1-th element of L_j is $\langle j, 1, 2i_0 + 2\rangle$ (that is, the i_0-th element of L_j is of the form $\langle j, 1, 2x + 1\rangle$ for some x, and the i_1-th element of L_j is of the form $\langle j, 1, 2x\rangle$ for some x).

Let $\mathcal{L} = \{L_j \mid j \in N\}$. Note that for all j, if one can determine whether (a) or (b) holds in Proposition 9 for $S = S_j$, $m = m_j$, $\mathbf{M} = \mathbf{M}_j$, then from any element for $X_j - S_j$, one can determine X_j and thus L_j. Note that one can determine whether (a) or (b) holds in Proposition 9 for $S = S_j$, $m = m_j$, $\mathbf{M} = \mathbf{M}_j$, from the i_0-th element of L_j. Similarly, one can determine whether (a) or (b) holds in Proposition 9 for $S = S_j$, $m = m_j$, $\mathbf{M} = \mathbf{M}_j$, from the i_1-th element of L_j. Thus, $\mathcal{L} \in \mathbf{SublangEx}_{\mathcal{A}_1}$ and $\mathcal{L} \in \mathbf{SublangEx}_{\mathcal{A}_2}$.

On the other hand, it follows from the construction of X_j and L_j that $\mathbf{M}_j$ cannot **SublangBc**-identify L_j from a text for X_j. It follows that, $\mathcal{L} \notin \mathbf{UniSublangBc}_{\mathcal{A}_1 \cup \mathcal{A}_2}$. ∎

Corollary 11. *There exists a class $\mathcal{A}$ of samplings such that* $\mathbf{PUniSublangEx}_{\mathcal{A}} - \mathbf{UniSublangBc}_{\mathcal{A}} \neq \emptyset$.

The next proposition establishes a sufficient condition for the existence of a uniform learner on the union of the two sets of samplings when a class is learnable on each of the two sets of samplings separately, provided that all learners have access to samplings A (using an oracle).

Proposition 12. *Let $\mathcal{A}_1, \mathcal{A}_2$ be two sets. Suppose there exists a computable F such that F on any text T for A converges to* 1, *if $A \in \mathcal{A}_1 - \mathcal{A}_2$; F on any text T for A converges to* 2, *if $A \in \mathcal{A}_2 - \mathcal{A}_1$; and F on any text T for A converges to either* 1 *or* 2, *if $A \in \mathcal{A}_2 \cap \mathcal{A}_1$. Then,*
$\mathbf{PUniSublangEx}_{\mathcal{A}_1} \cap \mathbf{PUniSublangEx}_{\mathcal{A}_2} \subseteq \mathbf{PUniSublangEx}_{\mathcal{A}_1 \cup \mathcal{A}_2}$.

Now we turn our attention to the problem of extending learners from large natural sets of samplings to larger sets of samplings. For example, can a learner inferring correct grammars on all recursive samplings be extended to a learner on all recursively enumberable samplings? Or, can a learner on all recursively enumerable samplings be extended to a learner on all infinite samplings? So far, we have been able to establish only negative results. Our first result demonstrates that there is a class of languages uniformly learnable from all infinite recursively enumerable samplings, but not learnable from all infinite samplings, even non-uniformly.

We begin with a number of technical propositions.

Proposition 13. *Let $R = \{x_0, x_1, x_2, \ldots\}$, be any infinite recursive set such that, for all i, $x_{i+1} - x_i \geq i + 2$. Then, there exists a recursively enumerable set $S \supseteq R$ such that Order(R, S) is immune, x_0 is the least element of S, and, for all i,* $\mathrm{card}(S \cap \{x \mid x_i < x < x_{i+1}\}) \leq i + 1$.

Proof. Let $R = \{x_0, x_1, \ldots\}$ be as given in the hypothesis of the proposition. By implicit use of Kleene's recursion theorem [Rog67] one can define $S = W_e$ in stages as follows. Initially, W_e contains all of R, and for all k, sat_k is false and $b_k = 0$. Intuitively, if sat_k is true at the beginning of any stage, then

$$Req_k : W_k \text{ intersects with } \mathrm{Order}(S - R, S)$$

is satisfied at the beginning of stage s, and this holds as long as we do not change membership in W_e for elements $\leq b_k$. We will also enumerate at most $r + 1$ elements x in W_e such that $x_r < x < x_{r+1}$.

Go to stage 0.

Stage s

1. If there exists a $k \leq s$ such that sat_k is currently false and $W_{k,s}$ enumerates an element $w > \{b_r \mid r \leq k\}$ and the w-th element of W_e (as of now) is $> x_k$, then pick the least such k and go to step 2. Otherwise, go to stage $s + 1$.
2. Let z be the w-th element in W_e as of now.

3. If $z = x_{r+1}$ for some r, then insert a new element $(x_r + k + 1)$ in W_e.
4. Set $sat_k = true$ and set $b_k = z$.
5. Set $sat_{k'} = false$, for all $k' > k$.
6. Go to stage $s + 1$.

End stage s

By induction on k, one can show that sat_k eventually takes a fixed value: once $sat_{k'}$ get stabilized for $k' < k$, then sat_k can change at most once, from false to true. Furthermore, if W_k is infinite then sat_k is eventually always true. Thus, Req_k eventually holds for all infintie W_k. Also clearly, the algorithm adds at most $r + 1$ elements in between x_r and x_{r+1} to W_e (see the requirement on the w-th element of W_e being $> x_k$ in step 1). It follows that $S = W_e$ satisfies the requirements of the Proposition. ∎

Proposition 14. *Let R, S be as in Proposition 13. Fix $j \in N$. Let $\{z_0, z_1, \ldots\}$, where $z_0 < z_1 < \ldots$, be an infinite recursive set such that, for all i and for all $e \leq i$, number of elements in $\{\langle j, e+1, x\rangle \mid \langle j, 0, z_i\rangle < \langle j, e+1, x\rangle < \langle j, 0, z_{i+1}\rangle\} \geq i + 1$.*

Then, for $R_j = \{\langle j, 0, z_i\rangle \mid i \in N\}$, there exists an e and a set $S_j \supseteq R_j$ such that

(a) $S_j - R_j$ is an infinite subset of $\{\langle j, r + 1, x\rangle \mid r, x \in N\}$.
(b) for all but finitely many $\langle j, r + 1, x\rangle \in S_j - R_j, r = e$,
(c) $W_e = S_j$, and
(d) $Order(R_j, S_j) = Order(R, S)$.

Proof. Let $R_j = \{z_0, z_1, \ldots\}$ be as in the hypothesis of the proposition. Let $R = \{x_0, x_1, \ldots\}$ and S be as in Proposition 13. Then, by implicit use of Kleene's recursion theorem [Rog67] there exists an e such that W_e may be defined as follows.

W_e contains R_j, $\langle j, 0, z_0\rangle$ is the minimal element of W_e and for each i, W_e contains exactly card$(S \cap \{x \mid x_i < x < x_{i+1}\})$ elements from the set:

(i) $\{\langle j, e + 1, x\rangle \mid \langle j, 0, z_i\rangle < \langle j, e + 1, x\rangle < \langle j, 0, z_{i+1}\rangle\}$, if $i > e$ and
(ii) $\{\langle j, 1, x\rangle \mid \langle j, 0, z_i\rangle < \langle j, 1, x\rangle < \langle j, 0, z_{i+1}\rangle\}$, if $i \leq e$.

Note that, for each i, the gap between $\langle j, 0, z_i\rangle$ and $\langle j, 0, z_{i+1}\rangle$ is large enough to allow the above, and thus, W_e can be so defined. It is easy to verify that $S_j = W_e$ satisfies the requirements of the Proposition. ∎

Proposition 15. *Suppose X is an infinite recursive set and* **M** *is an IIM. Then for all finite $Y \subseteq X$, there exists an infinite recursive Z such that $Y \subseteq Z \subseteq X$, and for an increasing text T for Z, for infinitely many n, $W_{\mathbf{M}(T[n])} \cap X \neq Z$.*

Proof. The proof of $\{L_f \mid f \in \mathcal{R}\}$ not being **TxtBc**-learnable (see [Gol67]), can be easily modified to show this proposition. ∎

Now we can prove the desired result.

Theorem 16. $\mathbf{UniSublangEx}_{REinf} - \mathbf{SublangBc}_{INF} \neq \emptyset$.

Proof. For each j, let $y_0^j < y_1^j < \ldots$ be such that $\{y_0^j, y_1^j, \ldots\}$ is recursive, and for all i and for all $e \leq i$, number of elements in $\{\langle j, e+1, x\rangle \mid \langle j, 0, y_i\rangle < \langle j, e+1, x\rangle < \langle j, 0, y_{i+1}\rangle\} \geq i+1$.

Let R_j be Z as obtained by Proposition 15, when $Y = \emptyset$, $X = X_j = \{\langle j, 0, y_r^j\rangle \mid r \in N\}$ and $\mathbf{M} = \mathbf{M}_j$. Let S_j be as obtained in Proposition 14 for this R_j.

Let $\mathcal{L} = \{S_j \mid j \in N\}$.

Then, S_j witnesses that $\mathcal{L}$ is not $\mathbf{SublangBc}_{\mathrm{Order}(R,S)}$-identified by $\mathbf{M}_j$. Thus, $\mathcal{L} \notin \mathbf{SublangBc}_{\mathrm{Order}(R,S)}$ and, therefore, $\mathcal{L} \notin \mathbf{SublangBc}_{\mathrm{INF}}$.

On the other hand, $\mathcal{L}$ is clearly in $\mathbf{UniSublangEx}_{\mathrm{REinf}}$, as for any $S_j \in \mathcal{L}$, for any Y such that $\mathrm{Order}(Y, S_j)$ is an infinite r.e. set, we have that Y contains infinitely many elements in $S_j - R_j$. It follows that Y contains infinitely many elements of the form $\langle j, r+1, x\rangle$, and all but finitely many such r are equal to some grammar e for S_j. ∎

The next result shows that the learnability of a class from all simple recursively enumerable samplings does not imply its learnability from all recursively enumerable samplings.

Theorem 17. *Let $\mathcal{A}_1 = \{A \mid A$ is an infinite simple set $\}$. Let $A_2 = \{2y \mid y \in N\}$.*

Then, $\mathbf{UniSublangEx}_{\mathcal{A}_1} - \mathbf{SublangBc}_{A_2} \neq \emptyset$.

Proof. For each j, let $y_0^j < y_1^j < \ldots$ be such that $\{y_0^j, y_1^j, \ldots\}$ is recursive, and for all i and for all $e \leq i$, number of elements in $\{\langle j, e+1, x\rangle \mid \langle j, 0, y_i^j\rangle < \langle j, e+1, x\rangle < \langle j, 0, y_{i+1}^j\rangle\} \geq 1$.

Let $R_j = Z$, as in Proposition 15 for $\mathbf{M} = \mathbf{M}_j$, $X = X_j = \{\langle j, 0, y_r^j\rangle \mid r \in N\}$ and $Y = \emptyset$.

By implicit use of Kleene's recursion theorem [Rog67], there exists an e such that $W_e = S_j$ satisfies (i) $S_j - R_j \subseteq \{\langle j, y+1, x\rangle \mid x, y \in N\}$, (ii) $\mathrm{Order}(R_j, S_j) = \{2y \mid y \in N\}$, and (iii) for all but finitely many $\langle j, y+1, x\rangle \in S_j$, $y = e$.

Let $\mathcal{L} = \{S_j \mid j \in N\}$. It is easy to verify that $\mathcal{L} \in \mathbf{UniSublangEx}_{\mathcal{A}_1}$, as any simple set intersects with $\{2y+1 \mid y \in N\}$ infinitely often. Also, by Proposition 15, it follows that $\mathbf{M}_j$ does not $\mathbf{SublangBc}_{\mathrm{Order}(R_j,S_j)}$-identify S_j. As $\mathrm{Order}(R_j, S_j)$ is $\{2y \mid y \in N\}$, we have that $\mathcal{L} \notin \mathbf{SublangBc}_{A_2}$. ∎

Our next result shows that, even learnability of a class from all samplings but subsets of one sampling does not imply its learnability from all samplings.

Theorem 18. *Suppose A is an infinite recursive set different from N. Then,* $\mathbf{SublangEx}_{\mathcal{P}(N)-\mathcal{P}(A)} - \mathbf{SublangBc}_A \neq \emptyset$.

Proof. Without loss of generality assume $0 \in A$. (Proof can be easily modified for any other element in A, by considering appropriate modification of SD and AZext).

Let $p_0, p_1, p_2, \ldots$ be a recursive sequence of increasing prime numbers. Let $A = \{x_0, x_1, \ldots\}$, where $x_0 < x_1 < \ldots$.
For $g \in \mathrm{SD}$, let $h_g(x) = p_0^{g(0)} \cdot p_{x+1}^{2g(x)} \cdot \mathit{\Pi}_{y<x}(p_{y+1}^{2g(y)+1})$.

For $g \in$ AZext, let

$h_g(x) = p_0^{g(0)} \cdot p_{x+1}^{2g(x)} \cdot \Pi_{y<x}(p_{y+1}^{2g(y)+1})$, if $x \in A$, and

$h_g(x) = p_0^{g(0)} \cdot p_{x+1}^{2g(x)+1} \cdot \Pi_{y<x}(p_{y+1}^{2g(y)+1})$, if $x \notin A$.

Note that h_g is an increasing function in both the above cases (thus the r-th element of L_{h_g} is $\langle r, h_g(r)\rangle$). Also, every $h_g(x)$ gives away the value of $g(0)$. Furthermore, the only difference in the two cases (of $g \in$ SD or $g \in$ AZext) is how $h_g(x)$ is defined for $x \notin A$.

Let $\mathcal{L} = \{L_{h_g} \mid g \in \text{SD} \cup \text{AZext}\}$.

It is easy to verify that $\mathcal{L} \in \mathbf{SublangEx}_{\mathcal{P}(N)-\mathcal{P}(A)}$, as from any input element $\langle x, h_g(x)\rangle$, for $x \notin A$, one can determine $g(0)$ and whether $g \in$ SD or $g \in$ AZext.

Now for $g, g' \in \text{SD} \cup \text{AZext}$,

(i) one can effectively convert an infinite subgraph for h_g into a graph for g,

(ii) one can effectively convert a graph for g into a graph for h_g restricted to the domain A,

(iii) if $g \neq g'$, then $h_g(x) = h_{g'}(x)$ only for finitely many x.

Thus, $\mathcal{L} \in \mathbf{SublangBc}_A$ implies that $\{L_f \mid f \in \text{SD} \cup \text{AZext}\} \in \mathbf{TxtBc}$, contradicting Theorem 5. It follows that $\mathcal{L} \notin \mathbf{SublangBc}_A$. ∎

A corollary from the above theorem shows that **Ex**-learners from all positive data may sometimes be more powerful than any **Bc**-learner from an A-sampling defined by any infinite recursive sampling $A \neq N$.

Corollary 19. $\mathbf{TxtEx} - \mathbf{SublangBc}_A \neq \emptyset$, *for all infinite recursive* $A \neq N$.

On the other hand, uniform **Bc**-learners from all infinite samplings can sometimes be more powerful than **Ex**-learners from all positive data.

Theorem 20. $\mathbf{UniSublangBc}_{INF} - \mathbf{TxtEx} \neq \emptyset$.

Proof. Let $\mathcal{C} = \{f \mid (\forall^\infty x)[\varphi_{f(x)} = f]\}$. Let $\mathcal{L} = \{L_f \mid f \in \mathcal{C}\}$. Then, it is easy to verify that $\mathcal{L} \in \mathbf{UniSublangBc}_{\text{INF}}$. On the other hand [CS83] showed that $\mathcal{L} \notin \mathbf{TxtEx}$ ([CS83] actually showed this for function learning, which implies the result for language learning). ∎

5 Conclusion

This paper can be viewed as the first step in the study of learnability of various projections of target languages. Firstly, as we mentioned in the Introduction, several different formalizations of the concept of samplings are possible, and the notion of projection itself can be formalized in several ways. For example, a projection of a language may be defined as the set of examples satisfying some predicate — say, when one considers a language of strings over the alphabet $\{a, b, c\}$, a projection may be its sublanguage consisting of all strings over the alphabet $\{a, b\}$.

Yet, even within the framework of formalization suggested in our paper, many interesting questions remain open. We obtained some non-union type results, however, we have not been able to establish more general non-union results for large (and natural, say, containing all recursive samplings) sets of samplings, or, alternatively, find situations when learnability on two different sets of samplings implies uniform learnability on the union of the given sets. The problem of expanding learnability from all samplings from some natural large class to another class of samplings embracing it is also far from being fully explored. For example, we have not been able to find out if learnability from all recursive infinite samplings implies learnability on all infinite recursively enumerable samplings.

References

[BB75] Blum, L., Blum, M.: Toward a mathematical theory of inductive inference. Information and Control 28, 125–155 (1975)

[Blu67] Blum, M.: A machine-independent theory of the complexity of recursive functions. Journal of the ACM 14, 322–336 (1967)

[CL82] Case, J., Lynes, C.: Machine inductive inference and language identification. In: Nielsen, M., Schmidt, E.M. (eds.) ICALP 1982. LNCS, vol. 140, pp. 107–115. Springer, Heidelberg (1982)

[CS83] Case, J., Smith, C.: Comparison of identification criteria for machine inductive inference. Theoretical Computer Science 25, 193–220 (1983)

[Fre74] Freivalds, R.: Uniform and non-uniform predictability. In: Theory of Algorithms and Programs, vol. 1, pp. 89–100. Latvian State University, Riga (1974)

[Ful90] Fulk, M.: Prudence and other conditions on formal language learning. Information and Computation 85, 1–11 (1990)

[Gol67] Gold, E.M.: Language identification in the limit. Information and Control 10, 447–474 (1967)

[HU79] Hopcroft, J., Ullman, J.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley, Reading (1979)

[JK08] Jain, S., Kinber, E.: Learning and extending sublanguages. Theoretical Computer Science A 397(1-3), 233–246 (2008); Special Issue on Forty Years of Inductive Inference. Dedicated to the 60th Birthday of Rolf Wiehagen

[OSW86] Osherson, D., Stob, M., Weinstein, S.: Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists. MIT Press, Cambridge (1986)

[OW82] Osherson, D., Weinstein, S.: Criteria of language learning. Information and Control 52, 123–138 (1982)

[Rog67] Rogers, H.: Theory of Recursive Functions and Effective Computability. McGraw-Hill, New York (1967); Reprinted by MIT Press in 1987

[Tra73] Trakhtenbrot, B.A.: Frequency computations. In: Proceedings of Steklov Mathematical Institute, vol. 133, pp. 221–232. Academy of Sciences of USSR (1973) (in Russian)

Optimality Issues of Universal Greedy Agents with Static Priors

Laurent Orseau

UMR AgroParisTech 518 / INRA
16 rue Claude Bernard, 75005 Paris, France
laurent.orseau@agroparistech.fr
http://www.agroparistech.fr/-Laurent-ORSEAU-.html

Abstract. Finding the universal artificial intelligent agent is the old dream of AI scientists. Solomonoff Induction was one big step towards this, giving a universal solution to the general problem of Sequence Prediction, by defining a universal prior distribution. Hutter defined AIXI which extends the latter to the Reinforcement Learning framework, where almost all if not all AI problems can be formulated. However, new difficulties arise, because the agent is now active, whereas it is only passive in the Sequence Prediction case. This makes proving AIXI's optimality difficult. In fact, we prove that the current definition of AIXI can sometimes be only suboptimal in a certain sense, and we generalize this result to infinite horizon agents and to any static prior distribution.

Keywords: AIXI, Universal Artificial Intelligence, Solomonoff Induction, Reinforcement Learning.

1 Introduction

In [2], Hutter developed what could be called the *optimally rational agent* AIXI. By merging the very general framework of Reinforcement Learning [12] with the universal sequence predictor defined by Solomonoff Induction [11], AIXI is supposed to optimally solve any problem, at least when the solution is computable. Hutter even proved that AIXI is Pareto optimal [3], i.e. that no other agent can do better in at least one environment and do at least as well in all other environments. AIXI is parameter-free, apart from the horizon function, which is a (usually decreasing) weighting function of the future time steps.

One important problem is that, like Solomonoff Induction, the AIXI model is not computable. But since it is an upper-bound of intelligent agents, this can be used as a lighthouse for defining computable agents. With this in mind, a Monte-Carlo version of AIXI was created [13], showing that MC-AIXI could solve several different problems with the exact same setting. However, the authors had to augment MC-AIXI with an exploration strategy to obtain better results. They state:

> It is worth noting that, in principle, the AIXI agent does not need to explore according to any heuristic policy. This is since the value of information, in terms of expected future reward, is implicitly captured in

© Springer-Verlag Berlin Heidelberg 2010

> the expectimax operation [...]. Theoretically, ignoring all computational concerns, it is sufficient just to choose a large horizon and pick the action with the highest expected value at each timestep. Unfortunately, this result does not carry over to our approximate AIXI agent.

The following shows that this is a concern with AIXI in general, and not specifically of the scaled-down version. It is the transposition of the exploration/exploitation dilemma [12] in the universal setting of computable environments.

Intuitively, suppose that at each time step, the agent AIXI is given a choice between actions a and b. At first, doing a will give the agent a reward, whereas b returns a punishment. As time goes on, AIXI infers with higher and higher probability that this setting will remain as is forever, although it may still try action b from time to time. But, after some sufficiently long time, the inferred probability that b is useless is so high that AIXI considers it a waste of time (and resources) to try it. In one sense (that of Pareto optimality), it is in fact right. But if after such long time b eventually starts to return an even higher reward than a, AIXI will never find it out. In such a setting, AIXI could do better at each time step, but does not: its number of suboptimal choices tends to infinity.

First, some notation is introduced, followed by the definition of how an agent that has full knowledge about the real distribution of the possible environments can act optimally. The next section defines agents that do not have knowledge about that distribution but use the interaction history with the environment to act in the most rational way, given some prior distribution. Then the special case of the universal agent AIXI is defined. The following section proves that no such agent can sufficiently explore to converge to the current optimal policy. We conclude by some remarks on how to solve this issue.

2 Notation

The notation is similar to that in [5].

X and Y are the vocabulary of the interaction between the agent and the environment. An action/output $y_k \in Y$ of the agent at step k is also the input of the environment. The output $x_k \in X$ of the environment is also the input of the agent, and is composed of a reward part $r_k = r(x_k)$ and of some information $o_k = o(x_k)$. One interaction cycle is noted by $\underline{yx}_k = y_k x_k$.

A string $x_1 x_2 \dots x_n$ is a succession of $x_t \in X$, ordered by t. We write $x_{n:m} = x_n x_{n+1} \dots x_m$, and also $x_{<k} = x_{1:k-1}$ and similarly for y and $\underline{yx}$. The empty string is denoted by ϵ.

At each new step k, the agent outputs an action y_k depending on history $\underline{yx}_{<k}$, then the environment outputs x_k depending on $\underline{yx}_{<k} y_k$, and then the next step $k+1$ begins. The string $\underline{yx}_{1:k} = y_1 x_1 y_2 x_2 \dots y_k x_k$ is the interaction string between the agent and the environment.

The size of a string is denoted $|\underline{yx}_{1:k}| = |x_{1:k}| = k$. Note that this is different from the length $l(x)$ of a string which is defined as the number of bits used to encode x on the device under consideration, such as a Turing machine.

3 AIMU

The definition of the agent AIMU can be reduced to the following question: what is the best action to do at current step k, given the history $y\!x_{<k}$ of inputs and outputs, and the knowledge of the true/generating probability distribution μ of the environment?

In the following, we provide a generalization of AIMU and AIXI to general horizon functions. This is directly based on [5, p.8-10], where the binary horizon function m_k is replaced by a general function $w_{k,t}$. See also [3] for an iterative definition to general horizon functions. We use the functional definition of AIMU, modifying it slightly to be able to compare the values of the different choices for finding the next action y_k.

First, we recall the functional definition of AIMU.

Let Q_k be the set of programs/environments as partial functions $q : Y^* \to X^*$, defined on *chronological Turing Machines* [5, p.8], that are consistent with the history $y\!x_{<k}$:

$$Q_k = \{q : q(y_{<k}) = x_{<k}\}.$$

Also define P_k the set of programs/policies of the agent that are consistent with the history $y\!x_{<k}$:

$$P_k = \{p : \exists y_k \in Y : p(x_{<k}) = y_{<k}y_k\}.$$

In the remainder, the sets Q and P contain all environments and all policies, respectively.

For a given horizon function $w_{k,t}$ (a temporal weight), the value of a given policy p in a given environment q at step k is:

$$V_k^{qp} = \sum_{t=k}^{\infty} w_{k,t} \cdot r_t \tag{1}$$

with $r_t = r(x_t)$, where x_t is defined by $x_{<t}x_t = q(y_{1:t})$, and y_t is defined by $y_{<t}y_t = p(x_{<t})$. Various possibilities for $w_{k,t}$ are discussed in the next section. In the remainder, without loss of generality, we always consider that $r_t \in [0; 1]$.

The value of a given policy p on average in all environments is:

$$V_k^{\mu p} = \sum_{q \in Q_k} \mu(q) \cdot V_k^{qp} \tag{2}$$

where $\mu(q)$ is the prior probability of environment q, and the normalizing denominator $\sum_{q \in Q_k} \mu(q)$ is omitted as it plays no role in the comparison of the different policies.

The best policy, that the agent must follow (only) in step k to maximize the expected reward is:

$$p_k^{\mu *} = \arg\max_{p \in P_k} V_k^{\mu p}.$$

If there are more than one best policy p, we take the first one in lexicographical order.

Finally, the action y_k^μ chosen by the agent at step k is thus defined by:

$$p_k^{\mu *}(x_{<k}) = y_{<k} y_k^\mu. \tag{3}$$

Now, in order to define the expected value of each potential y_k, we split P_k into $|Y|$ sets P_k^i, which are the sets of programs in P_k that output y^i at step k:

$$\forall i, y^i \in Y : P_k^i = \{p : p(x_{<k}) = y_{<k} y^i\}.$$

We have then:

$$P_k = \bigcup_{i : y^i \in Y} P_k^i.$$

For a given action $y_k = y^i$, the best policy is:

$$p_k^{\mu * i} = \arg\max_{p \in P_k^i} V_k^{\mu p}.$$

The optimal policy is then the best among the best policies for each y^i:

$$p_k^{\mu *} = \arg\max_{p_k^{\mu * i} : y^i \in Y} V_k^{\mu p_k^{\mu * i}}.$$

The action y_k^μ that the agent chooses can be redefined as:

$$\forall i, y^i \in Y : y_k^\mu = y^i \Leftrightarrow p_k^{\mu *} = p_k^{\mu * i}.$$

We can now define the value of a given action:

$$\forall i, y^i \in Y : V_k^\mu(y^i) = V_k^{\mu p_k^{\mu * i}} = \max_{p \in P_k^i} V_k^{\mu p}. \tag{4}$$

Finally, actions can now be compared to determine the best one:

$$\forall i, y^i \in Y : \big(\forall j, j \neq i : V_k^\mu(y^i) > V_k^\mu(y^j)\big) \Rightarrow y_k^\mu = y^i. \tag{5}$$

3.1 Horizon Functions

In this section we give some simple examples of horizon functions $w_{k,t}$, which weight the future time steps and define "how far" the agent can foresee. See [4,5] for additional discussion.

The horizon function must fulfill some properties:

$$\forall t \geq k,\ w_{k,t} \geq 0,$$

$$\lim_{t \to \infty} \sum_{t=k}^{\infty} w_{k,t} < \infty. \tag{6}$$

Equation (6) avoids convergence problems in (1).

Fixed Lifetime. If the agent has a fixed initial lifetime T,

$$w_{k,t} = \begin{cases} 1 & \text{if } t \leq T \\ 0 & \text{otherwise} \end{cases} . \tag{7}$$

Constant Horizon. With a constant horizon, the agent can see m steps ahead:

$$w_{k,t} = \begin{cases} 1 & \text{if } (t-k) < m \\ 0 & \text{otherwise} \end{cases} , \ m > 0. \tag{8}$$

Variable Horizon. With a variable horizon m_k, the agent can see a number of steps ahead, depending on its own age k:

$$w_{k,t} = \begin{cases} 1 & \text{if } (t-k) < m_k \\ 0 & \text{otherwise} \end{cases} , \ m_k > 0.$$

Hyperbolic/Harmonic Discounting. It seems that humans and animals have a hyperbolic discounting horizon [1]:

$$w_{k,t} = \frac{1}{1 + \alpha \cdot i}, \ i = t - k, \ \alpha > 0.$$

However, it is not clear whether the hyperbolic function is built in the reward system or if it is an epi-phenomenon, due for example to the growing (with time) uncertainty of the future.

Note that this does not satisfy (6), so we will not consider that case in the remainder. In order to ensure convergence, we can generalize to over-harmonic discounting:

$$w_{k,t} = \frac{1}{(1 + \alpha \cdot i)^\beta}, \ i = t - k, \ \alpha > 0, \ \beta > 1.$$

Exponential/Geometric Discounting. The exponential/geometric discounting is often used in Reinforcement Learning [12]:

$$w_{k,t} = \gamma^i, \ i = k - t, \ \gamma > 1.$$

In contrast to harmonic discounting, it has the property to be time independent: If one is offered \$50 today or \$100 tomorrow, then for $\gamma = 2 : 2^0 * 50 = 2^{-1} * 100$, the expected value is the same, and it remains so if the same problem is given in a distant future, e.g. if one is offered \$50 in 100 days or \$100 in 101 days: $2^{-100} * 50 = 2^{-101} * 100$.

4 Universal Agents

Now we turn to agents that have no initial specific knowledge about the true distribution μ of the environment. They have to gather information, and thus acquire knowledge, by interacting with the environment. After sufficient interaction steps, we would like the agent to act as well as possible.

For a given prior ρ about the *a priori* distribution of all environments/programs, we can define an agent AIρ by replacing μ by ρ in equations (2)-(5). The prior ρ defines the initial or model-free or μ-independent knowledge of the agent about the prior probabilities of environments. ρ does not need to be a probability distribution, but can be only a semi-measure that satisfies:

$$\sum_{q \in Q} \rho(q) < \infty \tag{9}$$

so that equation (1) does not diverge, and of course: $\forall q, \rho(q) \geq 0$.

Definition 1. *For a given agent AIρ with a prior distribution ρ on programs/environments, AIρ is said to be a* universal agent *iff:*

$$\forall q \in Q : \rho(q) > 0. \tag{10}$$

A universal agent does not discard any environment possibility *a priori*.

AIMU is in general not universal as its knowledge allows it to give a null probability to most environments.

Note that in this paper we consider only agents with a *static* prior, i.e. $\rho(q)$ does not change with the history (except if the program is not consistent with the history). Other, dynamic priors, such as the Speed Prior S [9], have different properties that are not investigated here.

5 Solomonoff Induction and AIXI

Suppose a predictor is given a sequence of only 0s and 1s. What is the best guess it could do about the next symbol, knowing only that this sequence is generated by some computable but unknown program?

Solomonoff Induction [11] addresses this exact problem. The predictor knows there is an infinity of possible environments that are compatible with the given sequence. Are all these environments equally probable? Or is there an *a priori* order upon them? Following both Occam's Razor Principle, which roughly states that the best hypothesis is the simplest one, and the Epicurean Principle, which recommends to keep all consistent hypotheses, Solmonoff's answer is to give a higher probability to simpler environments, and that all consistent environments should have a positive probability. He defines the notion of simplicity as the length $l(q)$ in bits of a given program q on a prefix universal Turing machine: the shortest programs are the simplest ones. For a sequence x, its simplicity measure is defined by:

$$M(x) = \sum_{q:q \text{ produces } x*} 2^{-l(q)}$$

where $x*$ is any string that begins with the the string x.

M is in fact only a semi-measure and satisfies Kraft's inequality [6] on a prefix universal Turing machine with a binary alphabet: $\sum_{q \in Q} 2^{-l(q)} \leq 1$.

Solomonoff showed that M converges to the true generator μ when the size of the sequence grows [10].

Based on this work, Hutter extended M to the Reinforcement Learning framework [2,5,4], where the agent is no longer a passive predictor, but an active agent which outputs actions that the environment may take into account. He also generalized the binary alphabet to a vocabulary of (words of) inputs X and outputs Y. Within this setting, the new semi-measure ξ is essentially equivalent to M. For a program q that is consistent with some history $y\!x_{<k}$:

$$\xi(q) = 2^{-l(q)}.$$

Taking $\rho = \xi$ in the previous section gives the definition of the agent AIXI.

Since ξ satisfies Kraft's inequality, it is a semi-measure, and since the definition of ξ implies (10), AIXI is a universal agent. It can even be thought of as *the* universal agent, or the *optimally rational agent* ; In [5, p.23], the author states:

> we expect no other model to converge faster to AIμ by analogy to S[equence]P[rediction]

Hutter has proved that AIXI is Pareto optimal[3], which could make one confident that it always ends up in finding the optimal policy for a given history, but we shall see that this is not true.

The Good Enough and Not Good Enough Effects:

Theorem 1. *For any history $y\!x_{<k}$, there exist programs that are consistent with the history, and that predict a constant reward R, $0 \leq R \leq 1$:*

$$\forall y\!x_{<k}, \forall R, 0 \leq R \leq 1, \exists q \in Q_k, \forall n \geq k : r(x_n) = R \textit{ with } q(y_{1:n}) = x_{1:n}.$$

Proof. By construction.

This means that even if the most probable environments predict a null reward for all actions, other lower probability programs will predict high rewards. As the agent is optimistic about other possible environments, it will explore those possibilities. We call this the *not good enough effect.*

There is an opposite *good enough effect* that can make the agent lazy when the most probable environments predict a constant high reward for a given action. The other environments will not have a sufficiently high cumulative weight to make the agent try other actions. The latter may then get stuck in some local "good enough" maximum.

We are now going to develop the good enough effect in details.

6 Asymptotic Non-learnability

Hutter mentions the simple Heaven&Hell environment [5, p.26], where the agent cannot converge to AIMU. At the first time step, the agent must choose between left and right, one direction leading to Heaven and the other to Hell. Then, for all the following time steps, the agent has no other choice than staying where it is, and receiving a constant reward (+1) in Heaven or a constant punishment (0) in Hell. An uninformed agent cannot know in what direction Heaven is, whereas AIMU does know. The difference of total rewards gathered by AIXI (or any universal AIρ) and AIMU can then tend to infinity. This is, in fact, not a problem concerning AIXI directly, since no agent that has no initial μ-specific information can do better.

As stated by Hutter, we should then consider other, more plausible criteria: the performance of the agent should then be compared with that of AIMU, but the history of AIMU should be the same as that of the agent, in order to compare the *mistakes* (suboptimal choices) of the agent, and not their consequences. He refers to this as *asymptotic learnability* and states [5, p.28]:

> We claim that AIξ (for $m_k \to \infty$) can asymptotically learn every problem μ of relevance, i.e. AIξ is asymptotically optimal.

The main claim of this paper is that, contrary to Hutter's expectations, AIXI cannot asymptotically learn every problem μ of relevance.

Intuitively, AIXI, in its current definition, as other universal agents AIρ choosing their actions with (3), is too greedy, and is in fact optimally greedy: exploration can sometimes lead to an expected average loss of rewards. It prevents the agent from sufficiently exploring the environment to find better rewards, even in some simple environments.

Note that since AIXI is Pareto optimal, this means that improving the convergence to AIMU will *not* improve the average performance of the agent.

There exists at least one class of environments where AIρ, and more specifically AIXI, given some particular history, does not explore sufficiently to find the optimal policy: ρ does not always converge toward the true distribution μ, and can get stuck in local maxima.

We use Hutter's asymptotic learnability and extend it to general horizon functions. Let:

$$D_k^{\mu\rho} = \frac{(V_k^{\mu *} - V_k^{\mu p_k^\rho})}{\sum_{t=k}^{\infty} w_{k,t}}$$

be the μ-expected policy value difference between AIMU and AIρ at step k, where $V_k^{\mu *}$ is the expected value by the optimal policy in μ, and $V_k^{\mu p_k^\rho}$ is the expected value computed by AIMU with the best current policy of AIρ. There is no need to consider the case of a null denominator, since in such case the agent has no longer any rational choice to make, since no future step has the least importance.

Definition 2. *An agent AIρ is said to asymptotically learn a problem μ iff:*

$$\forall y\!\!\!/x_{<i}, \lim_{k\to\infty} D_k^{\mu\rho} = 0 \tag{11}$$

when history $y\!\!\!/x_{i:k}$ is generated by AIρ.

This simply means that the optimal policy of AIρ converges to the optimal policy of AIMU when they have the same growing history.

Theorem 2. *For any universal agent AIρ, there exists at least one computable environment μ where AIρ, given some initial history $y\!\!\!/x_{<i}$, does not asymptotically learn μ.*

In the following subsections, we give constructive proofs, first for the case where the agent has a constant horizon, and then for more general horizon functions.

6.1 Proof for Bounded Horizons

We set $X = [0;1], r(x_t) = x_t$, $Y = \{a_0, a_1\}$, and we take $w_{k,t}$ as in (8). The agent has a constant horizon m, such that it can foresee the future up to step $k + m - 1$.

Define environment/program q_1 as follows:

$$q_1 : x_k = r_k = \begin{cases} R & \text{if } y_k = a_1 \\ 0 & \text{if } y_k = a_0 \end{cases}$$

with $0 < R < 1$.

Definition 3. *A program q_a is said to be H-equivalent to a program q_b for a history $H = y\!\!\!/x_{<k}$ iff:*

$$\forall n \geq k, \big(\nexists y_{k:n} : q_a(y_{1:n}) \neq q_b(y_{1:n})\big) \wedge q_a(y_{<k}) = q_b(y_{<k}) = x_{<k}.$$

This means that no future sequence can distinguish q_a from q_b. Note that there might have been a different (past) history that would have distinguished them.

Then run pseudo-algorithm 1.1, taking $c_1 = \frac{R}{m}$ (note that the horizon $m = 0$ does not make sense), to find a history H, the set Q_1 of high probability environments that are H-equivalent to q_1, and the set $Q_{\overline{1}}$ of low probability environments.

At the end of this (uncomputable) pseudo-algorithm, we have a history H that "discards" all the most probable programs that are not H-equivalent to q_1, so that the total weight of the remaining ones[1] in $Q_{\overline{1}}$ is lower than a fraction of the ones in Q_1. We do not need this algorithm to be computable, since we only need to show that the history H (which is finite) exists.

[1] $Q_{\overline{1}}$ may also contain some low probability programs H-equivalent to q_1 or not even consistent with H, but this has no influence on the remainder.

Listing 1.1. Generate history H that discards the most probable programs that are not H-equivalent to q_1

```
Inputs: a measure or semi−measure $\rho$, a constant $c_1 : 0 < c_1 \leq 1$
Outputs: a history $H$, a set $Q_1$ of programs $H$−equivalent to $q_1$,
  a set $Q_{\overline{1}}$ of low probability programs.
// Start with empty history
$H := \epsilon$
// Initialize $Q_1$ with the empty set:
$Q_1 := \phi$
// Initialize $Q_{\overline{1}}$ to the set of all programs:
$Q_{\overline{1}} := Q$
// Enumerate all programs in decreasing probability,
// and discard the ones that are not H−equivalent to $q_1$:
Repeat
  // Take the next most probable program
  // (or the first one in lexicographical order) in $Q_{\overline{1}}$:
  $q_a := \arg\max_{q \in Q_{\overline{1}}} \rho(q)$
  // Remove $q_a$ from $Q_{\overline{1}}$:
  $Q_{\overline{1}} := Q_{\overline{1}} \setminus \{q_a\}$
  // Look for a sequence of actions, after history $H = yx_{<i}$
  // where at least one output of $q_a$
  // differs from one output of $q_1$:
  Let $y_{i:j}$ be such that $q_a(y_{1:j}) \neq q_1(y_{1:j})$, $i = |H| + 1, j \geq i$
  If $y_{i:j}$ exists Then
    // Either $q_a$ produces another output than
    // the output of $q_1$,
    // or $q_a$ halts prematurely,
    // or $q_a$ does not halt and cannot produce some output $y_t$.
    // Then $q_a$ is not H−equivalent to $q_1$.
    // Discard it, by extending the history appropriately:
    $H := Hy_{i:j}$
  Else
    // $q_a$ is H−equivalent to $q_1$, add it to $Q_1$:
    $Q_1 := Q_1 \cup \{q_a\}$
  EndIf
Until $\sum_{q \in Q_{\overline{1}}} \rho(q) < c_1 \sum_{q \in Q_1} \rho(q)$
```

The stopping condition of the loop is always satisfied at some point, since Q_1 and thus $\sum_{q \in Q_1} \rho(q)$ always grow, $Q_{\overline{1}}$ always decreases in size, and $\sum_{q \in Q_{\overline{1}}} \rho(q)$ tends to 0 because we always remove the program with the highest ρ-value.

Define environment q_2 as follows:

$$q_2 : x_k = r_k = \begin{cases} R & \text{if } y_k = a_1 \\ 0 & \text{if } y_k = a_0 \text{ and } k \leq |H| \\ 1 & \text{if } y_k = a_0 \text{ and } k > |H| \end{cases}.$$

By construction, q_2 is consistent with q_1 for the first $|H|$ steps and has not been discarded by the history H. So we have $q_2 \in Q_{\overline{1}}$.

Now, run the agent in the true environment $\mu \equiv q_2$ with history $y\!x_{<k} = H$. Following equation (4), we can compute bounds for the expected values for the two actions for step k.

If the agent chooses action a_1 given history $y\!x_{<k} = H$ then it can guarantee a minimal payoff by choosing action a_1 for all the following time steps and thus receive at least $m \cdot R$ rewards in each environment H-equivalent to q_1:

$$V_k^\rho(a_1) \geq \sum_{q \in Q_1} \rho(q) \cdot m \cdot R. \tag{12}$$

Choosing action a_0 may at most give a constant reward of 1 for all programs not H-equivalent to q_1, and for the H-equivalent ones, no reward for the first step (a_0) and then a constant reward of R if the agent always chooses action a_1 afterwards. Still following equation (4) but splitting Q into the sets[2] Q_1 and $Q_{\overline{1}}$, the value of action a_0 is thus bounded by:

$$V_k^\rho(a_0) \leq \sum_{q \in Q_{\overline{1}}} \rho(q) \cdot m \cdot 1 + \sum_{q \in Q_1} \rho(q) \cdot (m-1) \cdot R. \tag{13}$$

Then we have:

$$V_k^\rho(a_1) > V_k^\rho(a_0) \Longleftarrow \sum_{q \in Q_{\overline{1}}} \rho(q) < \frac{R}{m} \sum_{q \in Q_1} \rho(q) \tag{14}$$

which r.h.s. is true since H has been generated by the listing 1.1, if we take $c_1 = \frac{R}{m}$. Therefore, from (5), the agent chooses $y_k^\rho = a_1$.

We must show now that this remains true for all the following steps.

The agent then receives $x_k = r_k = R$, which is still consistent with q_1, so no program $H-$equivalent to q_1 is discarded:

$$\sum_{q \in Q_1} \rho(q) = \sum_{\substack{q\,:\,q \in Q_1,\\ q(y_{<k}a_1) = x_{<k}R}} \rho(q)$$

but it is possible that some program that is not $H-$equivalent to q_1 is removed:

$$\sum_{q \in Q_{\overline{1}}} \rho(q) \geq \sum_{\substack{q\,:\,q \in Q_{\overline{1}},\\ q(y_{<k}a_1) = x_{<k}R}} \rho(q).$$

This means that at step $k+1$, $V_{k+1}^\rho(a_1) = V_k^\rho(a_1)$ and $V_{k+1}^\rho(a_0) \leq V_k^\rho(a_0)$ and thus equations (12), (13) and (14) are still satisfied with the new history

[2] Note that $Q_1 \cup Q_{\overline{1}}$ possesses at least all the programs compatible with history H, so no q is missing.

$H := Ha_1R$. Therefore, by recurrence, from now on the agent will always choose a_1 and will never try action a_0.

The optimal policy of AIρ is to always choose a_1 whereas for AIMU it is to always choose a_0. From equation (11), the expected difference of values computed by AIMU at any further step k is $D_k^{\mu\rho} = (1 - R)$.

Therefore, AIρ does not asymptotically learn the computable problem q_2. □

6.2 Proof Extension for General Horizons

The previous proof only holds for agents that have a constant horizon. But what if m is not bounded, but rather depends on k? What if the horizon function follows a geometric progression?

The problem with the previous proof is that if m is not bounded anymore but is replaced by some m_k and depends on the current time step, then the recurrence step does not hold: at $k+1$, if m_k grows, then R/m_k decreases, thus the value of the r.h.s. of the stopping condition in listing 1.1 decreases, and the r.h.s of equation (14) may not be satisfied anymore.

In order to solve this problem, we need to modify q_1 as follows: If the agent once chooses a_0, then a_1 will not return a reward for the next N_k time steps, where N_k must be defined according to $w_{k,t}$ as will be made clear later. Let B be the set of "banned" steps where a_1 returns 0:

$$\forall k,\ y_k = a_0 \quad \Rightarrow \quad \forall i,\ k < i \leq k + N_k,\ i \in B.$$

We redefine q_1 as follows:

$$q_1 : x_k = r_k = \begin{cases} R & \text{if } y_k = a_1 \text{ and } k \notin B \\ 0 & \text{if } y_k = a_1 \text{ and } k \in B \\ 0 & \text{if } y_k = a_0 \end{cases}.$$

We also need to modify q_2 to make it consistent with H and the new definition of q_1:

$$q_2 : x_k = r_k = \begin{cases} R & \text{if } y_k = a_1 \text{ and } k \notin B \\ 0 & \text{if } y_k = a_1 \text{ and } k \in B \\ 0 & \text{if } y_k = a_0 \ \text{ and } k \leq |H| \\ 1 & \text{if } y_k = a_0 \ \text{ and } k > |H| \end{cases}.$$

We now need to find the new value of c_1 for the listing 1.1, so that $V_k^\rho(a_1) > V_k^\rho(a_0)$. This is the aim of what follows.

After running the pseudo-algorithm 1.1, we need to increase the history with N_k steps of a_1 (with $k = |H| + 1$), to make sure the agent is not in a banned step after H, so that it can then choose a_1 and receive a reward:

$$H := H \underbrace{a_1 \ldots a_1}_{N_k \text{ times}}.$$

In this new setting, the values of the actions at the new step k, for $y\!x_{<k} = H$, are:

$$V_k^\rho(a_1) \geq \sum_{q \in Q_1} \rho(q) \sum_{t=k}^{\infty} w_{k,t} \cdot R. \tag{15}$$

$$V_k^\rho(a_0) \leq \sum_{q \in Q_{\overline{1}}} \rho(q) \sum_{t=k}^{\infty} w_{k,t} \cdot 1 \; + \sum_{q \in Q_1} \rho(q) \sum_{t=k+N_k+1}^{\infty} w_{k,t} \cdot R. \tag{16}$$

Then for the agent to choose a_1:

$$\begin{gathered} V_k^\rho(a_1) > V_k^\rho(a_0) \\ \Longleftarrow \\ \sum_{q \in Q_{\overline{1}}} \rho(q) < \frac{\sum_{t=k}^{k+N_k} w_{k,t}}{\sum_{t=k}^{\infty} w_{k,t}} R \sum_{q \in Q_1} \rho(q) \end{gathered} \quad . \tag{17}$$

Then, N_k must satisfy the following equation in function of $w_{k,t}$, in order to have a positive c_1 for listing 1.1:

$$\exists N_k, \ \forall k, \ 0 < c_1 \leq \frac{\sum_{t=k}^{k+N_k} w_{k,t}}{\sum_{t=k}^{\infty} w_{k,t}} R.$$

By (6) such N_k always exists, unless $\sum_{n=k}^{\infty} w_{k,t} = 0$, but that case does not need to be considered, as no rational choice of action will never need to be made, and the agent may well be considered "dead".

Now, by recurrence, as in the previous proof, the agent then always chooses action a_1 which is suboptimal compared to a_0: $D_k^{\mu\rho} = (1 - R)$.

Therefore AIρ does not asymptotically learn q_2.

□

Examples of N_k for some horizon functions. For a constant horizon or an exponential or over-harmonic discounting horizon that depends only on the index $i = (t - k)$ and not directly on k or t alone, even if the horizon is infinite, we can trivially take $N_k = 0$.

For a variable horizon m_k we can take for example $N_k = m_k$ so that the agent cannot receive any reward up to its horizon if it chooses a_0 at some step.

For a fixed lifetime T, we can simply take $N_k = T - k$.

7 Conclusion and Outlook

We have shown that *no* greedy universal agent with a prior that is independent of the history, and AIXI in particular, can always asymptotically converge to

AIMU, even when comparing the number of *mistakes* (suboptimal choices) made by the agent.

These proofs raise some questions. Is environment q_2 a plausible problem? The definition of q_2 is quite simple, and one can easily imagine an environment where some previously bad option suddenly turns, after a long time, into a better option that what the agent already has. Without further information from the environment, it is plausible that a human being or a animal will be very reluctant to try once in a while some option that generally returns the worst possible instantaneous reward 0, which is then a punishment. But that should not mean that it should never try again.

Another question is whether the initial history H given to the agent is plausible, i.e. whether the agent would generate such history itself. It is not easy to answer this question, but like listing 1.1 that discards the most probable environments, it is plausible to think that the agent will try by itself to quickly discard such environments in order to rapidly gain information about μ.

Is it possible to modify AIρ to make it a universal asymptotic learner? One possible way is to use exploration strategies like ϵ-greedy or Softmax [12], as proposed in [5, p.33] and as used in [13].

However, such setting always needs an additional parameter that must be tuned to the problem at hand, which is exactly what the AIXI framework is supposed to avoid. There is currently no known optimal stochastic exploration strategy for all computable problems, however there is hope that some variant of AIXI could expect to converge faster to AIMU than any other learner. We are currently working along these lines.

One of the most important questions is whether there exists a deterministic (non-stochastic) strategy that achieves the desired behavior, as Solomonoff Induction is for Sequence Prediction, and as AIXI was hoped to be for Reinforcement Learning.

Dynamic priors, that assign different probabilities to a given environment in function of the interaction history, might also be a solution to avoid local maxima while keeping a deterministic greedy agent, thus hopefully avoiding the introduction of a new parameter.

Acknowledgments. Many thanks to Marcus Hutter, Antoine Cornuéjols and the ALT reviewers for useful comments on previous versions of this paper.

References

1. Cohen, J.D., McClure, S.M., Yu, A.J.: Should I stay or should I go? How the human brain manages the trade-off between exploitation and exploration. Philosophical Transactions of the Royal Society B: Biological Sciences 362(1481), 933–942 (2007)
2. Hutter, M.: A theory of universal artificial intelligence based on algorithmic complexity. Arxiv (April 2000), http://arxiv.org/abs/cs/0004001
3. Hutter, M.: Self-optimizing and pareto-optimal policies in general environments based on bayes-mixtures. In: Kivinen, J., Sloan, R.H. (eds.) COLT 2002. LNCS (LNAI), vol. 2375, pp. 364–379. Springer, Heidelberg (2002)

4. Hutter, M.: Universal Artificial Intelligence: Sequential Decisions Based On Algorithmic Probability. Springer, Heidelberg (2005)
5. Hutter, M.: Universal algorithmic intelligence: A mathematical top-down approach. In: Artificial General Intelligence, pp. 227–290. Springer, Heidelberg (2007)
6. Kraft, L.G.: A device for quantizing, grouping, and coding amplitude modulated pulses. Ph.D. thesis, MIT, Electrical Engineering Department, Cambridge, MA (1949)
7. Li, M., Vitanyi, P.: An Introduction to Kolmogorov Complexity and Its Applications. Springer, New York (2008)
8. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 2nd edn. Prentice-Hall, Englewood Cliffs (2003)
9. Schmidhuber, J.: The speed prior: A new simplicity measure yielding near-optimal computable predictions. In: Kivinen, J., Sloan, R.H. (eds.) COLT 2002. LNCS (LNAI), vol. 2375, pp. 216–228. Springer, Heidelberg (2002)
10. Solomonoff, R.: Complexity-based induction systems: comparisons and convergence theorems. IEEE Transactions on Information Theory 24(4), 422–432 (1978)
11. Solomonoff, R.J.: A formal theory of inductive inference. Part I. Information and Control 7, 1–22 (1964)
12. Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998); a Bradford Book
13. Veness, J., Ng, K.S., Hutter, M., Silver, D.: A monte carlo AIXI approximation. Arxiv (September 2009), http://arxiv.org/abs/0909.0801

Consistency of Feature Markov Processes

Peter Sunehag and Marcus Hutter

RSISE @ Australian National University and SML@NICTA
Canberra, ACT, 0200, Australia
{Peter.Sunehag,Marcus.Hutter}@anu.edu.au

Abstract. We are studying long term sequence prediction (forecasting). We approach this by investigating criteria for choosing a compact useful state representation. The state is supposed to summarize useful information from the history. We want a method that is asymptotically consistent in the sense it will provably eventually only choose between alternatives that satisfy an optimality property related to the used criterion. We extend our work to the case where there is side information that one can take advantage of and, furthermore, we briefly discuss the active setting where an agent takes actions to achieve desirable outcomes.

1 Introduction

When studying long term sequence prediction one is interested in answering questions like: What will the next k observations be? How often will a certain event or a sequence of events occur? What is the average rate of a variable like cost or income? This can be interesting for forecasting time series and for choosing policies with desirable outcomes.

Hidden Markov Models [CMR05, EM02] are often used for long term forecasting and sequence prediction. In this article we will restrict our study to models based on states that result from a deterministic function of the history, in other words, states that summarize useful information that has been observed so far. We will consider finite state space maps with the property that given the current state and the next observation we can determine the next state. These maps are sometimes called Probabilistic-Deterministic Finite Automata (PDFA) [VTdlH$^+$5a] and they have recently been applied in reinforcement learning [Mah10]. A particular example of this is to use suffix trees [Ris83, Sin96, McC96].

Our goal is to prove consistency for our penalized Maximum Likelihood criteria for picking a map from histories to states in the sense that we want to eventually only choose between alternatives that are optimal for prediction. The sense of optimality could relate to predicting the next symbol, the next k symbols or to have minimal entropy rate for an infinite horizon.

After the preliminary Section 2 we begin our theory development in Section 3. In our problem setting we have a finite set $\mathcal{Y}$, a sequence y_n of elements from $\mathcal{Y}$, and we are interested in predicting the future of the sequence y_n. To do this, being inspired by [Hut09] where general criteria for choosing a feature map

© Springer-Verlag Berlin Heidelberg 2010

for reinforcement learning were discussed, we first want to learn a *feature map* $\Phi(y_{1:n}) = s_n$ where $y_{1:t} := y_1, ..., y_t$.

We would like the map to have the following properties:

1. The distribution for the sequence s_n induced by the distribution for the sequence y_n should be that of a Markov chain or should be a distribution which is indistinguishable from a Markov chain for the purpose of predicting the sequence y_n.
2. We want as few states as possible so that we can learn a model from a modest amount of data.
3. We want the model of the sequence y_n that arises as a function of the Markov chain s_n to be as good as possible. Ideally it should be the true distribution.

Our approach consists of defining criteria that can be applied to any class of Φ, but later we restrict our study to a class of maps that are defined by finite-state machines. These maps are defined by introducing a deterministic function ψ such that $s_n = \psi(s_{n-1}, y_n)$. If we have chosen such a map ψ and a first state s_0 then the sequence y_n determines a unique sequence s_n and therefore we have also defined a map $\Phi(y_{1:n}) = s_n$.

In Section 2 we provide some preliminaries on random sequences and Hidden Markov Models. We introduce a class of ergodic sequences which is the class of sequences that we work with in this article. They are sequences with the property that an individual sequence determines a distribution over infinite sequences. We present our consistency theory by first presenting very generic results in the beginning of Section 3 and then we show how various classes of maps and models fit into this. This has the consequence that we first have results where we guarantee optimality given that the individual sequence that we work with has certain properties (and these results, therefore, have no "almost sure" in the statement since the setting is not probabilistic) while in the latter part we show that if we sample the sequence in certain ways we will almost surely get a sequence with these properties. In particular in Section 4 we will take a closer look at suffix tree sources and maps based on finite state machines related to probabilistic deterministic finite automata. Section 5 summarizes the findings in a main theorem that says under some assumptions (a class of maps based on finite state machines of bounded memory and ergodicity) we will recover the true model (or the closest we can get to the true model). Section 6 contains a discussion of sequence prediction with side information, Section 7 briefly discusses the active case where an agent acts in an environment and earns rewards, and finally Section 8 contains our conclusions.

2 Preliminaries

In this section we will review some notions and results that the rest of the article will rely upon. We start with random sequences and then follows a section on Hidden Markov Models (HMM).

Random Sequences. Consider the set of all infinite sequences $y_t, t = 1, 2, ...$ of elements from a finite alphabet $\mathcal{Y}$. We equip the set with the σ-algebra that is generated by the cylinder sets $\Gamma_{y_{1:n}} = \{x_{1:\infty} | \, x_t = y_t, t = 1, ..., n\}$. A measure with respect to this space is determined by its values on the cylinder sets. Not every set of values is valid. We need to assume that the measure of $\Gamma_{y_{1:t}}$ is the sum of the measures of the sets $\Gamma_{y_{1:t}\tilde{y}}$ for all possible $\tilde{y} \in \mathcal{Y}$. If we want it to be a probability measure we furthermore need to assume that the measure of the whole space $\mathcal{Y}^\infty$ (which is the cylinder set Γ_ϵ of the empty string ϵ) equals to one. The concept that is introduced in the following two definitions is of central importance to this article. In particular *ergodic sequences* is the class of sequences that we intend to model. They are sequences that can be used to define a distribution over infinite sequences that we will be interested in learning.

Definition 1 (Distribution defined from one sequence). *A sequence $y_{1:\infty}$ defines a probability distribution on infinite sequences if the (relative) frequency of every finite substring of $y_{1:\infty}$ converges asymptotically. The probabilities of the cylinder sets are defined to equal those limits:*

$$\Gamma_{z_{1:m}} := \lim_{n\to\infty} \#\{t \le n : y_{t+1:t+m} = z_{1:m}\}/n$$

Definition 2 (ergodic sequence). *We say that a sequence is ergodic if the frequencies of every finite substring are converging asymptotically.*

As probabilistic models for random sequences we will in this article focus on Hidden Markov Models (HMMs) [BP66, Pet69]. More recent surveys on Hidden Markov Models are [EM02, CMR05].

Hidden Markov Models. Here we define distributions over sequences of elements from a finite set $\mathcal{Y}$ of size Y based on an unobserved Markov chain of elements from a finite state set $\mathcal{S}$ of size S.

Definition 3 (Hidden Markov Model, HMM). *Assume that we have a Markov chain with an $S \times S$ transition matrix $T = (T_{s,s'})$ and that we also have an $S \times Y$ emission matrix $E = (E_{s,y})$ where $E_{s,y}$ is the probability that state s will generate outcome $y \in \mathcal{Y}$. If we introduce a starting probability vector we have defined a probability distribution over sequences of elements from $\mathcal{Y}$. This is called a Hidden Markov Model (HMM).*

Sequence Prediction. One use of Hidden Markov Models (and functions of Markov chains) is sequence prediction. Given a history $y_1, ..., y_n$ we want to predict the future $y_{n+1}, ...$ In some situations we know what state we are in at time n and that state then summarizes the entire history without losing any useful information since the future is conditionally independent of the past, given the current state. If we are doing one step prediction we are interested in knowing $Pr(y_{n+1}|s_n)$. We can also consider a zero step lookahead (called filtering) $Pr(y_n|s_n)$ or an m step $Pr(y_{n+1}, ..., y_{n+m}|s_n)$. The m step could also be just $Pr(y_{n+m}|s_n)$. In a sense we can consider an infinite lookahead ability evaluated by the entropy rate $-\lim_{m\to\infty} \frac{1}{m} \log Pr(y_{n+1}, ..., y_{n+m}|s_n)$. If the Markov chain is ergodic this limit does not depend on the state s_n.

Limit Theorems. The following theory that is the foundation for studying consistency of HMMs was developed in [BP66] and [Pet69]. See [CMR05] chapter 12 for the modern state of the art.

Definition 4 (ergodic Markov chain). *A Markov chain (and the stochastic matrix that contains its transition probabilities) is called ergodic if it is possible to move from state s to state s' in a finite number of steps for all s and s'.*

The following theorem [CMR05] introduces the generalized cross-entropy H and shows that it is well defined and that it can be estimated for ergodic HMMs. It can be interpreted as the (idealized) expected number of bits needed for coding a symbol generated by a distribution defined by θ_0 but using the distribution defined by θ.

Theorem 5 (ergodic HMMs). *If θ and θ_0 are HMM parameters where the transition matrix for θ_0 is an ergodic stochastic matrix, then there exists a finite number $H(\theta_0, \theta)$ (which can also be defined as $\lim_{n\to\infty} H_{n,s}(\theta_0, \theta)$ for any initial state s where $H_{n,s}(\theta_0, \theta) := \frac{1}{n}\mathbb{E}_{\theta_0} \log Pr(y_1, ..., y_n | s_0 = s, \theta))$ such that P_{θ_0} a.s.*

$$-\lim_{n\to\infty} \frac{1}{n} \log Pr(y_1, ..., y_n |\ \theta) \ = \ H(\theta_0, \theta)$$

and the convergence is uniform in the parameter space.

Definition 6 (Equivalent HMMs). *For an HMM θ_0, let $M[\theta_0]$ be the set of all θ such that the HMM with parameters θ define the same distribution over outcomes as the HMM with parameters θ_0.*

Theorem 7 (Minimal cross-entropy for the truth and only the truth). *$H(\theta_0, \theta) \geq H(\theta_0, \theta_0)$ with equality if and only if $\theta \in M[\theta_0]$.*

3 Maps from Histories to States

Given a sequence of elements y_n from a finite alphabet we want to define a map $\Phi : \mathcal{Y}^* \to \mathcal{S}$, which maps histories (finite strings) of elements to states $\Phi(y_{1:n}) = s_n$. The reasons for this include, as was explained in the introduction, in particular the ability to learn a model efficiently. Suppose that every Φ under consideration is such that the size of its state space $\mathcal{S}$ is a finite number that depends on Φ.

We are also interested in the case when we have side information $x_n \in \mathcal{X}$ and we define a map $\Phi : (\mathcal{X} \times \mathcal{Y})^* \to \mathcal{S}$. In this more general case the models that we consider for the sequence y will have hidden states while in the case without side information the state (given the y sequence) is not hidden. We have two reasons for expressing everything in an HMM framework. We can model long-range dependence in the y_n sequence through having states and we include the more general case where there is side information.

Definition 8 (Feature sequence/process). *A map Φ from finite strings of elements from $\mathcal{Y}$ (or $\mathcal{X} \times \mathcal{Y}$) to elements in a finite set $\mathcal{S}$ and a sequence $y_{1:n}$ induces a state sequence $s_{1:n}$. Define an HMM through maximum likelihood estimation: The sequence $s_t = \Phi(y_{1:t})$ gives transition matrix $T(n) = (T_{s,s'})$ of probabilities*

$$T_{s,s'}(n) \;:=\; \frac{\#\{t \le n | s_t = s,\; s_{t+1} = s'\}}{\#\{t \le n | s_t = s\}}$$

and emission matrix $E(n)$ of probabilities

$$E_{s,y}(n) \;:=\; \frac{\#\{t \le n | s_t = s,\; y_t = y\}}{\#\{t \le n | s_t = s\}}.$$

Denote those HMMs by $\hat{\theta}_n := (T(n), E(n))$. We will refer to the sequence $\hat{\theta}_n$ as the parameters corresponding to Φ or generated by Φ.

We will first state results based on some generic properties that we have defined with just the goal of making the proofs work. Then we will show that some more easily understandable cases will satisfy these properties. We structure it this way not only for generality but also to make the proof techniques clearer.

Ergodic Sequences. We begin by defining the fundamental ergodicity properties that we will rely upon. We provide asymptotic results for individual sequences that satisfy these properties. In the next two subsections we identify situations where we will almost surely get such a sequence which satisfies these ergodicity properties.

Definition 9 (ergodic w.r.t. Φ). *As stated in Definition 2, we say that a sequence y_t is ergodic if all substring frequencies converge as $n \to \infty$. Furthermore we say that*
1. the sequence y_t is ergodic with respect to a map $\Phi(y_{1:t}) = s_t$ if all state transition frequencies $T_{s,s'}(n)$ and emission frequencies $E_{s,y}(n)$ converge as $n \to \infty$.
2. the sequence y_t is ergodic with respect to a class of maps if it is ergodic with respect to every map in the class.

Definition 10 (HMM-ergodic). *We say that a sequence y_t is HMM-ergodic for a set of HMMs Θ if there is an HMM with parameters θ_0 such that*

$$-\frac{1}{n} \log Pr(y_1, ..., y_n \mid \theta) \;\to\; H(\theta_0, \theta)$$

uniformly on compact subsets of Θ.

Definition 11 (Log-likelihood). $L_n(\Phi) = -\log Pr(y_1, ..., y_n | \hat{\theta}_n)$

We will prove our consistency results by first proving consistency using Maximum Likelihood (ML) for a finite class of maps and then we prove that we can add a sublinearly growing model complexity penalty and still have consistency.

Proposition 12 (HMM consistency of ML for finite class). *Suppose that y_t is HMM-ergodic for the parameter set Θ with optimal parameters (in the sense of Definition 10) θ_0, y_t is ergodic for the finite class of maps $\{\Phi_i\}_{i=1}^K$ and suppose that $\theta_i \in \Theta$ are the limiting parameters generated by Φ_i. Then it follows that there is $N < \infty$ such that for all $n \geq N$ the map Φ_i selected by minimizing L_n generates parameters $\hat{\theta}_i^n$ whose limit is in $\arg\min_{\theta_i} H(\theta_0, \theta_i)$.*

Proof. It follows from Definition 10 and continuity (in θ) of the log-likelihood that

$$\lim_{n\to\infty} \frac{1}{n} L_n(\Phi_i) \;=\; H(\theta_0, \theta_i)$$

since the convergence in Definition 10 is uniform. Note that the parameters that define the log-likelihood $L_n(\Phi_i)$ can be different for every n so the uniformity of the convergence is needed to draw the conclusion above. By Definition 9 we know that if $\hat{\theta}_n^i$ are the parameters generated by Φ_i at time n, then $\lim_{n\to\infty} \hat{\theta}_n^i = \theta_i$ exists for all i. It follows that if $\theta_i \notin \arg\min_{\theta_j} H(\theta_0, \theta_j)$ then there must be an $N < \infty$ such that Φ_i is not selected at times $n \geq N$. Since there are only finitely many maps in the class there will be a finite such N that works for all relevant i. ■

Definition 13 (HMM Cost function). *If the HMM with parameters $\hat{\theta}_n$ that has been estimated from Φ at time n has S states, then let*

$$Cost_n(\Phi) \;=\; -\log Pr(y_1, ..., y_n | \hat{\theta}_n) + pen(n, S)$$

where $pen(n, S)$ is a positive function that is increasing in both n and S and is such that $pen(n, S)/n \to 0$ for $n \to \infty$ for all S.

We call the negative log-probability term the *data coding cost* and the other term is the *model complexity penalty.* They are both motivated by coding (coding the data and the model). For instance in MDL/MML/BIC, $pen(n, S) = \frac{d}{2} \log n + O(1)$, where d is the dimensionality of the model θ.

Proposition 14. *Suppose that Φ_0 has optimal limiting parameters θ_0 with as few states as possible. In other words if an HMM has fewer states than the HMM defined by θ_0, then it has a strictly larger entropy rate. We use a (finite, countable, or uncountable) class of maps that includes only Φ_0 and maps that have strictly fewer states. We assume that all the maps generate converging parameters. Then there is an N such that the function Cost is minimized by Φ_0 at all times $n \geq N$.*

Proof. Suppose that θ_0 has S_0 states. We will use a bound for how close one can get to the true HMM using fewer states. We would like to have a constant $\varepsilon > 0$ such that $H(\theta_0, \theta) > H(\theta_0, \theta_0) + \varepsilon$ for all θ with fewer then S_0 states. The existence of such an ε follows from continuity of H (which is actually also differentiable [BP66]), the fact that the HMMs with fewer than S_0 states can be compactly (in the parameter space) embedded into the space of HMMs with exactly S_0 states, and that this embedded subspace has a strictly positive minimum Euclidean distance from θ_0 in this parameter space.

The existence of $\varepsilon > 0$ with this property implies the existence of $D > 0$ such that the alternatives with fewer than S_0 states have, for large n, at least Dn worse log probabilities than the distribution θ_0. Therefore the penalty term (for which $pen(n, S)/n \to 0$) will not be able to indefinitely compensate for the inferior modeling. ■

Theorem 15 (HMM consistency of Cost for finite class). *Proposition 12 is also true for Cost.*

Proof. $H(\theta_0, \theta_k) < H(\theta_0, \theta_j)$ implies that there is a constant $C > 0$ such that for large n, $L_n(\Phi_j) - L_n(\Phi_k) \geq Cn$. Since $pen(n, S)/n \to 0$ for $n \to \infty$ we know that any difference in model penalty will be overtaken by the linearly growing difference in data code length. ■

Maps that induce HMMs. In this section we will assume that we use a class of maps whose states we know form a Markov chain.

Definition 16 (Feature Markov Process, ΦMP). *Suppose that*

$$Pr(y_n|\Phi_0(y_1), ..., \Phi_0(y_{1:n})) \; = \; Pr(y_n|\Phi_0(y_{1:n}))$$

and that the state sequence is Markov, i.e.

$$Pr(\Phi_0(y_{1:n})|\Phi_0(y_1), ..., \Phi_0(y_{1:n-1})) \; = \; Pr(\Phi_0(y_{1:n})|\Phi_0(y_{1:n-1})).$$

Then we say that Φ_0 induces an HMM. We call HMMs induced by Φ_0, Feature Markov Process (ΦMP). If the HMM that is defined this way by Φ_0 is the true distribution for the sequence $y_1, y_2, ...$, then we say that "Φ_0 is correct".

We will only discuss the situation when the true HMM is ergodic so we will only say that there is a correct Φ_0 in those situations, hence the statement Φ_0 is correct will contain the assumption that the truth is ergodic.

Example 17. The map Φ which sends everything to the same state always induces an HMM but, unless the sequence $y_1, y_2, ...$ is i.i.d, it is not correct. ◊

Proposition 18 (Convergence of estimated distributions). *If Φ_0 is correct then $P_{\hat{\theta}_n} \to P_{\theta_0}$ for $n \to \infty$ (as distributions on finite strings of a (any) fixed length), where P_{θ_0} is the true HMM distribution for the outcomes, P_θ is the HMM distribution defined by θ and $\hat{\theta}_n$ are the parameters generated by Φ_0.*

Proof. We are estimating the parameters $\hat{\theta}_n$ through maximum likelihood for the generated sequence of states. Consistency of maximum likelihood estimation for Markov chains implies that $\hat{\theta}_n \to \theta_0$. This implies the proposition due to continuity with respect to the parameters of the likelihood (for any finite sequence length). ■

Proposition 19 (Inducing HMM implies drawing ergodic sequences). *If we have a set of maps that induce HMMs and the sequence y_t is drawn from one of the induced ergodic HMMs, then almost surely*
1. y_t is HMM-ergodic
2. we will draw an ergodic sequence y_t with respect to the considered class of maps.

Proof. 1. is a consequence of Theorem 5.
2. This follows from consistency of maximum likelihood for Markov chains (generalized law of large numbers) since the claim is that state transition frequencies and emission frequencies converge. ■

4 Maps Based on Finite State Machines (FSMs)

We will in this section consider maps of a special form that are related to PDFAs. We will assume that Φ is such that there is a ψ such that

$$\Phi(y_{1:n}) \;=\; \psi(\Phi(y_{1:n-1}), y_n).$$

In other words, the current state is derived deterministically from the previous state and the current perception. Given an initial state the state sequence is then deterministically determined by the perceptions and therefore the combination of ψ with an initial state defines a map Φ from histories to states. This class of maps Φ can also define a class of probabilistic models of the sequence y_n by assuming that y_n only depends on $s_{n-1} = \Phi(y_{1:n-1})$. This leads to the formula

$$Pr(s'|s) \;=\; \sum_{y:\psi(s,y)=s'} Pr(y|s)$$

and as a result we have defined an HMM for the sequence y_n.

Definition 20 (Sampling from FSM). *If we follow the procedure above we say that we have sampled the sequence y_t from the FSM. If the Markov chain of states is ergodic we say that we have sampled y_t ergodically from the FSM.*

Suffix Trees. We consider a class of maps based on FSMs that can be expressed using Suffix Trees [Ris86] with the same states (suffixes) as the FSM. The resulting models are sometimes called FSMX sources. A suffix tree is defined by a suffix set which is a set of finite strings. The set must have the property that none of the strings is an ending substring (a suffix) of another string in the set and such that any sufficiently long string ends with a substring in the suffix set. Given any sufficiently long string we then know that it ends with exactly one of the suffixes from the suffix set. If the suffix set furthermore has the property that given the previous suffix and the new symbol there is exactly one element (state) from the suffix set that can (and is) the end of the new longer string, then it is an FSMX source. Another terminology says that the suffix set is FSM closed. The property implies (directly by definition) that there is a map ψ such that $\psi(s_{t-1}, y_t) = s_t$.

The following proposition shows a very nice connection between ergodic sequences and FSMX sources which will be generalized in Proposition 25 to more general sources based on bounded-memory FSMs.

Proposition 21 (ergodicity of suffix trees). *If we have a set of maps based on FSMs that can be expressed by suffix trees, and the sequence y_t is sampled ergodically (Definition 20) using one of the maps, then almost surely we get a sequence y_t that is ergodic with respect to the considered class of maps and y_t is HMM-ergodic.*

Lemma 22. *If the sequence y_t is ergodic, then the state transition frequencies and emission (of y) frequencies for a FSM closed suffix tree are converging.*

Proof. Let the map Φ be defined by the suffix set in question. Suppose that s' is a suffix that can follow directly after s. This means that there is a symbol y such that if you concatenate it to the end of the string s, then this new string $\tilde{s}$ ends with the string s'. This means that whenever a string of symbols $y_{1:n}$ ends with $\tilde{s}$, then the sequence of states generated by applying the map Φ to the sequence $y_{1:n}$ will end with $s_{n-1} = s$ and $s_n = s'$. It is also true that whenever the state sequence ends with ss' then $y_{1:n}$ ends with $\tilde{s}$. Therefore, the counts (of ss' in the state sequence and $\tilde{s}$ in the y sequence) up until any finite time point are also equal. We will in this proof say that $\tilde{s}$ is the string that corresponds to ss'. Given any ordered pair of states (s, s') where s' can follow s, let $c_{s,s'}(n)$ be the number of times ss' occurs in the state sequence up to time n and let $d_{s,s'}(n)$ be the number of times the string $\tilde{s}$ that corresponds to ss' has occurred. We know that $c_{s,s'}(n) = d_{s,s'}(n)$ for any such pair ss' and any n. If s' cannot follow s we let both $c_{s,s'} = 0$ and $d_{s,s'} = 0$. The state transition frequency for the transition from s to s' up until time n is

$$\frac{c_{s,s'}(n)}{\sum_{s'} c_{s,s'}(n)} = \frac{d_{s,s'}(n)}{\sum_{s'} d_{s,s'}(n)} = \frac{d_{s,s'}(n)}{d_s(n)} = \frac{d_{s,s'}(n)}{n} \frac{n}{d_s(n)}$$

where $d_s(n)$ is the number of times that the string that defines s has occurred up until time n in the y sequence. The right hand side converges to the frequency of the string $\tilde{s}$ divided by the frequency of the string that defines s. Thus we have proved that state transition frequencies converge. Emissions work the same way. ■

Lemma 23. *If we sample y_t ergodically from a suffix tree FSM, then the frequency for each finite substring will converge almost surely. In other words the sequence y_t is almost surely ergodic.*

Proof. If the suffix tree defines an FSM as we have defined it above, the states of the suffix tree will form an ergodic Markov chain. An ergodic Markov chain is stationary. For any state and finite string of perceptions there is a certain fixed probability of drawing the string in question. The frequency of the string *str* is $\sum_s Pr(s)Pr(str|s)$ where $Pr(s)$ is the stationary probability of seeing s and $Pr(str|s)$ is the probability of directly seeing exactly *str* conditioned on being in state s. It follows from the law of large numbers that the frequency of any finite string *str* converges.

Another way of understanding this result is that it is implied by the convergence of the frequency of any finite string of states in the state sequence. ■

Proof. **of Proposition 21.** Lemma 22 and Lemma 23 together imply the proposition since they say that if we sample from a suffix tree then we almost surely get converging frequencies for all finite substrings and this implies converging transition frequencies for the states from any suffix tree. ■

Bounded-Memory FSMs. We here notice that the reasons that the suffix tree theory above worked actually relate to a larger class, namely a class of FSMs where the internal state is determined by at most a finite number of previous time steps in the history.

Definition 24 (bounded memory FSM). *Suppose that there is a constant κ such that if we know the last $\kappa+1$ perceptions $y_{t-\kappa}, ..., y_t$ then the present state s_t is uniquely determined. Then we say that the FSM has memory of at most length κ (not counting the current) and that it has bounded memory.*

Proposition 25 (ergodicity of FSMs). *1. Consider a sequence y_t whose finite substring frequencies converge (i.e. the sequence is ergodic) and an FSM of bounded memory, then the sequence is ergodic with respect to the map defined by the FSM.*
2. If we sample a sequence y_t ergodically from an FSM with bounded memory then almost surely y_t is HMM-ergodic and its finite substring frequencies converge.

Proof. The proof works the same way as for suffix tree FSMs. If an FSM has finite memory of length κ then there is a suffix tree of that depth with every suffix of full length and every state of the FSM is a subset of the states of that suffix tree. The FSM is a partition of the suffix set into disjoint subsets. Every state transition for the FSM is exactly one of a set of state transitions for the suffix tree states and the frequency of every ordered pair of suffix tree states converge almost surely as before. Therefore, the state transition frequencies for the FSM will almost surely converge.

A distribution that is defined using an FSM of bounded memory can also be defined using a suffix tree, so 2. reduces to this case ■

5 The Main Result for Sequence Prediction

In this section we summarize our results in a main theorem. It follows directly from a combination of results in previous sections. They are stated with respect to our main class of maps, namely the class that is defined by bounded-memory FSMs. The generating models that we consider are models that are defined from a map in this class in such a way that the states form an ergodic Markov chain. We refer to this as sampling ergodically from the FSM. Our conclusion is that we will under these circumstances eventually only choose between maps which generate the best possible HMM parameters that can be achieved for the purpose of long-term sequence prediction. The model penalty term will influence the choice between these options towards simpler models.

The following theorem guarantees that we will almost surely asymptotically find a correct HMM for the sequence of interest under the assumption that it is possible.

Theorem 26. *If we consider a finite class of maps $\Phi_i, i = 0, 1, ..., k$ based on finite state machines of bounded memory and if we sample ergodically from a finite state machine of bounded memory, then there almost surely exist limiting parameters θ_i for all i and there is $N < \infty$ such that for all $n \geq N$ the map Φ_i selected at time $n \geq N$ by minimizing Cost, generates parameters whose limit is θ_0 which is assumed to be the optimal HMM parameters.*

Proof. We are going to make use of Proposition 25 together with Theorem 15. Proposition 25 shows that our assumptions imply the assumptions of Theorem 15 which provides our conclusion. ∎

Extension to countable classes. To extend our results from finite to countable classes of maps we need the model complexity penalty to be sufficiently rapidly growing in n and m. This is also necessary if we want to be sure that we eventually find a minimal representation of the optimal model that can be achieved by the class of maps.

Proposition 27 (Consistency for countable class). *Suppose that we have a countable class of maps $\Phi_i,\ i = 0, 1, ...$ and*

1. *Suppose that our class is such that for every finite k, there are at most finitely many maps with at most k states.*
2. *Suppose that θ_0 is an optimal HMM for the sequence y_t, that it has m states and that θ_0 is the limit of the parameters generated by Φ_0. Furthermore, suppose that there is finite N such that whenever $n > N$, $\tilde{m} > m$ and $\tilde{\theta}$ is any HMM with $\tilde{m}$ states we have $pen(n, m) - \log P_{\hat{\theta}_0^n}(y_1, ..., y_n) < pen(n, \tilde{m}) - \log P_{\tilde{\theta}}(y_1, ..., y_n)$. where $\hat{\theta}_0^n$ are the parameters generated by Φ_0.*

then Theorem 15 is true also for this countable class and we will furthermore eventually pick a map with at most m states.

Proof. The idea of the proof is to reduce the countable case to the finite case that we have already proven by using that when $n > N$ we will never pick a Φ with more than m states and then use the first property to say that the remaining class if finite. This reduction also shows that we will eventually not pick a map with more states than m. ∎

The first property in the proposition above holds for the class of suffix trees and for the class based on FSMs with bounded memory. The second property, but with the HMM maximum likelihood parameters $\theta(n)$ with m states (while we have ML for a sequence of states and observations) will almost surely hold if the penalty is such that we have strong consistency for the HMM criteria $\theta^* = \arg\max \log P_\theta(y_1, ..., y_n) - pen(n, m)$. This is studied in many articles, e.g.

[GB03] where strong consistency is proven for a penalty of the form $\beta(m)\log n$ where β is a cubic polynomial. Note that in the case without side information (if our map has the properties that $\Phi_0(y_{1:n})$ determine y_n and that $\Phi(y_{n-1})$ and y_n determine $\Phi(y_{1:n})$) the emissions are deterministic and the state sequence generated by any map is determined by the y sequence. This puts us in a simpler situation akin to the Markov order estimation problem [FLN96, CS00] where it is studied which penalties (e.g. BIC) will give us property 2. above.

Conjecture 28. We almost surely have Property 2. from Proposition 27 for the BIC penalty studied in [CS00].

6 Sequence Prediction with Side Information

In this section we will broaden our problem to the setting where we have side information available to help in our prediction task. In our problem setting we have two finite sets $\mathcal{X}$ and $\mathcal{Y}$, a sequence $p_n = (x_n, y_n)$ of elements from $\mathcal{X} \times \mathcal{Y}$, and we are interested in predicting the future of the sequence y_n. To do this we first want to learn a *feature map* $\Phi(p_{1:n}) = s_n$. In other words we want our current state to summarize all useful information from both the x and y sequence for the purpose of predicting the future of y only.

One obvious approach is to predict the future of the entire sequence p, i.e. predicting both x and y and then in the end only notice what we find out about y. This brings us back to the case we have studied already, since from this point of view there is no side information. A drawback with that approach can be that we create an unnecessarily complicated state representation since we are really only interested in predicting the y sequence.

In the case when there is no side information, $s_t = \Phi(y_{1:t})$. An important difference of the case with side information is that the sequence $s_{1:t}$ depends on both $y_{1:t}$ and $x_{1:t}$. Therefore for the latter case, if we would like to consider a distribution for y only, $y_1, ..., y_n$ does not determine the state sequence $s_1, ..., s_n$:

$$Pr(y_1, ..., y_n|\hat{\theta}_n) \;=\; \sum_{s_{1:n}, x_{1:n}} Pr(s_1, ..., s_n) Pr(x_1, ..., x_n, y_1, ..., y_n|s_1, ..., s_n, \hat{\theta}_n).$$

This is expression is of course also true in the absence of side information x, but then the sum collapses to one term since there is only one sequence of states $s_{1:n}$ that is compatible with $y_{1:n}$.

An alternative to using the *Cost* criteria on the p sequence is to only model the y sequence and let

$$L_n(\Phi) \;=\; -\log Pr(y_1, ..., y_n|\hat{\theta}_n)$$

and then define *Cost* in exactly the same way as before. This cost function was called *ICost* in [Hut09].

Theorem 29. *Theorem 26 is true for sequence prediction with side information using*

$$ICost_n(\Phi_i) \;=\; -\log Pr(y_1, ..., y_n|\hat{\theta}_n) + pen(n, S)$$

if we define "sample ergodically" to refer to the sequence $p_t = (x_t, y_t)$ instead of y_t.

Proof. The proofs work exactly as they are written for the case without side information. ■

Note that a map that is optimal for predicting the y sequence can have fewer states than a minimal map that can generate the model of the p sequence.

It is interesting to note that the interpretation of this result is not as clear as the case without side information. It guarantees that, given enough history, the chosen Φ can and will (with the asymptotic parameters) define the correct model for the y_t sequence but the x_t sequence has only played a part in the estimation and we are not guaranteed that we will make use of the extra information if it does not impact the entropy rate. In particular it is true if the information in x_t is only helpful for a finite number of time steps forward. In this case that gain will not affect the entropy rate which is a limit of averages. We have a more conclusive result for the case with side information when we use the first mentioned approach of applying *Cost* to the sequence p, since we proved consistency in the previous section in the sense of finding the true model when possible.

If we have injective maps Φ, e.g. maps defined by non-empty suffix trees, then we can rewrite *Cost* in a form that was used in [Hut09] also more generally. Therein a cost called *original cost* was defined as follows:

Definition 30 (OCost)

$$OCost \;=\; -\log Pr(s_1, ..., s_n) - \log Pr(y_1, ..., y_n | s_1, ..., s_n, \hat{\theta}_n) + pen(n, S).$$

Remark 31. If Φ_i is injective and we calculate *Cost* in the side information case then *Cost* = *OCost*. ◇

If we have no side information both *OCost* and *ICost* will be the same as *Cost* but they may differ when there is side information available. We remarked above that if we consider only injective Φ (e.g. non-empty suffix tree based maps) then *OCost* equals using *Cost* on the joint sequence $p_t = (x_t, y_t)$. As noted in [Hut09] *OCost* penalizes having many states more than *ICost* does and when considering non-injective Φ one risks getting a smaller than desired state space.

7 The Active Case

In this very brief section we will discuss how to map the active case to the previously introduced notions. The active case will be treated in depth in future articles. In the active case [RN10, SB98] we have an agent that interacts with an environment. The agent perceives observations o_t and real-valued rewards r_t and the agent takes actions a_t from a finite set of possible actions $\mathcal{A}$ with the goal of receiving high total reward in some sense. We will denote the events that have just occurred when the agent will take an action at time step t, i.e. a_t,

o_t, and r_t by e_t. We consider maps based on FSMs (PDFAs) that takes event sequences e_t as input. In the previous section's notation $x_t = (o_t, a_t)$ and $y_t = r_t$ and $p_t = e_t$. We chose this since we are interested in predicting which future rewards will result from actions chosen with the help of the observations. This would give us the possibility of determining which actions will earn the highest rewards.

At time $t-1$ the past $e_1, ..., e_{t-1}$ determines s_{t-1} and the agent takes an action a_{t-1} and o_t and r_t are generated according to distributions that only depend on s_{t-1} and a_{t-1}. Then we have generated e_t and $s_t = \psi(s_{t-1}, e_t)$.

Definition 32. *The above describes what we mean when we say that the FSM generates the environment. We say that the FSM generates the environment ergodically, if for any sequence of actions chosen such that the action frequencies for any state converge asymptotically, we will have state transitions and emission frequencies that converge almost surely to an ergodic HMM.*

Proposition 33. *Suppose that we have an FSM of bounded-memory generating the environment ergodically and the action frequencies for any state converge asymptotically, then we will almost surely generate an ergodic sequence of events and the reward sequence is HMM-ergodic.*

Proof. The situation reduces through Definition 32 to that of Proposition 25. ■

Theorem 34. *If we consider a finite class of maps $\Phi_i, i = 0, 1, ..., k$ based on finite state machines of bounded memory and if the environment is generated ergodically by a finite state machine of bounded memory and if the action frequencies for any internal state of the generating finite state machine converge, then there almost surely exist limiting state transition parameters θ_i for all i and there is $N < \infty$ such that for all $n \geq N$ the map Φ_i selected by minimizing ICost at time $n \geq N$ generates parameters whose limit is θ_0 which is the optimal HMM.*

Proof. We combine Proposition 33 with Theorem 29. ■

How to choose the actions to make the implications for reinforcement learning what we want them to be is the subject of ongoing work [Hut09].

8 Conclusions

Feature Markov Decision Processes were introduced [Hut09] as a framework for creating generic reinforcement learning agents that can learn to perform well in a large variety of complex environments. It was introduced as a concept without theory or empirical studies. First empirical results are reported in [Mah10]. Here we provide a consistency theory by focusing on the sequence prediction case with and without side information. We briefly discuss the active case where an agent takes actions that may affect the environment. The active case and empirical studies is the subject of ongoing and future work.

Acknowledgement. We thank the reviewers for their meticulous reading and valuable feedback and the Australian Research Council for support under grant DP0988049.

References

[BP66] Baum, L.E., Petrie, T.: Statistical inference for probabilistic functions of Finite State Markov chains. The Annals of Mathematical Statistics 37(6), 1554–1563 (1966)

[CMR05] Cappé, O., Moulines, E., Rydenp, T.: Inference in Hidden Markov Models. Springer Series in Statistics. Springer, New York (2005)

[CS00] Csiszr, I., Shields, P.C.: The consistency of the bic markov order estimator (2000)

[EM02] Ephraim, Y., Merhav, N.: Hidden Markov processes. IEEE Transactions on Information Theory 48(6), 1518–1569 (2002)

[FLN96] Finesso, L., Liu, C., Narayan, P.: The optimal error exponent for markov order estimation. IEEE Trans. Inform. Theory 42, 1488–1497 (1996)

[GB03] Gassiat, E., Boucheron, S.: Optimal error exponents in hidden Markov models order estimation. IEEE Transactions on Information Theory 49(4), 964–980 (2003)

[Hut09] Hutter, M.: Feature reinforcement learning: Part I: Unstructured MDPs. Journal of Artificial General Intelligence 1, 3–24 (2009)

[Mah10] Mahmud, M.M.: Constructing states for reinforcement learning. In: The 27:th International Conference on Machine Learning, ICML 2010 (2010)

[McC96] McCallum, A.K.: Reinforcement learning with selective perception and hidden state. PhD thesis, The University of Rochester (1996)

[Pet69] Petrie, T.: Probabilistic functions of Finite State Markov chains. The Annals of Mathematical Statistics 40(1), 97–115 (1969)

[Ris83] Rissanen, J.: A universal data compression system. IEEE Transactions on Information Theory 29(5), 656–663 (1983)

[Ris86] Rissanen, J.: Complexity of strings in the class of Markov sources. IEEE Transactions on Information Theory 32(4), 526–532 (1986)

[RN10] Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 3rd edn. Prentice-Hall, Englewood Cliffs (2010)

[SB98] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning). MIT Press, Cambridge (March 1998)

[Sin96] Singer, Y.: Adaptive mixtures of probabilistic transducers. Neural Computation 9, 1711–1733 (1996)

[VTdlH$^+$5a] Vidal, E., Thollard, F., de la Higuera, C., Casacuberta, F., Carrasco, R.C.: Probabilistic finite-state machines – Part I. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(7), 1013–1025 (2005a)

Algorithms for Adversarial Bandit Problems with Multiple Plays

Taishi Uchiya, Atsuyoshi Nakamura, and Mineichi Kudo

Graduate School of Information Science and Technology, Hokkaido University
Kita 14, Nishi 9, Kita-ku, Sapporo 060-0814, Hokkaido, Japan
{uchiya,atsu,mine}@main.ist.hokudai.ac.jp

Abstract. Adversarial bandit problems studied by Auer et al. [4] are multi-armed bandit problems in which no stochastic assumption is made on the nature of the process generating the rewards for actions. In this paper, we extend their theories to the case where $k(\geq 1)$ distinct actions are selected at each time step. As algorithms to solve our problem, we analyze an extension of **Exp3** [4] and an application of a bandit online linear optimization algorithm [1] in addition to two existing algorithms (**Exp3**,**ComBand** [5]) in terms of time and space efficiency and the regret for the best fixed action set. The extension of **Exp3**, called **Exp3.M**, performs best with respect to all the measures: it runs in $O(K(\log k + 1))$ time and $O(K)$ space, and suffers at most $O(\sqrt{kTK \log(K/k)})$ regret, where K is the number of possible actions and T is the number of iterations. The upper bound of the regret we proved for **Exp3.M** is an extension of that proved by Auer et al. for **Exp3**.

Keywords: Multi-armed bandit problem, adversarial bandit problem, online learning.

1 Introduction

Multi-armed bandit problems are a kind of sequential resource allocation problems in which one resource is allocated to one action among several alternative actions at each time step. Each allocation yields a reward and the objective of the problem is the maximization of the total reward. The problems are known as paradigms of the trade-off between exploration (for better future rewards) and exploitation (for high current rewards). These problems have been becoming more and more important in this Internet age because several problems such as server selection in network, Internet ad placement and market pricing at e-commerce sites [8] can all be formulated as multi-armed bandit problems.

Vast studies have been done so far on these problems [10], and the majority of them assumes that the bandit processes are stochastic. However, adversarial bandit problems studied by Auer et al. [4] make no stochastic assumption on the nature of the process generating the rewards for the actions, and they also have been becoming popular recently. There have been several extensions on this line such as the on-line shortest path problem [7], bandit online linear optimization [1] and combinatorial bandits [5].

© Springer-Verlag Berlin Heidelberg 2010

In this paper, we study adversarial bandit problems extended in one direction, namely, those problems with multiple plays. In this extension, k resources are allocated at each time step. The multiple play setting is practically useful for such problems as multiple ad placement on one web page, which is studied as the problem of multi-impressions in [11]. As for stochastic bandit problems, there are already several studies along this direction [2, 3, 13, 14], but, to the best of our knowledge, only the study made so far in the adversarial setting is combinatorial bandits of Cesa-Bianchi and Lugosi [5]. They considered a general bandit problem in which a player select one binary vector from a fixed set $\mathcal{S} \subseteq \{0,1\}^K$ at each time step. The k-sized subset version of their algorithm **ComBand** is just an algorithm for the multiple play setting. The regret for the best fixed k-sized action subset is $O(k^{\frac{3}{2}}\sqrt{TK\ln K})$, where T is the number of iterations and K is the number of possible actions. The time and space complexities of this algorithm are $O(kK^3)$ and $O(K^3)$, respectively. Note that here we only consider the case where selected k actions must be distinct, different from the studies in [12, 15].

Time and space complexities of **ComBand** can be improved a little by algorithm **BOLOM**, which is made by applying the bandit online linear optimization algorithm [1] to our problem. **BOLOM** runs in $O(K^3)$ time per iteration and $O(K^2)$ space regardless of the value of k. The regret of **BOLOM** for the best fixed action set is bounded by $O(kK^{\frac{3}{2}}\sqrt{T\log T})$, which is worse than **ComBand**. The best algorithm for the multiple play setting is algorithm **Exp3.M**, which is an extension of **Exp3** [4]. The action space is the same as that of **Exp3**, namely, **Exp3.M** keeps just K weights. Using the efficient k-combination selection procedure developed by Gandhi et al. [6], which can select a set S of k distinct actions so as to satisfy that each action i is selected with given probability p_i, **Exp3.M** runs in $O(K(\log k+1))$ time per iteration and $O(K)$ space, and achieves an upper bound $O(\sqrt{kTK\log(K/k)})$ on the regret for the best fixed action set. Note that this upper bound is an extension of that proved by Auer et al. [4] because they coincide when $k=1$. We also show that a lower bound of the regret on the problem is $\Omega(((K-k)/K)^2\sqrt{KT})$, which is also an extension of that proved in [4] on the original problem.

2 Problem Setting

An *adversarial bandit problem* [4] is specified by a set of possible player's actions $[K](\doteq \{1,2,\ldots,K\})$ and an assignment of rewards $\boldsymbol{x}(t)=(x_1(t),x_2(t),\ldots,x_K(t))$ at time step $t=1,2,\ldots,T$, where $x_i(t)\in[0,1]$ denotes the reward obtained by the player. In *multiple play* setting, the player selects a set of k distinct actions $S(t)\in\mathbf{C}([K],k)$ at each time step t, and after that, the player gets rewards $x_i(t)$ for $i\in S(t)$, where $\mathbf{C}([K],k)=\{S\subseteq[K] : |S|=k\}$, namely, the set of all subsets of size k in $[K]$. Throughout the paper, we use $|S|$ as the number of elements in S for any set S. Note that we also use notation $\mathbf{C}(K,k)$ which denotes the number $|\mathbf{C}([K],k)|\left(=\binom{K}{k}\right)$. All the information the player can obtain at time step t is only the rewards for the actions the player has selected at that time step.

The *cumulative reward* G_A of a player algorithm A is defined as

$$G_A \doteq \sum_{t=1}^{T} \sum_{i \in S(t)} x_i(t)$$

if the algorithm A chooses an action sequence $S(1), S(2), \ldots, S(T)$. The problem is to design a player algorithm A that maximizes its cumulative reward under the condition that an adversary who knows the strategy of A decides an assignment of rewards.

We measure the performance of algorithm A by the regret of A for the best fixed set of actions (that is called weak regret in [4]), which is defined by $G_{\textbf{max-k}} - G_A$, where

$$G_{\textbf{max-k}} \doteq \max_{S \in \mathbf{C}([K],k)} \sum_{t=1}^{T} \sum_{i \in S} x_i(t)$$

is the cumulative reward for the best fixed set of k distinct actions.

3 Previous Works

First, an adversarial multi-armed bandit problem with multiple plays can be solved by using **Exp3** developed by Auer et. al [4]. Regarding a set of k actions as one action makes **Exp3** applicable to our multiple play setting[1]. However, this arises a problem that the size of action space becomes large, namely, $\mathbf{C}(K,k) = \Omega(K^k)$. As a result, the regret upper bound obtained by Corollary 3.2 in [4] becomes

$$G_{\textbf{max-k}} - E[G_{\textbf{Exp3}}] \le 2k\sqrt{e-1}\sqrt{T\mathbf{C}(K,k)\ln \mathbf{C}(K,k)} \le 2.63\sqrt{k^3 T K^k \ln K},$$

and the time and space complexities becomes $\Omega(K^k)$.

The regret upper bound can be improved significantly even using $\mathbf{C}(K,k)$ weights like **Exp3**. Algorithm **ComBand** shown in Fig. 1 is the k-sized subset version of the algorithm developed by Cesa-Bianchi and Lugosi [5], which is just the algorithm that solves our problem as it is. Like **Exp3**, **ComBand** has one weight for each k-sized subset. At each time step, it randomly selects one k-sized subset according to a distribution calculated by the weights, and updates the weights depending on the obtained reward. The randomized selection method of **ComBand** is the same as that of **Exp3**, but **ComBand** uses more sophisticated method of weight update than **Exp3**. **ComBand** calculates a $K \times K$ matrix P_t that is defined as $E_{p_U}[\mathbf{1}_U \mathbf{1}_U^\top]$, where $\mathbf{1}_U$ is a K-dimensional vector whose ith component is 1 if $i \in U$ and 0 otherwise, and $\top$ denotes transpose. Then, the

[1] A reward for a set of k actions must be divided by k in the application.

```
**ComBand**(The k-sized subset version of the algorithm for Combinatorial Bandits)
Parameters:  $\gamma \in (0,1]$
Initialization:  $w_U(1) = 1$ for $U \in \mathbf{C}([K],k)$
**For each** $t = 1, 2, \ldots, T$,
    1. For $U \in \mathbf{C}([K],k)$ set
         $p_U(t) = (1-\gamma)\frac{w_U(t)}{\sum_{U' \in \mathbf{C}([K],k)} w_{U'}(t)} + \frac{\gamma}{\mathbf{C}(K,k)}.$
    2. Choose $S(t)$ randomly according to the distribution $p_U(t)$ for $U \in \mathbf{C}([K],k)$.
    3. Receive a reward $\sum_{i \in S(t)} x_i(t) \in [0,k]$.
    4. Set
         $P_t = \sum_{U \in \mathbf{C}([K],k)} p_U(t) \mathbf{1}_U \mathbf{1}_U^\top$ and
         $\hat{\mathbf{l}}(t) = \left(k - \sum_{i \in S(t)} x_i(t)\right) P_t^+ \mathbf{1}_{S(t)}.$
    5. For $U \in \mathbf{C}([K],k)$ set
         $w_U(t+1) = w_U(t) \exp\left(-\frac{\gamma(K-k)\sum_{i \in U} \hat{l}_i(t)}{kK(K-1)}\right).$
```

Fig. 1. Pseudocode of algorithm **ComBand** [5] (k-sized subset version)

K-dimensional vector of pseudo-losses $\hat{\mathbf{l}}(t)$ is calculated as $P_t^+ \mathbf{1}_{S(t)}$ multiplied by the loss $k - \sum_{i \in S(t)} x_i(t)$, where P_t^+ is the pseudo-inverse of P_t. For each k-sized subset U, the weight $w_U(t+1)$ is $w_U(t)$ multiplied by $exp\left(-\eta \sum_{i \in U} \hat{l}_i(t)\right)$, where $\hat{l}_i(t)$ is the ith component of $\hat{\mathbf{l}}(t)$ and $\eta = \gamma(K-k)/kK(K-1)$. By Theorem 1 and Proposition 15 in [5], for $k \le K/2$, we obtain

$$G_{\textbf{max-k}} - E[G_{\textbf{ComBand}}] \le \left(2 + \frac{K-1}{K-k}\right) k\sqrt{TK \ln \mathbf{C}(K,k)} \le 4\sqrt{k^3 TK \ln K}.$$

As commented in [5], there is an implementation of **ComBand** in which the time and space complexity is also significantly improved. In the efficient implementation, one weight w_i for each original action i is enough because weight w_U for a k-sized subset can be represented by $\prod_{i \in U} w_i$ and Step 5 in **ComBand** can be replaced with

$$\text{5'. For } i \in [K] \text{ set } w_i(t+1) = w_i(t) \exp\left(-\frac{\gamma(K-k)\hat{l}_i(t)}{kK(K-1)}\right).$$

Since each k-sized subset can be represented by a path from the source to the sink in G of Fig. 2, by dynamic programming technique of Takimoto and Warmuth

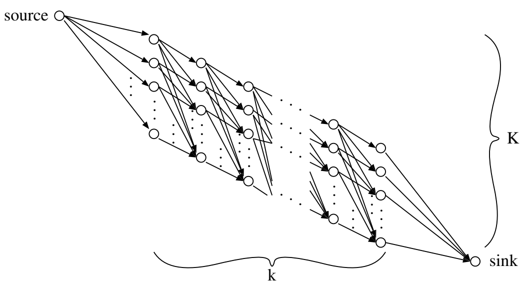


Fig. 2. The directed graph G such that a path in G has one-to-one correspondence to a k-sized subset of $[K]$

[16], $S(t)$ can be chosen without calculating $p_U(t)$ for all $U \in \mathbf{C}([K], k)$ in Step 2, and P_t can be calculated without taking a sum over all $U \in \mathbf{C}([K], k)$ in Step 4. (See also the proof of Theorem 3 in [7].) From the fact that the number of nodes in G is $O(kK)$ and the weights of the edges incoming into the same vertex are the same, $S(t)$ can be chosen in $O(kK)$ time and $O(kK)$ space, and P_t can be calculated in $O(kK^3)$ time and $O(K^3)$ space. Note that we do not have to keep edge information in memory because of its regularity. The pseudo-inverse P_t^+ can be calculated in $O(K^3)$ time and $O(K^2)$ space. In total, the time and space complexities of **ComBand** is $O(kK^3)$ and $O(K^3)$, respectively.

4 Application of Bandit Online Linear Optimization

A more efficient algorithm for large k can be obtained by applying an algorithm developed in the context of bandit online linear optimization [1]. In the bandit online linear optimization problem, the space corresponding to the action space is a compact closed convex set $\mathcal{K}$ in $\mathbb{R}^n$. At each time step t, a player chooses $\boldsymbol{q}_t \in \mathcal{K}$, then an adversary returns $\boldsymbol{x}_t^\top \boldsymbol{q}_t$ to the player. The player's goal is to minimize his regret defined as

$$\max_{\boldsymbol{q}^* \in \mathcal{K}} \sum_{i=1}^{T} \boldsymbol{x}_t^\top \boldsymbol{q}^* - \sum_{i=1}^{T} \boldsymbol{x}_t^\top \boldsymbol{q}_t.$$

To consider our multiple play setting bandit problem in this framework, set $\mathcal{K}$ to the convex hull of $\left\{\mathbf{1}_U \in \mathbb{R}^K : U \in \mathbf{C}([K], k)\right\}$, where $\mathbf{1}_U$ is a vector whose ith component is 1 if $i \in U$ and 0 otherwise. The following proposition holds.

Proposition 1. *The convex hull of* $\left\{\mathbf{1}_U \in \mathbb{R}^K : U \in \mathbf{C}([K], k)\right\}$ *is equal to* $\left\{\boldsymbol{q} : 0 \le q_i \le 1 \ \text{ for } \ i = 1, 2, \ldots, K, \sum_{i=1}^K q_i = k\right\}$.

Proof. Let $A = \left\{\boldsymbol{q} : 0 \le q_i \le 1 \ \text{ for } \ i = 1, 2, \ldots, K, \sum_{i=1}^K q_i = k\right\}$ and $B = \left\{\mathbf{1}_U \in \mathbb{R}^K : U \in \mathbf{C}([K], k)\right\}$. Then, B is a compact convex subset of $\mathbb{R}^K$

and the set of its extreme points is A. Thus, the convex hull of A is B by Krein-Milman theorem [9]. $\square$

By the above proposition, any $\boldsymbol{q} \in \mathcal{K}$ satisfies $\sum_{i=1}^{K} q_i = k$, so $\mathcal{K}$ can be regarded as a $(K-1)$-dimensional subspace $\mathcal{K}_{K-1}$ defined by

$$\mathcal{K}_{K-1} \doteq \left\{ \boldsymbol{q} : 0 \leq q_i \leq 1 \;\; \text{for} \;\; i = 1, 2, \ldots, K-1, 0 \leq k - \sum_{i=1}^{K-1} q_i \leq 1 \right\}.$$

Therefore, a linear optimization problem under the constraint of range $\mathcal{K}_{K-1}$ can be solved as the unconstrained linear optimization problem with the θ-self concordant barrier

$$R(\boldsymbol{q}) \doteq -\ln \left[\left\{ \prod_{i=1}^{K-1} q_i(1-q_i) \right\} \left(k - \sum_{i=1}^{K-1} q_i \right) \left(1 - k + \sum_{i=1}^{K-1} q_i \right) \right]$$

on $\mathcal{K}_{K-1}$, where $\theta = 2K$ for this barrier R. By applying Algorithm 1 in [1] to our multiple play setting bandit problem, we can obtain algorithm **BOLOM** shown in Fig. 3. In the application, there are two things we have to take care of. One is the difference of the problem settings: in linear optimization setting, the player can select any element $\boldsymbol{p}$ in $\mathcal{K}$, but in the multiple play setting, the player must select a set U from $\mathbf{C}([K], k)$, which corresponds to the original set of $\mathbf{C}(K, k)$ vectors before taking its convex hull.

This can be overcome by selecting a set $U \in \mathbf{C}([K], k)$ at random so as to satisfy the condition that each action i is selected with probability p_i. This selection guarantees that $E(\mathbf{1}_U) = \boldsymbol{p}$ holds, and the bound of Theorem 1 in [1] is still valid for the algorithm of the above modification by their Proposition 1 in Sec. 7 in [1]. Function **DepRound** [6] used in our algorithm is an efficient algorithm that makes such a selection, whose details are described in Sec. 7.

The other problem is that the reward $\boldsymbol{x}(t)^\top \boldsymbol{p}(t)$ expectedly received by the player at each time step t cannot be expressed linearly in $\mathcal{K}_{K-1}$: it is expressed as

$$(x_1(t) - x_K(t), \ldots, x_{K-1}(t) - x_K(t)) \begin{pmatrix} p_1(t) \\ \vdots \\ p_{K-1}(t) \end{pmatrix} + kx_K(t),$$

that is, it is expressed as a combination of a linear part with a new reward vector $(x_1(t) - x_K(t), \cdots, x_{K-1}(t) - x_K(t))^\top$ and a bias $kx_K(t)$.
Let $\boldsymbol{y}(t) = \frac{1}{k}(x_1(t) - x_K(t), \cdots, x_{K-1}(t) - x_K(t))^\top$. Then,

$$\begin{aligned} E_{S(t)} \frac{1}{k} \boldsymbol{x}(t)^\top \mathbf{1}_{S(t)} &= \frac{1}{k} \boldsymbol{x}(t)^\top E_{S(t)} \mathbf{1}_{S(t)} = \frac{1}{k} \boldsymbol{x}(t)^\top \boldsymbol{p}(t) \\ &= \boldsymbol{y}(t)^\top \begin{pmatrix} p_1(t) \\ \vdots \\ p_{K-1}(t) \end{pmatrix} + x_K(t) = \boldsymbol{y}(t)^\top (\boldsymbol{q}(t) + \varepsilon_t \lambda^{-\frac{1}{2}} \boldsymbol{e}_{i_t}) + x_K(t) \end{aligned}$$

```
BOLOM(Bandit Online Linear Optimization with Multiple plays)
Parameters:   $\eta > 0$
Initialization:
    $R(\boldsymbol{q}) = -\ln\left[\left\{\prod_{i=1}^{K-1} q_i(1-q_i)\right\}\left(k-\sum_{i=1}^{k-1} q_i\right)\left(1-k+\sum_{i=1}^{K-1} q_i\right)\right]$ for $\boldsymbol{q} \in \mathcal{K}_{K-1}$,
    $q_i(1) = \frac{k}{K}$ for $i = 1, 2, \ldots, K-1$
For each $t = 1, 2, \ldots, T$
    1. Calculate the set of eigenvectors $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{K-1}\}$ and
       eigenvalues $\{\lambda_1, \ldots, \lambda_{K-1}\}$ of $\nabla^2 R(\boldsymbol{q}(t))$.
    2. Choose $i_t$ uniformly at random from $\{1, \ldots, K-1\}$ and
       $\varepsilon_t = \pm 1$ with probability 1/2.
    3. Set
           $p_j(t) = q_j(t) + \varepsilon_t \lambda^{-\frac{1}{2}} e_{i_t,j}$ for $j = 1, 2, \ldots, K-1$,
           $p_K(t) = k - \sum_{i=1}^{K-1} p_i(t)$.
    4. Set
           $S(t) = \mathbf{DepRound}(k, (p_1(t), p_2(t), \ldots, p_K(t)))$.
    5. Receive rewards $x_i(t) \in [0,1]$ for $i \in S(t)$.
    6. Set
           $\hat{\boldsymbol{y}}(t) = (K-1)\left(\frac{1}{k}\sum_{i \in S(t)} x_i(t)\right)\varepsilon_t \lambda^{\frac{1}{2}} \boldsymbol{e}_{i_t}$,
           $\boldsymbol{q}(t+1) = \arg\min_{\boldsymbol{q} \in \mathcal{K}_{K-1}}\left[-\eta\sum_{s=1}^{t} \hat{\boldsymbol{y}}(s)^\top \boldsymbol{q} + R(\boldsymbol{q})\right]$.
```

Fig. 3. Pseudocode of algorithm **BOLOM**

holds. Thus, by virtue of random choice of ε_t,

$$
\begin{aligned}
E_{\varepsilon_t} E_{S(t)} \hat{\boldsymbol{y}}(t) = \frac{1}{2}(K-1)(\boldsymbol{y}(t)^\top(\boldsymbol{q}(t) + \lambda^{-\frac{1}{2}} \boldsymbol{e}_{i_t}) + x_K(t))\lambda_{i_t}^{\frac{1}{2}} \boldsymbol{e}_{i_t} \\
-\frac{1}{2}(K-1)(\boldsymbol{y}(t)^\top(\boldsymbol{q}(t) - \lambda^{-\frac{1}{2}} \boldsymbol{e}_{i_t}) + x_K(t))\lambda_{i_t}^{\frac{1}{2}} \boldsymbol{e}_{i_t} \\
= (K-1)(\boldsymbol{y}(t)^\top \boldsymbol{e}_{i_t})\boldsymbol{e}_{i_t}
\end{aligned}
$$

holds, so $E\hat{\boldsymbol{y}}(t) = E_{i_t} E_{\varepsilon_t} E_{S(t)} \hat{\boldsymbol{y}}(t) = \boldsymbol{y}(t)$ is still implied.

Therefore, by Theorem 1 in [1], we obtain the following theorem.

Theorem 1. *Set* $\eta = \sqrt{2K \log T}/(4(K-1)\sqrt{T})$. *Then*

$$
G_{\textbf{max-k}} - E[G_{\textbf{BOLOM}}] \leq 16k(K-1)\sqrt{2KT \log T} + \sqrt{(K-1)T}
$$

holds for any $T > 16K \log T$ and for any assignment of rewards.

The regret upper bound of **BOLOM** is worse than that of **ComBand** in both T and K. Algorithm **BOLOM** runs in $O(K^3)$ time per iteration and needs $O(K^2)$ space. Thus, **BOLOM** is more efficient than **ComBand**.

5 Multiple Play Version of Exp3

In Sec. 3, we saw that a direct application of **Exp3** for action space $\mathbf{C}([K], k)$ have to deal with $\Omega(K^k)$ weights, which caused a large regret upper bound and large time and space complexities. Can we apply algorithm **Exp3** for the original action space $[K]$ to solve the multiple play setting of a bandit problem? The answer is yes, and we develop such an algorithm in this section.

As **Exp3** does, our algorithm selects action i with probability $p_i(t)$ and estimates $x_i(t)$ by $\hat{x}_i(t) = x_i(t)/p_i(t)$ when action i is selected and by $\hat{x}_i(t) = 0$ otherwise. This calculation guarantees our algorithm to satisfy $E[\hat{x}_i(t)] = x_i(t)$ if action i is selected randomly with probability $p_i(t)$. Then, the problem is reduced to how to select k distinct actions under the condition that each action i is selected randomly with probability $p_i(t)$. This can be done in $O(K)$ time per iteration by using function **DepRound** [6]. (See Sec. 7.) Note that $\sum_{i=1}^{K} p_i(t)$ must be k in this problem setting because the expected total number of selection is $\sum_{i=1}^{K} p_i(t)$.

However, a new problem arises using this selection: probability $p_i(t)$ possibly becomes more than 1 if it is set to a value proportional to weight $w_i(t)$ that is more than $\frac{1}{k}\sum_{j=1}^{K} w_j(t)$. Our countermeasure for this situation is to let $p_i(t)$ linearly depend on modified weight $w'_i(t)$ that is made from $w_i(t)$ by cutting off at some threshold α_t.

Our extension of algorithm **Exp3** for multiple play setting is algorithm **Exp3.M** shown in Fig. 4. If all $w_j(t)$ are less than $\left(\frac{1}{k} - \frac{\gamma}{K}\right)\sum_{i=1}^{K} w_i(t)/(1-\gamma)$, which is checked at Step 1, $p_j(t)$ calculated at Step 3 is less than 1 for all $i = 1, 2, \ldots, K$ without weight modification. In this case, $S_0(t)$ is set to $\emptyset$ at Step 1. Otherwise, threshold α_t is set to an appropriate value, and all the actions i with $w_i(t) \geq \alpha_t$ are classified into $S_0(t)$. The temporal weight $w'_i(t)$ is set to α_t for $i \in S_0(t)$ and $w_i(t)$ for $i \notin S_0(t)$. Since

$$\frac{w'_i(t)}{\sum_{j=1}^{K} w'_j(t)} = \frac{\alpha_t}{\sum_{w_j(t)\geq\alpha_t} \alpha_t + \sum_{w_j(t)<\alpha_t} w_j(t)}$$

holds for all $i \in S_0(t)$, $p_i(t)$ is set to 1 for all $i \in S_0(t)$ in Step 3 if α_t is decided as in Step 1, namely, α_t is decided so as to satisfy

$$\frac{\alpha_t}{\sum_{w_i(t)\geq\alpha_t} \alpha_t + \sum_{w_i(t)<\alpha_t} w_i(t)} = \left(\frac{1}{k} - \frac{\gamma}{K}\right)/(1-\gamma).$$

```
**Exp3.M**(The extended version of **Exp3** for bandit problems with Multiple plays)
Parameters:  $\gamma \in (0,1]$
Initialization:  $w_i(1) = 1$ for $i = 1, 2, ..., K$
**For each** $t = 1, 2, ...$
    1. **if** $\arg\max_{j\in[K]} w_j(t) \ge \left(\frac{1}{k} - \frac{\gamma}{K}\right)\sum_{i=1}^{K} w_i(t)/(1-\gamma)$ **then**
         Decide $\alpha_t$ so as to satisfy
             $$\frac{\alpha_t}{\sum_{w_i(t)\ge\alpha_t}\alpha_t + \sum_{w_i(t)<\alpha_t} w_i(t)} = \left(\frac{1}{k} - \frac{\gamma}{K}\right)/(1-\gamma).$$
         Set $S_0(t) = \{i : w_i(t) \ge \alpha_t\}$ and $w_i'(t) = \alpha_t$ for $i \in S_0(t)$.
       **else**
         Set $S_0(t) = \emptyset$.
    2. Set
             $$w_i'(t) = w_i(t) \text{ for } i \in \{1, 2, ..., K\} - S_0(t).$$
    3. Set
             $$p_i(t) = k\left((1-\gamma)\frac{w_i'(t)}{\sum_{j=1}^{K} w_j'(t)} + \frac{\gamma}{K}\right) \text{ for } i = 1, 2, ..., K.$$
    4. Set
             $$S(t) = \mathbf{DepRound}(k, (p_1, p_2, \ldots, p_n)).$$
    5. Receive rewards $x_i(t) \in [0,1]$ for $i \in S(t)$.
    6. For $i = 1, 2, ..., K$ set
             $$\hat{x}_i(t) = \begin{cases} x_i(t)/p_i(t) & \text{if } i \in S(t),\\ 0 & \text{otherwise.}\end{cases}$$
             $$w_i(t+1) = \begin{cases} w_i(t)\exp(k\gamma\hat{x}_i(t)/K) & \text{if } i \notin S_0(t),\\ w_i(t) & \text{otherwise.}\end{cases}$$
```

Fig. 4. Pseudocode of algorithm **Exp3.M**

Note that $S_0(t) \subseteq S(t)$ since $p_i(t) = 1$ for all $i \in S_0(t)$. Another point of our algorithm is that weights $w_i(t)$ are not updated for $i \in S_0(t)$ in Step 6, namely, $w_i(t+1) = w_i(t)$ for actions i with relatively too large weights. The bottleneck of the algorithm is the calculation of α_t, which can be calculated in $O(K(\log k + 1))$ time by finding the largest k weights. Therefore, **Exp3.M** runs in $O(K(\log k + 1))$ time at each time step and needs $O(K)$ space.

The following theorem is an extension of Theorem 3.1 in [4].

Theorem 2. *For any $K > 0$ and for any $\gamma \in (0,1]$,*

$$G_{\textit{\textbf{max-k}}} - E[G_{\textit{\textbf{Exp3.M}}}] \le (e-1)\gamma G_{\textit{\textbf{max-k}}} + \frac{K}{\gamma}\ln\frac{K}{k}$$

holds for any assignment of rewards and for any $T > 0$.

Proof. Let W_t, W'_t denote $\sum_{i=1}^K w_i(t), \sum_{i=1}^K w'_i(t)$, respectively. Then, for any $t = 1, 2, ..., T$,

$$\begin{aligned}
\frac{W_{t+1}}{W_t} &= \sum_{i\in[K]-S_0(t)} \frac{w_i(t+1)}{W_t} + \sum_{i\in S_0(t)} \frac{w_i(t+1)}{W_t} \\
&= \sum_{i\in[K]-S_0(t)} \frac{w_i(t)}{W_t}\exp\left(\frac{k\gamma}{K}\hat{x}_i(t)\right) + \sum_{i\in S_0(t)} \frac{w_i(t)}{W_t} \\
&\le \sum_{i\in[K]-S_0(t)} \frac{w_i(t)}{W_t}\left[1+\frac{k\gamma}{K}\hat{x}_i(t)+(e-2)\left(\frac{k\gamma}{K}\hat{x}_i(t)\right)^2\right]+\sum_{i\in S_0(t)}\frac{w_i(t)}{W_t} \quad (1)\\
&= 1 + \frac{W'_t}{W_t}\sum_{i\in[K]-S_0(t)} \frac{w_i(t)}{W'_t}\left[\frac{k\gamma}{K}\hat{x}_i(t) + (e-2)\left(\frac{k\gamma}{K}\hat{x}_i(t)\right)^2\right] \\
&= 1 + \frac{W'_t}{W_t}\sum_{i\in[K]-S_0(t)} \frac{\frac{p_i(t)}{k} - \frac{\gamma}{K}}{1-\gamma}\left[\frac{k\gamma}{K}\hat{x}_i(t) + (e-2)\left(\frac{k\gamma}{K}\hat{x}_i(t)\right)^2\right] \\
&\le 1+\frac{\gamma}{K(1-\gamma)}\sum_{i\in[K]-S_0(t)} p_i(t)\hat{x}_i(t) + \frac{(e-2)k\gamma^2}{K^2(1-\gamma)}\sum_{i\in[K]-S_0(t)} p_i(t)\hat{x}_i(t)^2 \quad (2)\\
&\le 1 + \frac{\gamma}{K(1-\gamma)}\sum_{i\in S(t)-S_0(t)} x_i(t) + \frac{(e-2)k\gamma^2}{K^2(1-\gamma)}\sum_{i\in[K]}\hat{x}_i(t). \quad (3)
\end{aligned}$$

Inequality (1) uses $e^a \le 1 + a + a^2$ for $a \le 1$, inequality (2) holds because $W'_t/W_t \le 1$, and inequality (3) uses the fact that $p_i(t)\hat{x}_i(t) = x_i(t) \le 1$ for $i \in S(t)$ and $p_i(t)\hat{x}_i(t) = 0$ for $i \notin S(t)$. Since $1 + x \le e^x$, we have

$$\ln\frac{W_{t+1}}{W_t} \le \frac{\gamma}{K(1-\gamma)}\sum_{i\in S(t)-S_0(t)} x_i(t) + \frac{(e-2)k\gamma^2}{K^2(1-\gamma)}\sum_{i\in[K]}\hat{x}_i(t).$$

By summing over t, we obtain

$$\ln\frac{W_{T+1}}{W_1} \le \frac{\gamma}{K(1-\gamma)}\sum_{t=1}^{T}\sum_{i\in S(t)-S_0(t)} x_i(t) + \frac{(e-2)k\gamma^2}{K^2(1-\gamma)}\sum_{t=1}^{T}\sum_{i\in[K]}\hat{x}_i(t). \quad (4)$$

On the other hand, for the set $A^* \subset [K]$ of k elements with the maximum total reward $\sum_{j\in A}\sum_{t=1}^T x_j(t)$ among all subsets A containing k elements,

$$\begin{aligned}
\ln\frac{W_{T+1}}{W_1} \ge \ln\frac{\sum_{j\in A^*} w_j(T+1)}{W_1} &\ge \frac{\sum_{j\in A^*}\ln w_j(T+1)}{k} + \ln\frac{k}{K} \quad (5)\\
&= \frac{\gamma}{K}\sum_{j\in A^*}\sum_{t:j\notin S_0(t)}\hat{x}_j(t) + \ln\frac{k}{K}. \quad (6)
\end{aligned}$$

The second inequality in (5) uses the fact that

$$\sum_{j\in A^*} w_j(T+1) \ge k(\prod_{j\in A^*} w_j(T+1))^{1/k}$$

and equation (6) uses $w_j(T+1) = \exp((k\gamma/K)\sum_{t:j\notin S_0(t)} \hat{x}_j(t))$.

From (4) and (6), we get

$$\sum_{j\in A^*}\sum_{t:j\notin S_0(t)} \hat{x}_j(t) + \frac{K}{\gamma}\ln\frac{k}{K} \le \frac{1}{(1-\gamma)}\sum_{t=1}^{T}\sum_{i\in S(t)-S_0(t)} x_i(t) + \frac{(e-2)k\gamma}{K(1-\gamma)}\sum_{t=1}^{T}\sum_{i\in[K]}\hat{x}_i(t).$$

Since $\sum_{j\in A^*}\sum_{t:j\in S_0(t)} x_j(t) \le \frac{1}{1-\gamma}\sum_{t=1}^{T}\sum_{i\in S_0(t)} x_i(t)$ trivially holds, we have

$$\begin{aligned}&\sum_{j\in A^*}\sum_{t:j\notin S_0(t)} \hat{x}_j(t) + \sum_{j\in A^*}\sum_{t:j\in S_0(t)} x_j(t) + \frac{K}{\gamma}\ln\frac{k}{K}\\ &\le \frac{1}{(1-\gamma)}G_{\mathbf{Exp3.M}} + \frac{(e-2)k\gamma}{K(1-\gamma)}\sum_{t=1}^{T}\sum_{i\in[K]}\hat{x}_i(t).\end{aligned}$$

Taking expectation of both sides of this inequality, we obtain

$$G_{\mathbf{max\text{-}k}} + \frac{K}{\gamma}\ln\frac{k}{K} \le \frac{1}{(1-\gamma)}E[G_{\mathbf{Exp3.M}}] + \frac{(e-2)k\gamma}{K(1-\gamma)}\sum_{t=1}^{T}\sum_{i\in[K]} x_i(t)$$

because equation $E[\hat{x}_i(t)|S(1),S(2),\ldots,S(i-1)] = x_i(t)$ holds from the fact that **DepRound** selects action i with probability $p_i(t)$.

From the fact that

$$\sum_{t=1}^{T}\sum_{i=1}^{K} x_i(t) \le \frac{K}{k}G_{\mathbf{max\text{-}k}},$$

we obtain the inequality in the statement of the theorem. □

The following corollary can be obtained by an appropriate choice of parameter γ. The proof is the same as that of Corollary 3.2 in [4].

Corollary 1. *Set* $\gamma = min\left\{1, \sqrt{K\ln(K/k)/((e-1)kT)}\right\}$. *Then*

$$G_{\boldsymbol{max\text{-}k}} - E[G_{\boldsymbol{Exp3.M}}] \le 2\sqrt{(e-1)}\sqrt{kTK\ln\frac{K}{k}} \le 2.63\sqrt{kTK\ln\frac{K}{k}}$$

holds for any $T > 0$ *and for any assignment of rewards.*

6 Lower Bounds on the Regret

Auer et al. showed a lower bound $\Omega(\sqrt{KT})$ on the regret of any player for adversarial bandit problem with single play ($k = 1$). Their theorem can be extended easily to the multiple play setting.

Theorem 3. *For any number of actions K, for any time horizon T and for any $k \in [K]$, there exists a distribution over the assignment of rewards such that*

$$E[G_{\mathbf{max\text{-}k}} - G_A] \geq \min\left\{\frac{1}{5}\left(\frac{K-k}{K}\right)^2\sqrt{KT}, \frac{K-k}{8K}kT\right\},$$

holds for any algorithm A.

Proof. This theorem can be proved by modifying the proof of Theorem 5.1 [4] a little. The reward assignment by the distribution whose existence is insisted by the theorem is made as follows. First, select a set of k actions I according to uniform distribution over $\mathbf{C}([K], k)$. For each $(i,t) \in [K] \times [T]$, independently assign 1 to reward $x_i(t)$ with probability $\frac{1}{2} + \varepsilon$ when $i \in I$ and with probability $\frac{1}{2}$ otherwise, where ε is a small constant value belonging to $(0, \frac{1}{2})$. Value 0 is assigned to $x_i(t)$ with the rest of the probability. Note that this distribution over reward assignment coincides with the one used to prove Theorem 5.1 in [4] when $k = 1$. Let $E_*[\cdot]$ denote expectation of some random variable with respect to this distribution. Then, we can prove

$$E_*[G_{\mathbf{max\text{-}k}} - G_A] \geq k\varepsilon\left(T - \frac{kT}{K} - 2\sqrt{\ln\frac{4}{3}}kT\sqrt{\frac{T}{K}}\varepsilon\right), \tag{7}$$

when $0 \leq \varepsilon \leq 1/4$. By choosing $\varepsilon = (1/4)\min\{(K-k)/(k\sqrt{\ln(4/3)}\sqrt{KT}), 1\}$, the lower bound of this theorem can be obtained.

The proof of Inequality (7) can be done by evaluating a random variable $N_{\mathbf{i}}$ which denotes the number of times action $i \in \mathbf{i}$ $(\in \mathbf{C}([K], k))$ is chosen, namely, $N_{\mathbf{i}} = \sum_{t=1}^{T} |S(t) \cap \mathbf{i}|$. Let $E_{\mathbf{i}}[\cdot]$ denote conditional expectation $E_*[\cdot | I = \mathbf{i}]$. Then,

$$E_*[G_A] = \frac{kT}{2} + \frac{\varepsilon}{\mathbf{C}(K,k)}\sum_{\mathbf{i} \in \mathbf{C}([K],k)} E_{\mathbf{i}}[N_{\mathbf{i}}] \tag{8}$$

holds. We use the following lemma, which is a straightforward extension of Lemma A.1 in [4].

Lemma 1. *Let $f : \{0,1\}^{kT} \to [0, M]$ be any function defined on reward sequences $\mathbf{r}$. Then for any set of actions $\mathbf{i} \in \mathbf{C}([K], k)$,*

$$E_{\mathbf{i}}[f(\mathbf{r})] \leq E_{\boldsymbol{unif}}[f(\mathbf{r})] + \frac{M}{2}\sqrt{-E_{\boldsymbol{unif}}[N_{\mathbf{i}}]\ln(1-4\varepsilon^2)},$$

where $E_{\boldsymbol{unif}}[\cdot]$ is the uniform distribution over assignment of rewards.

By Lemma 1, we obtain

$$E_{\mathbf{i}}[N_{\mathbf{i}}] \leq E_{\mathbf{unif}}[N_{\mathbf{i}}] + \frac{kT}{2}\sqrt{-E_{\mathbf{unif}}[N_{\mathbf{i}}]\ln(1-4\varepsilon^2)}. \tag{9}$$

Combining (8) and (9), and using Jensen's Inequality and the fact that $\sum_{\mathbf{i}\in\mathbf{C}([K],k)} E_{\mathbf{unif}}[N_{\mathbf{i}}] = \mathbf{C}\left(K-1,k-1\right)kT$, we can prove

$$E_*[G_A] \le \frac{kT}{2} + k\varepsilon\left(\frac{kT}{K} + \frac{kT}{2}\sqrt{-\frac{T}{K}\ln(1-4\varepsilon^2)}\right).$$

Inequality (7) can be derived from this inequality using the fact that $E_*[G_{\mathbf{max\text{-}k}}] \ge kT(1/2+\varepsilon)$ and the inequality $-\ln(1-x) \le (4\ln(4/3))x$ for $x \in [0,1/4]$. □

Remark 1. When $k = K$, there exists only one strategy, which means 0-regret for all algorithms. In this case, our upper bound shown in Corollary 1 and our lower bound shown in Theorem 3 become 0.

7 Efficient *k*-Combination Selection

In this section, we describe how to select efficiently a set of k distinct actions from $[K]$ so as to satisfy that each action i is selected with probability p_i for any given distribution $(p_1,p_2,\ldots,p_K)$ with $0 \le p_i \le 1$ for $i = 1,2,\ldots,K$ and $\sum_{i=1}^K p_i = k$. How to realize this selection plays an important role for efficient implementations of both algorithms **BOLOM** and **Exp3.M**. One solution is to select a set S according to a distribution q_S for $S \in \mathbf{C}\left([K],k\right)$ after solving the following problem.

Problem 1. For a given $(p_1,p_2,\ldots,p_K) \in [0,1]^K$ with $\sum_{i=1}^K p_i = k$, find q_S for $S \in \mathbf{C}\left([K],k\right)$ satisfying

$$\begin{aligned}\sum_{i\in S} q_S = p_i(t) \quad &\text{for all} \quad i = 1,2,\ldots,K,\\ 0 \le q_S \le 1 \qquad &\text{for all} \quad S \in \mathbf{C}\left([K],k\right).\end{aligned}$$

Note that Problem 1 always has a solution by Proposition 1. However, solving Problem 1 is not a good idea because it takes at least $\mathbf{C}\left(K,k\right) = \Omega(K^k)$ time for fixed k using a general solver. Note that $\sum_{i\in S} q_S = p_i$ for $i = 1,2,\ldots,K$ can be expressed as $A\boldsymbol{q} = \boldsymbol{p}$ using $K \times \mathbf{C}\left(K,k\right)$ matrix A, $\mathbf{C}\left(K,k\right)$-dimensional vector $\boldsymbol{q} = \left(\cdots,q_S,\cdots\right)^\top$ and K-dimensional vector $\boldsymbol{p} = \left(p_1,\cdots,p_K\right)^\top$ and for any mutually independent K column vectors $\boldsymbol{a}_{S_1},\boldsymbol{a}_{S_2},\ldots,\boldsymbol{a}_{S_K}$ of A, there is a solution of $A\boldsymbol{q} = \boldsymbol{p}$ whose variables are zero except $q_{S_1},q_{S_2},\ldots,q_{S_K}$, but it is not guaranteed that $0 \le q_{S_i} \le 1$ holds for all $i = 1,2,\ldots,K$.

Fortunately, there is a nice technique called *dependent rounding* [6], which can efficiently select a set of k distinct actions from $[K]$ while satisfying the condition that each action i is selected with probability p_i. Dependent rounding developed by Gandhi et al. [6] is a kind of technique that randomly selects a set of edges from a bipartite graph under some cardinality constraints. Our selection problem is a special case of the problems they considered, the case that the bipartite graph is a star. In such case, the algorithm can be described as shown in Fig. 5, which we call **DepRound** here. In the algorithm, $(p_1,p_2,...,p_K)$ is

```
DepRound % Dependent Rounding
Inputs: Natural number $k(< K)$, $(p_1, p_2, ..., p_K)$ with $\sum_{i=1}^K p_i = k$
Output: Subset of $[K]$ with $k$ elements
  while there is an $i$ with $0 < p_i < 1$ do
    Choose distinct $i$ and $j$ with $0 < p_i < 1$ and $0 < p_j < 1$
    Set $\alpha = \min\{1 - p_i, p_j\}$ and $\beta = \min\{p_i, 1 - p_j\}$
    Update $p_i$ and $p_j$ as
      $(p_i, p_j) = \begin{cases} (p_i + \alpha, p_j - \alpha) \text{ with probability } \frac{\beta}{\alpha+\beta} \\ (p_i - \beta, p_j + \beta) \text{ with probability } \frac{\alpha}{\alpha+\beta} \end{cases}$
  end while
  return $\{i : p_i = 1, 1 \le i \le K\}$
```

Fig. 5. Pseudocode of algorithm **DepRound** [6]

Table 1. Performance Comparison of the algorithms for multiple play setting

Algorithm	Base Algorithm	regret	time comp.	space comp.
Exp3 [4]	-	$O(k^{\frac{3}{2}} T^{\frac{1}{2}} K^{\frac{k}{2}} \sqrt{\log K})$	$\Omega(K^k)$	$\Omega(K^k)$
ComBand [5]	-	$O(k^{\frac{3}{2}} \sqrt{TK \ln K})$	$O(kK^3)$	$O(K^3)$
BOLOM	**Algorithm 1** [1]	$O(kK^{\frac{3}{2}} \sqrt{T \log T})$	$O(K^3)$	$O(K^2)$
Exp3.M	**Exp3** [4]	$O(\sqrt{kTK \log(K/k)})$	$O(K(\log k + 1))$	$O(K)$

probabilistically updated until all the components are 0 or 1 while keeping the condition that $\sum_{i=1}^K p_i = k$. The inside of the while-loop is executed at most K times because at least one of p_i and p_j becomes 0 or 1 in each time of the execution. The nice property of the update is to keep the expectation values of p_i, namely, $E[p_i^{t+1}] = E[p_i^t]$ for every $i \in [K]$, where p_i^t denotes p_i after the tth execution of the inside of the while-loop. This is trivial when i is not chosen at the tth execution, and holds even when i is chosen since

$$(p_i + \alpha) \times \frac{\beta}{\alpha + \beta} + (p_i - \beta) \times \frac{\alpha}{\alpha + \beta} = (p_i - \alpha) \times \frac{\beta}{\alpha + \beta} + (p_i + \beta) \times \frac{\alpha}{\alpha + \beta} = p_i.$$

Each execution of the inside of the while-loop needs a constant time, so **DepRound** runs in $O(K)$ time and $O(K)$ space.

8 Concluding Remarks

We have extended adversarial bandit problems studied by Auer et al. [4] to those with multiple plays, and analyzed algorithms for the problem.

From the result shown in Table 1, we can know that **Exp3.M** is the best algorithm for this problem among the four algorithms analyzed here, which might be caused by the difference of used information; only **Exp3.M** uses each rewards for selected k actions. We are now interested in applying our algorithms to real problems and demonstrating their practical usefulness.

Acknowledgements

We would like to thank anonymous referees for a lot of their helpful comments. This work was partially supported by JSPS KAKENHI 21500128.

References

[1] Abernethy, J., Hazan, E., Rakhlin, A.: Competing in the dark: An efficient algorithm for bandit linear optimization. In: Proceedings of the 21st Annual Conference on Learning Theory, COLT 2008 (2008)
[2] Agrawal, R., Hegde, M.V., Teneketzis, D.: Multi-armed bandits with multiple plays and switching cost. Stochastic and Stochastic Reports 29, 437–459 (1990)
[3] Anantharam, V., Varaiya, P., Walrand, J.: Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays –part i: I.i.d. rewards. IEEE Transactions on Automatic Control 32, 968–976 (1986)
[4] Auer, P., Cesa-bianchi, N., Freund, Y., Schapire, R.E.: The nonstochastic multiarmed bandit problem. SIAM Journal on Computing 32, 48–77 (2002)
[5] Cesa-bianchi, N., Lugosi, G.: Combinatorial bandits. In: Proceedings of the 22nd Annual Conference on Learning Theory, COLT 2009 (2009)
[6] Gandhi, R., Khuller, S., Parthasarathy, S., Srinivasan, A.: Dependent rounding and its applications to approximation algorithms. Journal of the ACM 53(3), 320–360 (2006)
[7] György, A., Linder, T., Lugosi, G., Ottucsák, G.: The on-line shortest path problem under partial monitoring. Journal of Machine Learning Research 8, 2369–2403 (2007)
[8] Kleinberg, R.: Notes from week 8: Multi-armed bandit problems. CS 683–Learning, Games, and Electronic Markets (2007), http://www.cs.cornell.edu/courses/cs683/2007sp/lecnotes/week8.pdf
[9] Krein, M., Milman, D.: On extreme points of regular convex sets. Studia Mathematica, 133–138 (1940)
[10] Mahajan, A., Teneketzis, D.: Multi-armed bandit problems. In: Foundations and Applications of Sensor Management, pp. 121–151. Springer, Heidelberg (2007)
[11] Nakamura, A., Abe, N.: Improvements to the linear programming based scheduling of web advertisements. Electronic Commerce Research 5, 75–98 (2005)
[12] Niculescu-Mizil, A.: Multi-armed bandits with betting. In: COLT 2009 Workshop, pp. 133–138 (2009)
[13] Pandelis, D.G., Tenekezis, D.: On the optimality of the gittins index rule in multiarmed bandits with multiple plays. Mathematical Methods of Operations Research 50, 449–461 (1999)
[14] Song, N.O., Teneketzis, D.: Discrete search with multiple sensors. Mathematical Methods of Operations Research 60, 1–14 (2004)
[15] Uchiya, T., Nakamura, A., Kudo, M.: Adversarial bandit problems with multiple plays. In: The IEICE Technical Report, COMP2009-27 (2009)
[16] Warmuth, M.K., Takimoto, E.: Path kernels and multiplicative updates. Journal of Machine Learning Research, 773–818 (2003)

Online Multiple Kernel Learning: Algorithms and Mistake Bounds

Rong Jin[1], Steven C.H. Hoi[2], and Tianbao Yang[1]

[1] Department of Computer Science and Engineering,
Michigan State University, MI, 48824, USA
[2] School of Computer Engineering,
Nanyang Technological University, 639798, Singapore
{rongjin,yangtia1}@cse.msu.edu, chhoi@ntu.edu.sg

Abstract. *Online learning* and *kernel learning* are two active research topics in machine learning. Although each of them has been studied extensively, there is a limited effort in addressing the intersecting research. In this paper, we introduce a new research problem, termed **Online Multiple Kernel Learning** (OMKL), that aims to learn a kernel based prediction function from a pool of predefined kernels in an online learning fashion. OMKL is generally more challenging than typical online learning because both the kernel classifiers and their linear combination weights must be learned simultaneously. In this work, we consider two setups for OMKL, i.e. combining binary predictions or real-valued outputs from multiple kernel classifiers, and we propose both deterministic and stochastic approaches in the two setups for OMKL. The deterministic approach updates all kernel classifiers for every misclassified example, while the stochastic approach randomly chooses a classifier(s) for updating according to some sampling strategies. Mistake bounds are derived for all the proposed OMKL algorithms.

Keywords: On-line learning and relative loss bounds, Kernels.

1 Introduction

In recent years, we have witnessed increasing interests on both *online learning* and *kernel learning*. Online learning refers to the learning process of answering a sequence of questions given the feedback of correct answers to previous questions and possibly some additional prior information [26]; Kernel learning aims to identify an effective kernel for a given learning task [19, 27, 12]. A well-known kernel learning method is Multiple Kernel Learning (MKL) [3, 27], that seeks the combination of multiple kernels in order to optimize the performance of kernel based learning methods (e.g., Support Vector Machines (SVM)).

Although kernel trick has been explored in online learning [10, 7], it is often assumed that kernel function is given apriori. In this work, we address a a new research problem, **Online Multiple Kernel Learning** (OMKL), which aims to simultaneously learn multiple kernel classifiers and their linear combinations from a pool of given kernels in an online fashion. Compared to the exiting methods for multiple kernel learning (see [31] and reference therein), online multiple kernel learning is computationally advantageous in that it only requires going through training examples once. We emphasize

© Springer-Verlag Berlin Heidelberg 2010

that online multiple kernel learning is significantly more challenging than typical online learning because both the optimal kernel classifiers and their linear combinations have to be learned simultaneously in an online fashion.

In this paper, we consider two different setups for online multiple kernel learning. In the first setup, termed as *Online Multiple Kernel Learning by Predictions* or OMKL-P, its objective is to combine the *binary predictions* from multiple kernel classifiers. The second setup, termed as *Online Multiple Kernel Learning by Outputs* or OMKL-O, improves OMKL-P by combining the *real-valued outputs* from multiple kernel classifiers. Our online learning framework for multiple kernel learning is based on the combination of two types of online learning techniques: the *Perceptron* algorithm [24] that learns a classifier for a given kernel, and the *Hedge* algorithm [9] that linearly combines multiple classifiers. Based on the proposed framework, we present two types of approaches for each setup of OMKL , i.e., *deterministic* and *stochastic* approaches. The deterministic approach updates each kernel classifier for every misclassified example, while the stochastic approach chooses a subset of classifiers for updating based on certain sampling strategies. Mistake bounds are derived for all the proposed algorithms for online kernel learning.

The rest of this paper is organized as follows. Section 2 reviews the related work on both online learning and kernel learning. Section 3 overviews the problem of online multiple kernel learning. Section 4 presents the algorithms for Online Multiple Kernel Learning by Predictions and their mistake bounds; Section 5 presents algorithms for Online Multiple Kernel Learning by Outputs and their mistake bounds. Section 6 concludes this study with future work.

2 Related Work

Our work is closely related to both *online learning* and *kernel learning*. Below we briefly review the important work in both areas.

Extensive studies have been devoted to online learning for classification. Starting from Perceptron algorithm [1, 24, 22], a number of online classification algorithms have been proposed including the ROMMA algorithm [20], the ALMA algorithm [11], the MIRA Algorithm [8], the NORMA algorithm [16, 15], and the online Passive-Aggressive algorithms [7]. Several studies extended the Perceptron algorithm into a nonlinear classifier by the introduction of kernel functions [16, 10]. Although these algorithms are effective for nonlinear classification, they usually assume that appropriate kernel functions are given apriori, which limits their applications. Besides online classification, our work is also related to online prediction with expert advices [9, 21, 29]. The most well-known work is probably the Hedge algorithm [9], which was a direct generalization of Littlestone and Warmuth's Weighted Majority (WM) algorithm [21]. We refer readers to the book [4] for the in-depth discussion of this subject.

Kernel learning has been actively studied thanks to the great successes of kernel methods, such as support vector machines (SVM) [28, 25]. Recent studies of kernel learning focus on learning an effective kernel automatically from training data. Various algorithms have been proposed to learn parametric or semi-parametric kernels from labeled and/or unlabeled data. Example techniques include cluster kernels [5], diffusion kernels [17], marginalized kernels [14], graph-based spectral kernel learning approaches [32, 13], non-parameric kernel learning [12, 6], and lower-rank kernel learning[18]. Among various approaches for kernel learning, Multiple Kernel Learning(MKL) [19], whose goal is to learn an optimal combination of multiple kernels, has emerged as a promising technique. A number of approaches have been proposed to solve the optimization problem related to MKL, including the conic combination approach via regular convex optimization [19], the semi-infinite linear program (SILP) approach [27], the Subgradient Descent approach [23], and the recent level method [30].

We emphasize that although both online learning and kernel learning have been extensively studied, little work has been done to address online kernel learning, especially online multiple kernel learning. To the best of our knowledge, this is the first theoretic study that addresses the OMKL problem.

3 Online Multiple Kernel Learning

Before presenting the algorithms for online multiple kernel learning, we first briefly describe the Multiple Kernel Learning (MKL) problem. Given a set of training examples $\mathcal{D}_T = \{(x_t, y_t), t = 1, \ldots, T\}$ where $y_t \in \{-1, +1\}, t = 1, \ldots, T$, and a collection of kernel functions $\mathcal{K}_m = \{\kappa_i : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, i = 1, \ldots, m\}$, our goal is to identify the optimal combination of kernel functions, denoted by $\mathbf{u} = (u_1, \cdots, u_m)^\top$, that minimizes the margin classification error. It is cast as the following optimization problem:

$$\min_{\mathbf{u} \in \Delta} \min_{f \in \mathcal{H}_{\kappa_\mathbf{u}}} \frac{1}{2}\|f\|^2_{\mathcal{H}_{\kappa_\mathbf{u}}} + C \sum_{t=1}^{T} \ell(f(x_t), y_t) \tag{1}$$

where $\mathcal{H}_\kappa$ denotes the reproducing kernel Hilbert space defined by kernel κ, Δ denotes a simplex, i.e. $\Delta = \{\theta \in \mathbb{R}^m_+ | \sum_{i=1}^m \theta_i = 1\}$, and

$$\kappa_\mathbf{u}(\cdot, \cdot) = \sum_{j=1}^{m} u_j \kappa_j(\cdot, \cdot), \ \ell(f(x_t), y_t) = \max(0, 1 - y_t f(x_t))$$

It can also be cast into the following minimax problem:

$$\min_{\mathbf{u} \in \Delta} \max_{\alpha \in [0,C]^T} \left\{ \sum_{t=1}^{T} \alpha_t - \frac{1}{2}(\alpha \circ \mathbf{y})^\top \left(\sum_{i=1}^{m} u_i K^i \right) (\alpha \circ \mathbf{y}) \right\} \tag{2}$$

where $K^i \in \mathbb{R}^{n \times n}$ with $K^i_{j,l} = \kappa_i(x_j, x_l)$, $\mathbf{y} = (y_1, \cdots, y_T)^\top$, and $\circ$ is the element-wise product between two vectors.

The formulation for batch mode multiple kernel learning in (1) aims to learn a single function in the space of $\mathcal{H}_{\kappa_\mathbf{u}}$. It is well recognized that solving the optimization problem in (1) is in general computationally expensive. In this work, we aim to alleviate the computational difficulty of multiple kernel learning by online learning that only needs to scan through training examples once.

The following theorem allows us to simplify this problem by decomposing it into two separate tasks, i.e., learning (i) a classifier for each individual kernel, and (ii) weights that combine the outputs of individual kernel classifier to form the final prediction.

Theorem 1. *The optimization problem in (1) is equivalent to*

$$\min_{\mathbf{u}\in\Delta,\{f_i\in\mathcal{H}_{\kappa_i}\}_{i=1}^m} \sum_{i=1}^m \frac{u_i}{2}\|f_i\|^2_{\mathcal{H}_{\kappa_i}} + C\sum_{t=1}^T \ell\left(\sum_{i=1}^m u_i f_i(x_t), y_t\right) \tag{3}$$

Proof. It is important to note that problem in (3) is non-convex, and therefore we can not directly deploy the standard approach to convert it into its dual form. In order to transform (3) into (1), we rewrite $\ell(z,y) = \max\limits_{\alpha\in[0,1]} \alpha(1-yz)$, and rewrite (3) as follows

$$\min_{\mathbf{u}\in\Delta} \min_{\{f_i\in\mathcal{H}_{\kappa_i}\}_{i=1}^m} \max_{\alpha\in[0,C]^T} \sum_{i=1}^m \frac{u_i}{2}\|f_i\|^2_{\mathcal{H}_{\kappa_i}} + \sum_{t=1}^T \alpha_t\left(1 - y_t\sum_{i=1}^m u_i f_i(x_t)\right)$$

Since the problem is convex in f_i and concave in α, we can switch the minimization of f_i with the maximization of α. By taking the minimization of f_i, we have

$$f_i(\cdot) = \sum_{t=1}^T \alpha_t y_t \kappa_i(x_t,\cdot), i = 1,\ldots,m$$

and the resulting optimization problem becomes

$$\min_{\mathbf{u}\in\Delta} \max_{\alpha\in[0,C]^T} \sum_{t=1}^T \alpha_t - \sum_{i=1}^m \frac{u_i}{2}(\alpha\circ\mathbf{y})^\top K^i(\alpha\circ\mathbf{y}),$$

which is identical to the optimization problem of batch mode multiple kernel learning in (2).

Based on the results of the above theorem, our strategy toward online kernel learning is to simultaneously learn a set of kernel classifiers $f_i, i = 1,\ldots,m$ and their combination weighs $\mathbf{u}$. We consider two setups for Online Multiple Kernel Learning (OMKL). In the first setup, termed Online Multiple Kernel Learning by Predictions (OMKL-P), we simplify the problem by only considering combining the binary predictions from multiple kernel classifiers, i.e., $\hat{y} = \sum_{i=1}^m u_i \text{sign}(f_i(x))$. In the second setup, termed Online Multiple Kernel Learning by Outputs (OMKL-O), we learn to combine the real-valued outputs from multiple kernel classifiers, i.e. $\hat{y} = \sum_{i=1}^m u_i f_i(x)$. In the next two sections, we will discuss algorithms and theoretic properties for OMKL-P and OMKL-O, respectively.

For the convenience of analysis, throughout the paper, we assume $\kappa_i(x,x) \leq 1$ for all the kernel functions $\kappa_i(\cdot,\cdot)$ and for any example x. Below we summarize the notations that are used throughout this paper:

- $\mathcal{D}_T = \{(x_t,y_t), t = 1,\ldots,T\}$ denotes a sequence of T training examples. $\mathcal{K}_m = \{\kappa_i : \mathcal{X}\times\mathcal{X}\to\mathbb{R}, i = 1,\ldots,m\}$ denotes a collection of m kernel functions.
- $\mathbf{f}_t(\cdot) = (f_1^t(\cdot),\cdots,f_m^t(\cdot))^\top$ denotes the collection of m classifiers in round t, where $f_i^t(\cdot)$ represents the classifier learned using the kernel $\kappa_i(\cdot,\cdot)$. For the purpose of presentation, we use $\mathbf{f}_t, f_i^t$ for short. $\mathbf{f}_t(x) = (f_1^t(x),\cdots,f_m^t(x))^\top$ denotes the real-valued outputs on example x by the m classifiers learned in round t, and $\text{sign}(\mathbf{f}_t(x)) = (\text{sign}(f_1^t(x)),\cdots,\text{sign}(f_m^t(x)))^\top$ denotes the binary predictions by the corresponding classifiers on example x.

– $\mathbf{w}_t = (w_1^t, \cdots, w_m^t)^\top$ denotes the weight vector for the m classifiers in round t; $W_t = \sum_{i=1}^m w_i^t$ represents the sum of weights in round t; $\mathbf{q}_t = (q_1^t, \ldots, q_m^t)^\top$ denotes the normalized weight vector, i.e. $q_i^t = w_i^t / W_t$.
– $\mathbf{z}_t = (z_1^t, \cdots, z_m^t)^\top$ denotes the indicator vector, where $z_i^t = I(y_t f_i^t(x_t) \leq 0)$ indicates if the ith kernel classifier makes a mistake on example x_t, where $I(C)$ outputs 1 when C is true and zero otherwise.
– $\mathbf{m}_t = (m_1^t, \cdots, m_m^t)^\top$ denotes the 0-1 random variable vector, where $m_i^t \in \{0, 1\}$ indicates if the ith kernel classifier is chosen for updating in round t.
– $\mathbf{p}_t = (p_1^t, \cdots, p_m^t)^\top$ denotes a probability vector, i.e. $p_i^t \in [0, 1]$.
– $\mathbf{a} \cdot \mathbf{b}$ denotes the dot-product between vector $\mathbf{a}$ and vector $\mathbf{b}$, $\mathbf{1}$ denotes a vector with all elements equal to 1, and $\mathbf{0}$ denotes a vector with all elements equal to 0.
– Multi_Sample($\mathbf{p}_t$) denotes a multinomial sampling process following the probability distribution $\mathbf{p}_t \in \Delta$ that outputs $i_t \in \{1, \ldots, m\}$. Bern_Sample(p_i^t) denotes a Bernoulli sampling process following the probability p_i^t that outputs a binary variable $m_i^t \in \{1, 0\}$.

4 Algorithms for Online Kernel Learning by Predictions (OMKL-P)

4.1 Deterministic Approaches(DA)

As already pointed out, the main challenge of OMKL is that both the kernel classifiers and their combination are unknown. The most straightforward approach is to learn a classifier for each individual kernel function and decide its combination weight based on the number of mistakes made by the kernel classifier. To this end, we combine the Perceptron algorithm and the Hedge algorithm together. In particular, for each kernel, the Perceptron algorithm is employed to learn a classifier, and the Hedge algorithm is used to update its weight. Algorithm 1 shows the deterministic algorithm for OMKL-P.

The theorem below shows the mistake bound for Algorithm 1. For the convenience of presentation, we define the optimal margin error for kernel $\kappa_i(\cdot, \cdot)$ with respect to a collection of training examples $\mathcal{D}_T$ as:

$$g(\kappa_i, \ell) = \min_{f \in \mathcal{H}_{\kappa_i}} \left(\|f\|^2_{\mathcal{H}_{\kappa_i}} + 2 \sum_{t=1}^T \ell(f(x_t), y_t) \right)$$

Theorem 2. *After receiving a sequence of T training examples $\mathcal{D}_T$, the number of mistakes made by running Algorithm 1 is bounded as follows*

$$M = \sum_{t=1}^T I(\mathbf{q}_t \cdot \mathbf{z}_t \geq 0.5) \leq \frac{2 \ln(1/\beta)}{1 - \beta} \min_{1 \leq i \leq m} g(\kappa_i, \ell) + \frac{2 \ln m}{1 - \beta} \quad (4)$$

The proof for the theorem as well as the following theorems is sketched in the Appendix.

Note that in Algorithm 1 the weight for each individual kernel classifier is based on whether they classify the training example correctly. An alternative approach for updating the weights is to take into account the output values of $\{f_i^t\}_{i=1}^m$ by penalizing a

Algorithm 1. DA for OMKL-P (1)

```
1: INPUT:
     – Kernels: $\mathcal{K}_m$
     – Discount weight: $\beta \in (0,1)$
2: Initialization: $\mathbf{f}_1 = \mathbf{0}, \mathbf{w}_1 = \mathbf{1}$
3: for $t = 1, 2, \ldots$ do
4:    Receive an instance $x_t$
5:    Predict: $\hat{y}_t = \text{sign}\left(\mathbf{q}_t \cdot \text{sign}(\mathbf{f}_t(x_t))\right)$
6:    Receive the class label $y_t$
7:    for $i = 1, 2, \ldots, m$ do
8:       Set $z_i^t = I(y_t f_i^t(x_t) \leq 0)$
9:       Update $w_i^{t+1} = w_i^t \beta^{z_i^t}$
10:      Update $f_i^{t+1} = f_i^t + z_i^t y_t \kappa_i(x_t, \cdot)$
11:   end for
12: end for
```

Algorithm 2. DA for OMKL-P (2)

```
1: INPUT:
     – Kernels: $\mathcal{K}_m$
     – Discount weight: $\beta \in (0,1)$
     – Max-misclassification level: $\gamma > 0$
2: Initialization: $\mathbf{f}_1 = \mathbf{0}, \mathbf{w}_1 = \mathbf{1}$
3: for $t = 1, 2, \ldots$ do
4:    Receive an instance $x_t$
5:    Predict: $\hat{y}_t = \text{sign}\left(\mathbf{q}_t \cdot \text{sign}(\mathbf{f}_t(x_t))\right)$
6:    Receive the class label $y_t$
7:    for $i = 1, 2, \ldots, m$ do
8:       Set $z_i^t = I(y_t f_i^t(x_t) \leq 0)$, $\nu_i^t = z_i^t(1/2 + \min(\gamma, -y_t f_i^t(x_t)))$
9:       Update $w_i^{t+1} = w_i^t \beta^{\nu_i(t)}$
10:      Update $f_i^{t+1} = f_i^t + z_i^t y_t \kappa_i(x_t, \cdot)$
11:   end for
12: end for
```

kernel classifier more if its degree of misclassification, measured by $-y_t f_i^t(x_t)$, is large. To this end, we present the second version of the deterministic approach for OMKL-P in Algorithm 2 that takes into account the value of $\{f_i^t\}_{i=1}^m$ when updating weights $\{w_i\}_{i=1}^m$. In this alternate algorithm, we introduce the parameter γ, which can be interpreted as the maximum level of misclassification. The key quantity introduced in Algorithm 2 is ν_i^t that measures the degree of misclassification by $1/2 + \min(\gamma, -y_t f_i^t(x))$. Note that we did not directly use $-y_t f_i^t(x)$ for updating weights $\{w_i\}_{i=1}^m$ because it is unbounded.

Theorem 3. *After receiving a sequence of T training examples $\mathcal{D}_T$, the number of mistakes made by Algorithm 2 is bounded as follows*

$$M = \sum_{t=1}^{T} I\left(\mathbf{q}_t \cdot \mathbf{z}_t \geq 0.5\right) \leq \frac{2(1/2+\gamma)\ln(1/\beta)}{1-\beta^{1/2+\gamma}} \min_{1 \leq i \leq m} g(\kappa_i, \ell) + \frac{4(1/2+\gamma)\ln m}{1-\beta^{1/2+\gamma}}$$

The proof is essentially similar to that of Theorem 2 with the modification that addresses variable $\nu_i(t)$ introduced in Algorithm 2.

One problem with Algorithm 1 is how to decide an appropriate value for β. A straightforward approach is to choose β that minimizes the mistake bound, leading to the following corollary.

Corollary 4. *By choosing $\beta = \frac{\sqrt{\min_{1\leq i\leq m} \sum_{t=1}^T z_i^t}}{\sqrt{\min_{1\leq i\leq m} \sum_{t=1}^T z_i^t} + \sqrt{\ln m}}$, we have the following mistake bound*

$$M \leq 2\left(\min_{1\leq i\leq m} g(\kappa_i, \ell) + \ln m + 2\sqrt{\min_{1\leq i\leq m} g(\kappa_i, \ell)\ln m}\right)$$

Proof. Followed by inequality in (4), we have

$$M \leq \frac{2}{\beta} \min_{1\leq i\leq m} \sum_{t=1}^{T} z_i^t + \frac{2\ln m}{1-\beta}$$

where we use $\ln(1/\beta) \leq 1/\beta - 1$. By setting the derivative of the above upper bound with respect to β to zero, and using the inequality $\sum_{t=1}^{T} z_i^t \leq g(\kappa_i, \ell)$ as shown in the appendix, we have the result.

Directly using the result in Corollary 4 is unpractical because it requires foreseeing the future to compute the quantity $\min\limits_{1\leq i\leq m} \sum_{t=1}^{T} z_i^t$. We resolve this problem by exploiting the doubling trick [4]. In particular, we divide the sequence $1, 2, \ldots, T$ into s segments:

$$[T_0 + 1 = 1, T_1], [T_1 + 1, T_2], \ldots, [T_{s-1} + 1, T_s = T]$$

such that $\min\limits_{1\leq i\leq m} \sum_{t=T_k+1}^{T_{k+1}} z_i^t \leq 2^k$ for $k = 0, \ldots, s-2, s-1$, and equality holds for $k = 0, 1, \cdots, s-2$. Now, for each segment $[T_k + 1, T_{k+1}]$, we introduce a different β, denote by β_k, and set its value as

$$\beta_k = \frac{2^{k/2}}{\sqrt{\ln m} + 2^{k/2}}, \quad k = 0, \ldots, s-1 \tag{5}$$

The following theorem shows the mistake bound of Algorithm 1 with such β.

Theorem 5. *By running Algorithm 1 with β_k specified in (5), we have the following mistake bound*

$$\begin{aligned} M \leq & 2\left(\min_{1\leq i\leq m} g(\kappa_i, \ell) + \ln m + \frac{2}{\sqrt{2}-1}\sqrt{\min_{1\leq i\leq m} g(\kappa_i, \ell)\ln m}\right) \\ & + 2\left\lceil \log_2\left(\min_{1\leq i\leq m} g(\kappa_i, \ell)\right)\right\rceil \ln m \end{aligned}$$

where $\lceil x \rceil$ computes the smallest integer that is larger than or equal to x.

4.2 Stochastic Approaches

The analysis in the previous section allows us to bound the mistakes when classifying examples with a mixture of kernels. The main shortcoming with the deterministic approach is that in each round, all the kernel classifiers have to be checked and potentially updated if the training example is classified incorrectly. This could lead to a high computational cost when the number of kernels is large. In this section, we present stochastic approaches for online multiple kernel learning that explicitly address this challenge.

Single Update Approach(SUA). Algorithm 3 shows a stochastic algorithm for OMKL-P. In each round, instead of checking every kernel classifier, we sample a single kernel classifier according to the weights that are computed based on the number of mistakes made by individual kernel classifiers. However, it is important to note that rather than using q_i^t directly to sample one classifier to update, we add a smoothing term δ/m to the sampling probability p_i^t, This smoothing term guarantees a low bound for p_i^t, which ensures that each kernel classifier will be explored with at least certain amount of probability, which is similar to methods for the multi-arm bandit problem [2] to ensure the tradeoff between exploration and exploitation. The theorem below shows the mistake bound of Algorithm 3.

Theorem 6. *After receiving a sequence of T training examples $\mathcal{D}_T$, the expected number of mistakes made by Algorithm 3 is bounded as follows*

$$\mathrm{E}[M] = \mathrm{E}\left[\sum_{t=1}^{T} I\left(\mathbf{q}_t \cdot \mathbf{z}_t \geq 0.5\right)\right] \leq \frac{2m\ln(1/\beta)}{\delta(1-\beta)} \min_{1\leq i\leq m} g(\kappa_i, \ell) + \frac{2m\ln m}{\delta(1-\beta)}$$

Remark. Comparing to the mistake bound in Theorem 2 by Algorithm 1, the mistake bound by Algorithm 3 is amplified by a factor of m/δ due to the stochastic procedure of updating one out of m kernel classifiers. The smoothing parameter δ essentially controls the tradeoff between efficacy and efficiency. To see this, we note that the bound for the expected number of mistakes is inversely proportional to δ; in contrast, the bound for the expected number of updates $\mathrm{E}\left[\sum_{t=1}^T \sum_{i=1}^m m_i^t z_i^t\right] = \mathrm{E}\left[\sum_{t=1}^T \sum_{i=1}^m p_i^t z_i^t\right] \leq (1-\delta)\mathrm{E}\left[\sum_{t=1}^T \sum_{i=1}^m q_i^t z_i^t\right] + \delta T$ has a leading term δT when δ is large, which is proportional to δ.

Multiple Updates Approach(MUA). Compared with the deterministic approaches, the stochastic approach, i.e. the single update algorithm, does significantly improve the computational efficiency. However, one major problem with the single update algorithm is that in any round, only one single kernel classifier is selected for updating. As a result, the unselected but possibly effective kernel classifiers lose their opportunity for updating. This issue is particularly critical at the beginning of an online multiple kernel learning task where most individual kernel classifiers could perform poorly.

In order to make a better tradeoff between efficacy and efficiency, we develop another stochastic algorithm for online multiple kernel learning. The main idea of this new algorithm is to randomly choose multiple kernel classifiers for updating and predictions. Instead of choosing a kernel classifier from a multinomial distribution, the updating of each individual kernel classifier is determined by a separate Bernoulli distribution governed by p_i^t for each classifier. The detailed procedure is shown in Algorithm 4. The theorem below shows the mistake bound of the multiple updates algorithm.

Theorem 7. *After receiving a sequence of T training examples $\mathcal{D}_T$, the expected number of mistakes made by Algorithm 4 is bounded as follows*

$$\mathrm{E}[M] = \mathrm{E}\left[\sum_{t=1}^{T} I\left(\mathbf{q}_t \cdot \mathbf{z}_t \geq 0.5\right)\right] \leq \frac{2\ln(1/\beta)}{\delta(1-\beta)} \min_{1\leq i\leq m} g(\kappa_i, \ell) + \frac{2\ln m}{\delta(1-\beta)}$$

Algorithm 3. SUA for OMKL-P

```
1: INPUT:
     – Kernels: $\mathcal{K}_m$
     – Discount weight: $\beta \in (0,1)$
     – Smoothing parameter: $\delta \in (0,1)$
2: Initialization: $\mathbf{f}_1 = 0, \mathbf{w}_1 = \mathbf{1}, \mathbf{p}_1 = \frac{\mathbf{1}}{m}$
3: for $t = 1, 2, \ldots$ do
4:    Receive an instance $x_t$
5:    Predict: $\hat{y}_t = \text{sign}\Big(\mathbf{q}_t \cdot \text{sign}(\mathbf{f}_t(x_t))\Big)$
6:    Receive the class label $y_t$
7:    $i_t$=Multi_Sample($\mathbf{p}_t$)
8:    for $i = 1, 2, \ldots, m$ do
9:       Set $m_i^t = I(i = i_t)$
10:      Set $z_i^t = I(y_t f_i^t(x_t) \leq 0)$
11:      Update $w_i^{t+1} = w_i^t \beta^{m_i^t z_i^t}$
12:      Update $f_i^{t+1} = f_i^t + m_i^t z_i^t y_t \kappa_i(x_t, \cdot)$
13:   end for
14:   Update $\mathbf{p}_{t+1} = (1-\delta)\mathbf{q}_{t+1} + \delta\mathbf{1}/m$
15: end for
```

Algorithm 4. MUA for OMKL-P

```
1: INPUT:
     – Kernels: $\mathcal{K}_m$
     – Discount weight: $\beta \in (0,1)$
     – Smoothing parameter: $\delta \in (0,1)$
2: Initialization: $\mathbf{f}_1 = \mathbf{0}, \mathbf{w}_1 = \mathbf{1}, \mathbf{p}_1 = \mathbf{1}$
3: for $t = 1, 2, \ldots$ do
4:    Receive an instance $x_t$
5:    Predict: $\hat{y}_t = \text{sign}\Big(\mathbf{q}_t \cdot \text{sign}(\mathbf{f}_t(x_t))\Big)$
6:    Receive the class label $y_t$
7:    for $i = 1, 2, \ldots, m$ do
8:       Sample $m_i^t$ = Bern_Sample($p_i^t$)
9:       Set $z_i^t = I(y_t f_i^t(x_t) \leq 0)$
10:      Update $w_i^{t+1} = w_i^t \beta^{m_i^t z_i^t}$
11:      Update $f_i^{t+1} = f_i^t + m_i^t z_i^t y_t \kappa_i(x_t, \cdot)$
12:   end for
13:   Update $\mathbf{p}_{t+1} = (1-\delta)\mathbf{q}_{t+1} + \delta\mathbf{1}$
14: end for
```

Remark. Compared to the mistake bound in Theorem 2 by Algorithm 1, the mistake bound by Algorithm 4 is amplified by a factor of $1/\delta$ due to the stochastic procedure. On the other hand, compared to the mistake bound of single update in Theorem 6, the mistake bound by Algorithm 4 is improved by a factor of m, mainly due to simultaneously updating multiple kernel classifiers in each round. The expected number of updates for multiple updates approach is bounded by $\mathrm{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{m} m_i^t z_i^t\right] = \mathrm{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{m} p_i^t z_i^t\right] \leq (1-\delta)\mathrm{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{m} q_i^t z_i^t\right] + \delta m T$, where the first term is discounted by a factor of m and the second term is amplified by a factor of m compared to that of single update approach.

5 Algorithms for Online Multiple Kernel Learning by Outputs (OMKL-O)

5.1 A Deterministic Approach

In the following analysis, we assume the functional norm of any classifier $f_i(\cdot)$ is bounded by R, i.e., $\|f_i\|_{\mathcal{H}_{\kappa_i}} \leq R$. We define domain Ω_{κ_i} as $\Omega_{\kappa_i} = \{f \in \mathcal{H}_{\kappa_i} : \|f\|_{\mathcal{H}_{\kappa_i}} \leq R\}$. Algorithm 5 shows the deterministic algorithm for OMKL-O. Compared to Algorithm 1, there are three key features of Algorithm 5. First, in step 11, the updated kernel classifier $f_i(\cdot)$ is projected into domain Ω_{κ_i} to ensure its norm is no more than R. This projection step is important for the proof of the mistake bound that will be shown later. Second, each individual kernel classifier is updated only when the prediction of the combined classifier is incorrect i.e., $y_t\hat{y}_t \leq 0$. This is in contrast to the Algorithm 1, where each kernel classifier $f_i^t(\cdot)$ is updated when it misclassifies the training example x_t. This feature will make the proposed algorithm significantly more efficient than Algorithm 1. Finally, in step 9 of Algorithm 5, we update weights w_i^{t+1} based on the output $f_i^t(x_t)$. This is in contrast to Algorithm 1 where weights are updated only based on if the individual classifiers classify the example correctly.

Theorem 8. *After receiving a sequence of T training examples $\mathcal{D}_T$, the number of mistakes made by Algorithm 5 is bounded as follows if $R^2 \ln(1/\beta) < 1$*

$$M \leq \frac{1}{1 - R^2 \ln(1/\beta)} \min_{\mathbf{u} \in \Delta, \{f_i \in \Omega_{\kappa_i}\}_{i=1}^m} g\left(\mathbf{u}, \{f_i\}_{i=1}^m\right) + \frac{2 \ln m}{(1 - R^2 \ln(1/\beta)) \ln(1/\beta)}$$

where $g\left(\mathbf{u}, \{f_i\}_{i=1}^m\right) = \sum_{i=1}^m u_i \|f_i\|_{\mathcal{H}_{\kappa_i}}^2 + 2 \sum_{t=1}^T \ell\left(\mathbf{u} \cdot \mathbf{f}(x_t), y_t\right)$.

Using the result in Theorem 1, we have the following corollary that bounds the number of mistakes of online kernel learning by the objective function used in the batch mode multiple kernel learning.

Corollary 9. *We have the following mistake bound for running Algorithm 5 if the inequality $R^2 \ln(1/\beta) < 1$ holds*

$$M \leq \frac{1}{1 - R^2 \ln(1/\beta)} \min_{\mathbf{u} \in \Delta, f \in \Omega_{\kappa(\mathbf{u})}} g\left(\kappa(\mathbf{u}), \ell\right) + \frac{2 \ln m}{(1 - R^2 \ln(1/\beta)) \ln(1/\beta)}$$

where $g\left(\kappa(\mathbf{u}), \ell\right) = \|f\|_{\mathcal{H}_{\kappa(\mathbf{u})}}^2 + 2 \sum_{t=1}^T \ell\left(f(x_t), y_t\right)$.

5.2 A Stochastic Approach

Finally, we present a stochastic strategy in Algorithm 6 for OMKL-O. In each round, we randomly sample one classifier to update by following the probability distribution $\mathbf{p}_t$. Similar to Algorithm 3, the probability distribution $\mathbf{p}_t$ is a mixture of the normalized weights for classifiers and a smoothing term δ/m. Different from Algorithm 5, the updating rule for Algorithm 6 has two additional factors, i.e. $\widetilde{m}_i^t$ which is non-zero for the chosen classifier and has expectation equal to 1, and the step size η which is essentially introduced to ensure a good mistake bound as shown in the following theorem.

Theorem 10. *After receiving a sequence of T training examples $\mathcal{D}_T$, the expected number of mistakes made by Algorithm 6 is bounded as follows*

$$\mathrm{E}[M] \leq \min_{\mathbf{u} \in \Delta, \{f_i \in \Omega_{\kappa_i}\}_{i=1}^m} \sum_{t=1}^T \ell\left(\mathbf{u} \cdot \mathbf{f}(x_t), y_t\right) + 2\sqrt{a(R, \beta, \delta) b(R, \beta, \delta) T}$$

where $a(R, \beta, \delta) = \dfrac{R^2}{2} + \dfrac{\ln m}{\ln(1/\beta)}$, $b(R, \beta, \delta) = \dfrac{\ln(1/\beta) R^2 m^2}{2\delta^2} + \dfrac{m}{2\delta}$, and η is set to

$$\eta = \sqrt{\frac{a(R, \beta, \delta)}{b(R, \beta, \delta)}}$$

Algorithm 5. DA for OMKL-O

```
1: INPUT:
     – Kernels: $\mathcal{K}_m$
     – Discount weight: $\beta \in (0,1)$
     – Maximum functional norm: $R$
2: Initilization: $\mathbf{f}_1 = \mathbf{0}, \mathbf{w}_1 = \mathbf{1}$
3: for $t = 1, 2, \ldots$ do
4:   Receive an instance $x_t$
5:   Predict: $\hat{y}_t = \mathrm{sign}\Big(\mathbf{q}_t \cdot \mathbf{f}_t(x_t)\Big)$
6:   Receive the class label $y_t$
7:   if $\hat{y}_t y_t \le 0$ then
8:     for $i = 1, 2, \ldots, m$ do
9:       Update $w_i^{t+1} = w_i^t \beta^{-y_t f_i^t(x_t)}$
10:      Update $\tilde{f}_i^{t+1} = f_i^t + y_t \kappa_i(x_t, \cdot)$
11:      Project $\tilde{f}_i^{t+1}$ into $\Omega_{\kappa_i}$ by
         $$f_i^{t+1} = \tilde{f}_i^{t+1} / \max(1, \|\tilde{f}_i^{t+1}\|_{\mathcal{H}_{\kappa_i}} / R)$$
12:    end for
13:  end if
14: end for
```

Algorithm 6. SUA for OMKL-O

```
1: INPUT:
     – $\mathcal{K}_m, \beta, R$ as in Algoirthm 5
     – Smoothing parameter $\delta \in (0,1)$, and
       Step size: $\eta > 0$
2: Initilization: $\mathbf{f}_1 = \mathbf{0}, \mathbf{w}_1 = \mathbf{1}, \mathbf{p}_1 = \mathbf{1}/m$
3: for $t = 1, 2, \ldots$ do
4:   Receive an instance $x_t$
5:   Predict: $\hat{y}_t = \mathrm{sign}\Big(\mathbf{q}_t \cdot \mathbf{f}_t(x_t)\Big)$
6:   Receive the class label $y_t$
7:   if $\hat{y}_t y_t \le 0$ then
8:     $i_t$=Multi_Sample($\mathbf{p}_t$)
9:     for $i = 1, 2, \ldots, m$ do
10:      Set $m_i^t = I(i = i_t)$, $\widetilde{m}_i^t = m_i^t / p_i^t$
11:      Update $w_i^{t+1} = w_i^t \beta^{-\eta \widetilde{m}_i^t y_t f_i^t(x_t)}$
12:      Update $\tilde{f}_i^{t+1} = f_i^t + \eta \widetilde{m}_i^t y_t \kappa_i(x_t, \cdot)$
13:      Project $\tilde{f}_i^{t+1}(x)$ into $\Omega_{\kappa_i}$.
14:    end for
15:    Update $\mathbf{p}_t = (1 - \delta)\mathbf{q}_t + \delta \mathbf{1}/m$
16:  end if
17: end for
```

6 Conclusions

This paper investigates a new research problem, online multiple kernel learning, which aims to attack an online learning task by learning a kernel based prediction function from a pool of predefined kernels. We consider two setups for online kernel learning, online kernel learning by predictions that combines the binary predictions from multiple kernel classifiers and online kernel learning by outputs that combines the real-valued outputs from kernel classifiers. We proposed a framework for OMKL by learning a combination of multiple kernel classifiers from a pool of given kernel functions. We emphasize that OMKL is generally more challenging than typical online learning because both the kernel classifiers and their linear combination are unknown. To solve this challenge, we propose to combine two online learning algorithms, i.e., the Perceptron algorithm that learns a classifier for a given kernel, and the Hedge algorithm that combines classifiers by linear weighting. Based on this idea, we present two types of algorithms for OMKL, i.e., deterministic approaches and stochastic approaches. Theoretical bounds were derived for the proposed OMKL algorithms.

Acknowledgement

The work was supported in part by National Science Foundation (IIS-0643494), Office of Naval Research (N00014-09-1-0663), and Army Research Office (W911NF-09-1-0421). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF, ONR and ARO.

References

[1] Agmon, S.: The relaxation method for linear inequalities. CJM 6(3), 382–392 (1954)
[2] Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.E.: The nonstochastic multiarmed bandit problem. SICOMP 32(1) (2003)
[3] Bach, F.R., Lanckriet, G.R.G., Jordan, M.I.: Multiple kernel learning, conic duality, and the smo algorithm. In: ICML (2004)
[4] Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, Cambridge (2006)
[5] Chapelle, O., Weston, J., Schölkopf, B.: Cluster kernels for semi-supervised learning. In: NIPS, pp. 585–592 (2002)
[6] Chen, Y., Gupta, M.R., Recht, B.: Learning kernels from indefinite similarities. In: ICML, pp. 145–152 (2009)
[7] Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., Singer, Y.: Online passive-aggressive algorithms. JMLR 7 (2006)
[8] Crammer, K., Singer, Y.: Ultraconservative online algorithms for multiclass problems. JMLR, 3 (2003)
[9] Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. JCSS 55(1) (1997)
[10] Freund, Y., Schapire, R.E.: Large margin classification using the perceptron algorithm. ML 37(3) (1999)
[11] Gentile, C.: A new approximate maximal margin classification algorithm. JMLR 2 (2001)
[12] Hoi, S.C., Jin, R., Lyu, M.R.: Learning non-parametric kernel matrices from pairwise constraints. In: ICML, pp. 361–368 (2007)
[13] Hoi, S.C.H., Lyu, M.R., Chang, E.Y.: Learning the unified kernel machines for classification. In: KDD, pp. 187–196 (2006)
[14] Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized kernels between labeled graphs. In: ICML, pp. 321–328 (2003)
[15] Kivinen, J., Smola, A., Williamson, R.: Online learning with kernels. IEEE Trans. on Sig. Proc. 52(8) (2004)
[16] Kivinen, J., Smola, A.J., Williamson, R.C.: Online learning with kernels. In: NIPS, pp. 785–792 (2001)
[17] Kondor, R.I., Lafferty, J.D.: Diffusion kernels on graphs and other discrete input spaces. In: ICML, pp. 315–322 (2002)
[18] Kulis, B., Sustik, M., Dhillon, I.: Learning low-rank kernel matrices. In: ICML, pp. 505–512 (2006)
[19] Lanckriet, G.R.G., Cristianini, N., Bartlett, P., Ghaoui, L.E., Jordan, M.I.: Learning the kernel matrix with semidefinite programming. JMLR 5 (2004)
[20] Li, Y., Long, P.M.: The relaxed online maximum margin algorithm. ML 46(1-3) (2002)
[21] Littlestone, N., Warmuth, M.K.: The weighted majority algorithm. In: FOCS (1989)
[22] Novikoff, A.: On convergence proofs on perceptrons. In: Proceedings of the Symposium on the Mathematical Theory of Automata, vol. XII (1962)
[23] Rakotomamonjy, A., Bach, F.R., Canu, S., Grandvalet, Y.: Simplemkl. JMLR 11 (2008)
[24] Rosenblatt, F.: The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review 65 (1958)
[25] Schölkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge (2001)
[26] Shalev-Shwartz, S.: Online learning: Theory, algorithms, and applications. In: Ph.D thesis (2007)

[27] Sonnenburg, S., Rätsch, G., Schäfer, C., Schölkopf, B.: Large scale multiple kernel learning. JMLR 7 (2006)
[28] Vapnik, V.N.: Statistical Learning Theory. Wiley, Chichester (1998)
[29] Vovk, V.: A game of prediction with expert advice. J. Comput. Syst. Sci. 56(2) (1998)
[30] Xu, Z., Jin, R., King, I., Lyu, M.R.: An extended level method for efficient multiple kernel learning. In: NIPS, pp. 1825–1832 (2008)
[31] Xu, Z., Jin, R., Yang, H., King, I., Lyu, M.: Simple and efficient multiple kernel learning by group lasso. In: ICML (2010)
[32] Zhu, X., Kandola, J.S., Ghahramani, Z., Lafferty, J.D.: Nonparametric transforms of graph kernels for semi-supervised learning. In: NIPS, pp. 1641–1648 (2004)

Appendix

Proof of Theorem 2

Proof. The proof essentially combines the proofs of the Perceptron algorithm [24] and the Hedge algorithm [9]. First, following the analysis in [9], we can easily have

$$\sum_{t=1}^{T} \mathbf{q}_t \cdot \mathbf{z}_t \leq \frac{\ln(1/\beta)}{1-\beta} \min_{1\leq i\leq m} \sum_{t=1}^{T} z_i^t + \frac{\ln m}{1-\beta}$$

Second, due to the convexity of ℓ and the updating rule for f_i, when $z_i^t = 1$ we have

$$\begin{aligned} &\ell(f_i^t(x_t), y_t) - \ell(f(x_t), y_t) \leq -y_t \langle f_i^t - f, z_i^t \kappa_i(x_t, \cdot)\rangle_{\mathcal{H}_{\kappa_i}} \\ &= -\langle f_i^t - f, f_i^{t+1} - f_i^t \rangle_{\mathcal{H}_{\kappa_i}} \leq \frac{1}{2}\left(\|f_i^t - f\|_{\mathcal{H}_{\kappa_i}}^2 - \|f_i^{t+1} - f\|_{\mathcal{H}_{\kappa_i}}^2 + z_i^t \right) \end{aligned}$$

Since $\ell(f_i^t(x_t), y_t) \geq z_i^t$, then $z_i^t \leq \|f_i^t - f\|_{\mathcal{H}_{\kappa_i}}^2 - \|f_i^{t+1} - f\|_{\mathcal{H}_{\kappa_i}}^2 + 2\ell(f(x_t), y_t)$. Taking summation on both sides, we have

$$\sum_{t=1}^{T} z_i^t \leq \min_{f\in\mathcal{H}_{\kappa_i}} \sum_{t=1}^{T} \left(\|f_i^t - f\|_{\mathcal{H}_{\kappa_i}}^2 - \|f_i^{t+1} - f\|_{\mathcal{H}_{k_i}}^2 \right) + 2\sum_{t=1}^{T} \ell(f(x_t), y_t) \leq g(\kappa_i, \ell)$$

Using the above inequality and noting that $M = \sum_{t=1}^{T} I\left(\mathbf{q}_t \cdot \mathbf{z}_t \geq 0.5\right) \leq 2\sum_{t=1}^{T} \mathbf{q}_t \cdot \mathbf{z}_t$, we have the result in the theorem.

Proof of Theorem 3

Proof. The proof can be constructed similarly to the proof for Theorem 2 by noting the following three differences. First, the updating rule for the weights can be written as $w_i^{t+1} = w_i^t (\beta^{1/2+\gamma})^{\nu_i^t/(1/2+\gamma)}$, where $\nu_i^t \leq 1/2 + \gamma$. Second, $\sum_i q_i^t \nu_i^t \geq \sum_i q_i^t z_i^t/2$. Third, we have $\ell(f_i^t(x_t), y_t) \geq \nu_i(t) + z_i(t)/2$, and therefore $\sum_{t=1}^{T} \nu_i^t \leq g(\kappa_i, \ell)/2$.

Proof of Theorem 5

Proof. We denote by M_k the number of mistakes made during the segment $[T_k + 1, T_{k+1}]$. It can be shown that the following equality

$$M_k \leq 2 \left(\min_{1 \leq i \leq m} \sum_{t=T_k+1}^{T_{k+1}} z_i^t + \ln m + 2\sqrt{\ln m} 2^{k/2} \right),$$

holds for any $k = 0, \ldots, s-1$. Taking summation over all M_k

$$M = \sum_{k=0}^{s-1} M_k \leq 2 \left(\sum_{k=0}^{s-1} \left[\min_{1 \leq i \leq m} \sum_{t=T_k+1}^{T_{k+1}} z_i^t + 22^{k/2}\sqrt{\ln m} \right] \right) + 2s \ln m$$

we can obtain the bound in the theorem by noting that $\sum_{k=0}^{s-1} \min_{1 \leq i \leq m} \sum_{t=T_k+1}^{T_{k_1}} z_i^t \leq \min_{1 \leq i \leq m} \sum_{t=1}^{T} z_i^t$, and $2^{s-1} - 1 \leq \min_{1 \leq i \leq m} \sum_{t=1}^{T} z_i^t \leq \min_{1 \leq i \leq m} g(\kappa_i, \ell)$.

Proof of Theorem 6

Proof. Similar to the proof for Theorem 2, we can prove

$$\sum_{t=1}^{T} \sum_{i=1}^{m} q_i^t m_i^t z_i^t \leq \frac{\ln(1/\beta)}{1-\beta} \sum_{t=1}^{T} m_i^t z_i^t + \frac{\ln m}{1-\beta}, \text{ and } \sum_{t=1}^{T} m_i^t z_i^t \leq g(\kappa_i, \ell)$$

Taking expectation on both sides, and noting that $\mathrm{E}[m_i^t] = p_i^t \geq \delta/m$, we have

$$\mathrm{E}\left(\sum_{t=1}^{T} \mathbf{q}_t \cdot \mathbf{z}_t \right) \leq \frac{m \ln(1/\beta)}{\delta(1-\beta)} \min_{1 \leq i \leq m} g(\kappa_i, \ell) + \frac{m \ln m}{\delta(1-\beta)}$$

Since $M \leq 2 \sum_{t=1}^{T} \mathbf{q}_t \cdot \mathbf{z}_t$, we have the result stated in the theorem.

Proof of Theorem 7

Proof. The proof can be duplicated similarly to the proof for Theorem 6, except for $p_i^t \geq \delta, i = 1, \ldots, m$ in this case.

Proof of Theorem 8

Proof. In the following analysis, we only consider the subset of iterations where the algorithm makes a mistake. By slightly abusing the notation, we denote by $1, 2, \ldots, M$ the trials where the examples are misclassified by Algorithm 5. For any combination weight $\mathbf{u} = (u_1, \cdots, u_m)^\top \in \Delta$ and any kernel classification function $f_i \in \Omega_{\kappa_i}, i = 1, \ldots, m$, we have

$$\begin{aligned}\ell\big(\mathbf{q}_t \cdot \mathbf{f}_t(x_t), y_t\big) - \ell\big(\mathbf{u} \cdot \mathbf{f}(x_t), y_t\big) &\leq g_t y_t \left(\mathbf{q}_t \cdot \mathbf{f}_t(x_t) - \mathbf{u} \cdot \mathbf{f}(x_t)\right) \\ &= g_t y_t \left(\mathbf{q}_t \cdot \mathbf{f}_t(x_t) - \mathbf{u} \cdot \mathbf{f}_t(x_t)\right) + g_t y_t \left(\mathbf{u} \cdot \mathbf{f}_t(x_t) - \mathbf{u} \cdot \mathbf{f}(x_t)\right)\end{aligned}$$

where $g_t = \frac{\partial \max(0,1-z)}{\partial z}\Big|_{z=y_t \mathbf{q}_t \cdot \mathbf{f}_t(x_t)} = -1$ because examples are misclassified in these trials. For the first term, following the proof for Theorem 11.3 in [4], we can have

$$g_t y_t (\mathbf{q}_t - \mathbf{u}) \cdot \mathbf{f}_t(x_t) \leq \frac{\ln(1/\beta)}{2} R^2 + \frac{1}{\ln(1/\beta)} \left\{\mathrm{KL}(\mathbf{u} \| \mathbf{q}_t) - \mathrm{KL}(\mathbf{u} \| \mathbf{q}_{t+1})\right\}$$

For the second term, following the analysis in the proof for Theorem 2, we can have

$$\begin{aligned}g_t y_t \left(\mathbf{u} \cdot \mathbf{f}_t(x_t) - \mathbf{u} \cdot \mathbf{f}(x_t)\right) &= \sum_{i=1}^m u_i \langle f_t^i - f^i, g_t y_t \kappa_i(x_t, \cdot)\rangle_{\mathcal{H}_{\kappa_i}} \\ &\leq \frac{1}{2} + \sum_{i=1}^m \frac{u_i}{2} \left(\|f_i^t - f_i\|^2_{\mathcal{H}_{\kappa_i}} - \|f_i^{t+1} - f_i\|^2_{\mathcal{H}_{\kappa_i}}\right)\end{aligned}$$

Combining the above results together and noting $\ell\left(\mathbf{q}_t \cdot \mathbf{f}_t(x_t), y_t\right) \geq 1$, we can have the result in the theorem.

Proof of Theorem 10

Proof. Similar to the proof for the Theorem 8, we have the following bound for the two terms

$$\begin{aligned}\mathrm{E}\left[g_t y_t \left(\mathbf{q}_t \cdot \mathbf{f}_t(x_t) - \mathbf{u} \cdot \mathbf{f}_t(x_t)\right)\right] &= \frac{1}{\eta} \mathrm{E}\left[g_t y_t (\mathbf{q}_t - \mathbf{u}_t) \cdot \left(\eta \widetilde{m}_t \circ \mathbf{f}_t(x_t)\right)\right] \\ &\leq \frac{1}{\eta \ln(1/\beta)} \mathrm{E}\left[\mathrm{KL}(\mathbf{u} \| \mathbf{q}(t)) - \mathrm{KL}(\mathbf{u} \| \mathbf{q}(t+1))\right] + \frac{\ln(1/\beta) R^2 \eta^2 m^2}{2\delta^2}\end{aligned}$$

$$\begin{aligned}\mathrm{E}\left[g_t y_t \mathbf{u} \cdot (\mathbf{f}_t(x_t) - \mathbf{f}(x_t))\right] &= \mathrm{E} \sum_{i=1}^m \frac{u_i}{\eta} \left\langle f_i^t - f_i, \eta \widetilde{m}_i^t g_t y_t \kappa_i(x_t, \cdot)\right\rangle_{\mathcal{H}_{\kappa_i}} \\ &\leq \frac{m\eta}{2\delta} + \mathrm{E}\left[\sum_{i=1}^m \frac{u_i}{2\eta} (\|f_i^t - f_i\|^2_{\mathcal{H}_{\kappa_i}} - \|f_i^{t+1} - f_i\|^2_{\mathcal{H}_{\kappa_i}})\right]\end{aligned}$$

Combining the above results together, and setting η as in the theorem, we have the result in the theorem.

An Identity for Kernel Ridge Regression

Fedor Zhdanov and Yuri Kalnishkan

Computer Learning Research Centre and Department of Computer Science,
Royal Holloway, University of London, Egham, Surrey, TW20 0EX, United Kingdom
{fedor,yura}@cs.rhul.ac.uk

Abstract. This paper provides a probabilistic derivation of an identity connecting the square loss of ridge regression in on-line mode with the loss of a retrospectively best regressor. Some corollaries of the identity providing upper bounds for the cumulative loss of on-line ridge regression are also discussed.

1 Introduction

Ridge regression is a powerful technique of machine learning. It was introduced in [9]; the kernel version of it is derived in [15].

Ridge regression can be used as a batch or on-line algorithm. This paper proves an identity connecting the square losses of ridge regression used on the same data in batch and on-line fashions. The identity and the approach to the proof are not entirely new. The identity implicitly appears in [2] for the linear case (it can be obtained by summing (4.21) from [2] in an exact rather than estimated form). The proof method based essentially on Bayesian estimation features in [10], which focuses on probabilistic statements and stops one step short of formulating the identity. In this paper we put it all together, explicitly formulate the identity in terms of ridge regression, and give a simple proof for the kernel case. The identity is obtained by calculating the likelihood in a Gaussian processes model by different ways. Another proof of this fact is given in unpublished technical report [18].

We use the identity to derive several inequalities providing upper bounds for the cumulative loss of ridge regression applied in the on-line fashion. Corollaries 2 and 3 deal with 'clipped' ridge regression. The later reproduces Theorem 4.6 from [2] (this result is often confused with Theorem 4 in [17], which, in fact, provides a similar bound for an essentially different algorithm). Corollary 4 (reproduced from [18]) shows that in the linear case the loss of (unclipped) on-line ridge regression is asymptotically close to the loss of a retrospectively best regressor.

In the literature there is a range of specially designed regression-type algorithms with better worst-case bounds or bounds covering wider cases. Aggregating algorithm regression (also known as Vovk-Azoury-Warmuth predictor) is described in [17], [2], and Section 11.8 of [6]. Theorem 1 in [17] provides an upper bound for aggregating algorithm regression, which is better than the bound given by Corollary 3 for clipped ridge regression. The bound from [17] has also been shown to be optimal. The exact relation between the performances of ridge

© Springer-Verlag Berlin Heidelberg 2010

regression and aggregating algorithm regression is not known. Theorem 3 in [17] describes a case where aggregating algorithm regression performs better, but in the case of unbounded signals. An important class of regression-type algorithms achieving different bounds is based on the gradient descent idea; see [5], [11], and Section 11 in [6]. Algorithms in [8] and [4] provide regression-type algorithms dealing with changing dependencies.

The paper is organised as follows. Section 2 introduces kernels and kernel ridge regression in batch and on-line settings. We take the simplest approach and use an explicit formula to introduce ridge regression. Section 3 contains the statement of the identity and Section 4 discusses corollaries of the identity. The rest of the paper is devoted to the proof of the identity. Section 5 introduces a probabilistic interpretation of ridge regression in the context of Gaussian fields and Section 6 contains the proof. Section 7 contains an outline of an alternative proof based on the aggregating algorithm.

2 Kernel Ridge Regression in On-Line and Batch Settings

2.1 Kernels

A *kernel* on a domain X, which is an arbitrary set with no structure assumed, is a symmetric positive semi-definite function of two arguments, i.e., $\mathcal{K} : X \times X \to \mathbb{R}$ such that

1. for all $x_1, x_2 \in X$ we have $\mathcal{K}(x_1, x_2) = \mathcal{K}(x_2, x_1)$ and
2. for any positive integer T, any $x_1, x_2, \ldots, x_T \in X$ and any real numbers $\alpha_1, \alpha_2, \ldots, \alpha_T \in \mathbb{R}$ we have $\sum_{i,j=1}^{T} \mathcal{K}(x_i, x_j)\alpha_i\alpha_j \geq 0$.

An equivalent definition can be given as follows. There is a Hilbert space $\mathcal{F}$ of functions on X such that

1. for every $x \in X$ the function $\mathcal{K}(x, \cdot)$, i.e., $\mathcal{K}$ considered as a function of the second argument with the first argument fixed, belongs to $\mathcal{F}$ and
2. for every $x \in X$ and every $f \in \mathcal{F}$ the value of f at x equals the scalar product of f by $\mathcal{K}(x, \cdot)$, i.e., $f(x) = \langle f, \mathcal{K}(x, \cdot)\rangle_{\mathcal{F}}$; this property is often called the *reproducing property.*

The second definition is sometimes said to specify a *reproducing kernel.* The space $\mathcal{F}$ is called the *reproducing kernel Hilbert space (RKHS)* for the kernel $\mathcal{K}$ (it can be shown that the RKHS for a kernel $\mathcal{K}$ is unique). The equivalence of the two definitions is proven in [1].

2.2 Regression in Batch and On-Line Settings

Suppose that we are given a sample of pairs

$$S = ((x_1, y_1), (x_2, y_2), \ldots, (x_T, y_T)) \ ,$$

where all $x_t \in X$ are called *signals* and $y_t \in \mathbb{R}$ are called *outcomes* for the corresponding signals. A pair (x_t, y_t) is called *an example.*

The task of regression is to fit a function (usually from a particular class) to the data. The method of *kernel ridge regression* with a kernel $\mathcal{K}$ and a real regularisation parameter $a > 0$ suggests the function $f_{\mathrm{RR}}(x) = Y'(K + aI)^{-1}k(x)$, where $Y = (y_1, y_2, \ldots, y_T)'$ is the column vector of outcomes,

$$K = \begin{pmatrix} \mathcal{K}(x_1,x_1) & \mathcal{K}(x_1,x_2) & \ldots & \mathcal{K}(x_1,x_T) \\ \mathcal{K}(x_2,x_1) & \mathcal{K}(x_2,x_2) & \ldots & \mathcal{K}(x_2,x_T) \\ \vdots & \vdots & \ddots & \ldots \\ \mathcal{K}(x_T,x_1) & \mathcal{K}(x_T,x_2) & \ldots & \mathcal{K}(x_T,x_T) \end{pmatrix}$$

is the *kernel matrix* and

$$k(x) = \begin{pmatrix} \mathcal{K}(x_1,x) \\ \mathcal{K}(x_2,x) \\ \vdots \\ \mathcal{K}(x_T,x) \end{pmatrix} .$$

Note that the matrix K is positive-semidefinite by the definition of a kernel, therefore the matrix $K + aI$ is positive-definite and thus non-singular.

It is easy to see that $f_{\mathrm{RR}}(x)$ is a linear combination of functions $\mathcal{K}(x_t, x)$ (note that x does not appear outside of $k(x)$ in the ridge regression formula) and therefore it belongs to the RKHS $\mathcal{F}$ specified by the kernel $\mathcal{K}$. It can be shown that on this f the minimum of the expression $\sum_{t=1}^{T}(f(x) - y_t)^2 + a\|f\|_{\mathcal{F}}^2$ (where $\|\cdot\|_{\mathcal{F}}$ is the norm in $\mathcal{F}$) over all f from the RKHS $\mathcal{F}$ is achieved.

Suppose now that the sample is given to us example by example. For each example we are shown the signal and then asked to produce a prediction for the outcome. One can say that the learner operates according to the following protocol:

Protocol 1

```
for $t = 1, 2, \ldots$
  read signal $x_t$
  output prediction $\gamma_t$
  read true outcome $y_t$
endfor
```

This learning scenario is called *on-line* or *sequential.* The scenario when the whole sample is given to us at once as before is called *batch* to distinguish it from on-line.

One can apply ridge regression in the on-line scenario in the following natural way. On step t we form the sample S_t from the $t - 1$ known examples $(x_1, y_1), (x_2, y_2), \ldots, (x_{t-1}, y_{t-1})$ and output the prediction suggested by the ridge regression function for this sample.

For the on-line scenario we will use the same notations as in the batch mode but with the index t denoting the time. Thus K_t is the kernel matrix on step t

(note that its size is $(t-1)\times(t-1)$), Y_t is the vector of outcomes $y_1, y_2, \dots, y_{t-1}$, and k_t is $k(x)$ for step t. We will be referring to the prediction output by on-line ridge regression on step t as γ_t^{RR}.

3 The Identity

Theorem 1. *Take a kernel $\mathcal{K}$ on a domain X and a parameter $a > 0$. Let $\mathcal{F}$ be the RKHS for the kernel $\mathcal{K}$. For a sample $(x_1, y_1), (x_2, y_2), \dots, (x_T, y_T)$ let $\gamma_1^{\mathrm{RR}}, \gamma_2^{\mathrm{RR}}, \dots, \gamma_T^{\mathrm{RR}}$ be the predictions output by ridge regression with the kernel $\mathcal{K}$ and the parameter a in the on-line mode. Then*

$$\sum_{t=1}^{T} \frac{(\gamma_t^{\mathrm{RR}} - y_t)^2}{1 + d_t/a} = \min_{f\in\mathcal{F}} \left(\sum_{t=1}^{T} (f(x_t) - y_t)^2 + a\|f\|_{\mathcal{F}}^2 \right) = aY'(K_{T+1} + aI)^{-1}Y \ ,$$

where $d_t = \mathcal{K}(x_t, x_t) - k_t'(x_t)(K_t + aI)^{-1}k_t(x_t) > 0$ and all other notation is as above.

The left-hand side term in this equality is *close* to the cumulative squared loss of ridge regression in the on-line mode. The difference is in the denominators $1 + d_t/a$. The values d_t have the meaning of variances of ridge regression predictions according to the probabilistic view discussed below.

Note that the minimum in the middle term is attained on f specified by batch ridge regression knowing the whole sample. It is thus *nearly* the squared loss of the *retrospectively* best fit $f \in \mathcal{F}$.

The right-hand side term is a simple closed-form expression.

4 Corollaries

In this section we use the identity to obtain upper bounds on cumulative losses of on-line algorithms.

It is easy to obtain a basic multiplicative bound on the loss of on-line ridge regression. The matrix $(K_t + aI)^{-1}$ is positive-definite as the inverse of a positive-definite, therefore $k_t'(x_t)(K_t + aI)^{-1}k_t(x_t) \geq 0$ and $d_t \leq \mathcal{K}(x_t, x_t)$. Assuming that there is $c_{\mathcal{F}} > 0$ such that $\mathcal{K}(x, x) \leq c_{\mathcal{F}}^2$ on X (i.e., the evaluation functional on $\mathcal{F}$ is uniformly bounded by $c_{\mathcal{F}}$), we get

$$\sum_{t=1}^{T} (\gamma_t^{\mathrm{RR}} - y_t)^2 \leq \left(1 + \frac{c_{\mathcal{F}}^2}{a}\right) \min_{f\in\mathcal{F}} \left(\sum_{t=1}^{T} (f(x_t) - y_t)^2 + a\|f\|_{\mathcal{F}}^2 \right) = a\left(1 + \frac{c_{\mathcal{F}}^2}{a}\right) Y'(K_{T+1} + aI)^{-1}Y \ . \quad (1)$$

More interesting bounds can be obtained on the following assumption. Suppose that we know in advance that outcomes y come from an interval $[-Y, Y]$, and Y is known to us. It does not make sense then to make predictions outside of the interval. One may consider *clipped ridge regression*, which operates as follows. For a given signal the ridge regression prediction γ^{RR} is calculated; if it falls inside the interval, it is kept; if it is outside of the interval, it is replaced by the closest point from the interval. Denote the prediction of clipped ridge regression by $\gamma^{\mathrm{RR},Y}$. If $y \in [-Y, Y]$ indeed holds, then $(\gamma^{\mathrm{RR},Y} - y)^2 \le (\gamma^{\mathrm{RR}} - y)^2$ and $(\gamma^{\mathrm{RR},Y} - y)^2 \le 4Y^2$.

Corollary 2. *Take a kernel $\mathcal{K}$ on a domain X and a parameter $a > 0$. Let $\mathcal{F}$ be the RKHS for the kernel $\mathcal{K}$. For a sample $(x_1, y_1), (x_2, y_2), \ldots, (x_T, y_T)$ such that $y_t \in [-Y, Y]$ for all $t = 1, 2, \ldots, T$, let $\gamma_1^{\mathrm{RR},Y}, \gamma_2^{\mathrm{RR},Y}, \ldots, \gamma_T^{\mathrm{RR},Y}$ be the predictions output by clipped ridge regression with the kernel $\mathcal{K}$ and the parameter a in the on-line mode. Then*

$$\sum_{t=1}^{T} (\gamma_t^{\mathrm{RR},Y} - y_t)^2 \le$$

$$\min_{f \in \mathcal{F}} \left(\sum_{t=1}^{T} (f(x_t) - y_t)^2 + a\|f\|_{\mathcal{F}}^2 \right) + 4Y^2 \ln\det\left(I + \frac{1}{a} K_{T+1}\right) ,$$

where K_{T+1} is as above.

Proof. We have

$$\frac{1}{1 + d_t/a} = 1 - \frac{d_t/a}{1 + d_t/a}$$

and

$$\frac{d_t/a}{1 + d_t/a} \le \ln(1 + d_t/a) ;$$

indeed, for $b \ge 0$ the inequality $b/(1+b) \le \ln(1+b)$ holds and can be checked by differentiation. Therefore

$$\begin{aligned} \sum_{t=1}^{T} (\gamma_t^{\mathrm{RR},Y} - y_t)^2 &= \sum_{t=1}^{T} (\gamma_t^{\mathrm{RR},Y} - y_t)^2 \frac{1}{1 + d_t/a} + \sum_{t=1}^{T} (\gamma_t^{\mathrm{RR},Y} - y_t)^2 \frac{d_t/a}{1 + d_t/a} \\ &\le \sum_{t=1}^{T} (\gamma_t^{\mathrm{RR}} - y_t)^2 \frac{1}{1 + d_t/a} + 4Y^2 \sum_{t=1}^{T} \ln(1 + d_t/a) . \end{aligned}$$

Lemma 7 proved below yields

$$\prod_{t=1}^{T} (1 + d_t/a) = \frac{1}{a^T} \det(K_{T+1} + aI) = \det\left(I + \frac{1}{a} K_{T+1}\right) .$$

□

There is no sublinear upper bound on the regret term $4Y^2 \ln\det\left(I + \frac{1}{a} K_{T+1}\right)$ in the general case; indeed, consider the kernel

$$\delta(x_1, x_2) = \begin{cases} 1 & \text{if } x_1 = x_2; \\ 0 & \text{otherwise.} \end{cases}$$

However we can get good bounds in special cases.

It is shown in [16] that for the radial-basis kernel $\mathcal{K}(x_1, x_2) = e^{-b\|x_1 - x_2\|^2}$, where $x_1, x_2 \in \mathbb{R}^d$, we can get an upper bound on average. Suppose that all xs are independently identically distributed according to the Gaussian distribution with the mean of 0 and variance of cI. Then for the expectation we have $E \ln \det \left(I + \frac{1}{a} K_{T+1}\right) = O((\ln T)^{d+1})$ (see Section IV.B in [16]). This yields a bound on the expected loss of clipped ridge regression.

Consider the linear kernel $\mathcal{K}(x_1, x_2) = x_1' x_2$ defined on column vectors from $\mathbb{R}^n$. We have $\mathcal{K}(x, x) = \|x\|^2$, where $\|\cdot\|$ is the quadratic norm in $\mathbb{R}^n$. The reproducing kernel Hilbert space is the set of all linear functions on $\mathbb{R}^n$. We have $K_t = X_t' X_t$, where X_{T+1} is the *design matrix* made up of column vectors $x_1, x_2, \ldots, x_T$. The Sylvester determinant identity (see, e.g., [7]) implies that

$$\det\left(I + \frac{1}{a} X_{T+1}' X_{T+1}\right) = \det\left(I + \frac{1}{a} X_{T+1} X_{T+1}'\right) = \det\left(I + \frac{1}{a}\sum_{t=1}^{T} x_t x_t'\right) .$$

Estimating the determinant by the product of its diagonal elements (see, e.g., Theorem 7 in Chapter 2 of [3]) and assuming that all coordinates of x_t are bounded by B, we get

$$\det\left(I + \frac{1}{a}\sum_{t=1}^{T} x_t x_t'\right) \le \left(1 + \frac{TB^2}{a}\right)^n .$$

We get the following corollary.

Corollary 3. *For a sample $(x_1, y_1), (x_2, y_2), \ldots, (x_T, y_T)$, where $x_t \in [-B, B]^n$ and $y_t \in [-Y, Y]$ for all $t = 1, 2, \ldots, T$, let $\gamma_1^{\mathrm{RR},Y}, \gamma_2^{\mathrm{RR},Y}, \ldots, \gamma_T^{\mathrm{RR},Y}$ be the predictions output by clipped linear ridge regression with a parameter $a > 0$ in the on-line mode. Then*

$$\sum_{t=1}^{T} (\gamma_t^{\mathrm{RR},Y} - y_t)^2 \le \min_{\theta \in \mathbb{R}^n} \left(\sum_{t=1}^{T} (\theta' x_t - y_t)^2 + a\|\theta\|^2\right) + 4Y^2 n \ln\left(1 + \frac{TB^2}{a}\right) .$$

It is an interesting problem if the bound is optimal. As far as we know, there is a gap in existing bounds. Theorem 2 in [17] shows that $Y^2 n \ln T$ is a lower bound for *any* learner and in the constructed example $\|x_t\|_\infty = 1$. Theorem 3 in [17] provides a stronger lower bound, but at a cost of allowing unbounded xs.

For the linear kernel the expression d_t/a in the denominator of the identity can be rewritten as follows:

$$\begin{aligned}\frac{d_t}{a} &= \frac{1}{a}\left[\mathcal{K}(x_t, x_t) - k_t'(x_t)(K_t + aI)^{-1} k_t(x_t)\right] \\ &= \frac{1}{a}\left[x_t' x_t - (x_t' X_t)(X_t' X_t + aI)^{-1}(X_t' x_t)\right] .\end{aligned}$$

We can apply the matrix identity $A(BA + I)^{-1} = (AB + I)^{-1} A$ (it holds if both the inversions can be performed and can be proven by multiplying both the sides by $BA + I$ and $AB + I$ and opening up the brackets) and further obtain

$$\begin{aligned}\frac{d_t}{a} &= \frac{1}{a}\left[x_t'x_t - x_t'(X_tX_t'+aI)^{-1}X_tX_t'x_t\right]\\ &= \frac{1}{a}\left[x_t'(I-(X_tX_t'+aI)^{-1}X_tX_t')x_t\right]\\ &= x_t'(X_tX_t'+aI)^{-1}x_t\end{aligned}$$

We will denote $X_tX_t'+aI$ by A_t. One can easily see that

$$A_t = aI + \sum_{i=1}^{t-1} x_ix_i' = a\sum_{i=1}^{n} e_ie_i' + \sum_{i=1}^{t-1} x_ix_i' \ ,$$

where e_i are unit vectors from the standard basis. If one assumes that the norms $\|x_t\|$, $t=1,2,\ldots$ are bounded, one can apply Lemma A.1 from [12] and infer that $x_t'A_t^{-1}x_t \to 0$ as $t\to\infty$. Note that this convergence does not hold in the general kernel case. Indeed, if $\mathcal{K}=\delta$ defined above and all x_t are different, we get $d_t=1$.

The leftmost side of the identity is thus asymptotically close to the cumulative loss of on-line ridge regression and the regularised loss of the retrospectively best regressor in the linear case. We will reproduce a corollary from [18] formalising this intuition.

Corollary 4. *Let $x_t\in\mathbb{R}^n$, $t=1,2,\ldots$ and $\sup_{t=1,2,\ldots}\|x_t\|<\infty$; let γ_t^{RR} be the predictions output by on-line ridge regression with the linear kernel and a parameter $a>0$. Then*

1. *if there is $\theta\in\mathbb{R}^n$ such that $\sum_{t=1}^{\infty}(y_t-\theta'x_t)^2<+\infty$ then*

$$\sum_{t=1}^{\infty}(y_t-\gamma_t^{\mathrm{RR}})^2<+\infty \ ;$$

2. *if for all $\theta\in\mathbb{R}^n$ we have $\sum_{t=1}^{\infty}(y_t-\theta'x_t)^2=+\infty$, then*

$$\lim_{T\to\infty}\frac{\sum_{t=1}^{T}(y_t-\gamma_t^{\mathrm{RR}})^2}{\min_{\theta\in\mathbb{R}^n}\left(\sum_{t=1}^{T}(y_t-\theta'x_t)^2+a\|\theta\|^2\right)}=1 \ . \qquad (2)$$

Proof. Part 1 follows from bound (1).

Let us prove Part 2. First note that $x_t'A_t^{-1}x_t\ge 0$ implies

$$\sum_{t=1}^{T}(y_t-\gamma_t^{\mathrm{RR}})^2 \ge \sum_{t=1}^{T}\frac{(y_t-\gamma_t^{\mathrm{RR}})^2}{1+x_t'A_t^{-1}x_t} = \min_{\theta\in\mathbb{R}^n}\left(\sum_{t=1}^{T}(y_t-\theta'x_t)^2+a\|\theta\|^2\right)$$

and thus the fraction in (2) is always greater than or equal to 1.

Let us show that $\min_{\theta\in\mathbb{R}^n}\left(\sum_{t=1}^{T}(y_t-\theta'x_t)^2+a\|\theta\|^2\right)\to+\infty$ as $t\to\infty$. Suppose that this does not hold. Then there is a sequence T_k and θ_{T_k} such that the expressions $\sum_{t=1}^{T_k}(y_t-\theta_{T_k}'x_t)^2+a\|\theta_{T_k}\|^2$ are bounded. Hence there is $C < +\infty$ such that $\sum_{t=1}^{T_k}(y_t - \theta'_{T_k}x_t)^2 \le C$ for all $k = 1, 2, \dots$ and the norms of θ_{T_k} are also bounded uniformly in k. Therefore the sequence θ_{T_k} has a converging subsequence. Let θ_0 be the limit of this subsequence. Let us show that $\sum_{i=1}^{T_k}(y_t - \theta'_0 x_t)^2 \le C$. Indeed, let $\sum_{i=1}^{T_k}(y_t - \theta'_0 x_t)^2 > C$. For sufficiently large m the sum $\sum_{i=1}^{T_k}(y_t - \theta'_{T_m}x_t)^2$ is sufficiently close to $\sum_{i=1}^{T_k}(y_t - \theta'_0 x_t)^2$ so that

$$\sum_{i=1}^{T_m}(y_t - \theta'_{T_m}x_t)^2 \ge \sum_{i=1}^{T_k}(y_t - \theta'_{T_m}x_t)^2 > C \ ,$$

which contradicts $\sum_{t=1}^{T_m}(y_t - \theta'_{T_m}x_t)^2 \le C$. In the limit we get $\sum_{i=1}^{\infty}(y_t - \theta'_0 x_t)^2 \le C < +\infty$, which contradicts the condition of Part 2.

Take $\varepsilon > 0$. There is T_0 such that for all $T \ge T_0$ we have $1 + x'_T A_T^{-1} x_T \le 1 + \varepsilon$ and

$$\begin{aligned}
\sum_{t=1}^{T}(y_t - \gamma_t^{\mathrm{RR}})^2 &= \sum_{t=1}^{T_0}(y_t - \gamma_t^{\mathrm{RR}})^2 + \sum_{t=T_0+1}^{T}(y_t - \gamma_t^{\mathrm{RR}})^2 \\
&\le \sum_{t=1}^{T_0}(y_t - \gamma_t^{\mathrm{RR}})^2 + (1+\epsilon)\sum_{t=1}^{T}\frac{(y_t - \gamma_t^{\mathrm{RR}})^2}{1 + x'_T A_T^{-1} x_T} \\
&= \sum_{t=1}^{T_0}(y_t - \gamma_t^{\mathrm{RR}})^2 + (1+\epsilon)\min_{\theta\in\mathbb{R}^n}\left(\sum_{t=1}^{T}(y_t - \theta' x_t)^2 + a\|\theta\|^2\right) .
\end{aligned}$$

Therefore for all sufficiently large T the fraction in (2) does not exceed $1 + \varepsilon$. □

5 Probabilistic Interpretation

We will prove the identity by means of the probabilistic interpretation of ridge regression.

Suppose that we have a Gaussian random field[1] z_x with the means of 0 and the covariances $\mathrm{cov}(z_{x_1}, z_{x_2}) = \mathcal{K}(x_1, x_2)$. Such a field exists. Indeed, for any finite set of $x_1, x_2, \dots, x_T$ our requirements imply the Gaussian distribution with the mean of 0 and the covariance matrix of K. These distributions satisfy the consistency requirements and thus the Kolmogorov extension (or existence) theorem (see, e.g., [13], Appendix 1 for a proof sketch[2]) can be applied to construct a field over X.

Let ε_x be a Gaussian field of mutually independent and independent of z_x random values with the variance σ^2. The existence of such a field can be shown using the same Kolmogorov theorem. Now let $y_x = z_x + \varepsilon_x$. Intuitively, ε_x can

[1] We use the term 'field' rather than 'process' to emphasise the fact that X is not necessarily a subset of $\mathbb{R}$ and its elements do not have to be moments of time; some textbooks still use the word 'process' in this case.

[2] Strictly speaking, we do not need to construct the field for the whole X in order to prove the theorem; is suffices to consider a finite-dimensional Gaussian distribution of $(z_{x_1}, z_{x_2}, \dots, z_{x_T})$.

be thought of as random noise introduced by measurements of the original field z_x.

The learning process can be thought of as estimating the values of the field y_t given the values of the field at sample points. One can show that the conditional distribution of z_x given a sample $S = ((x_1, y_1), (x_2, y_2), \ldots, (x_T, y_T))$ is Gaussian with the mean of $\gamma_x^{\mathrm{RR}} = Y'(K + \sigma^2 I)^{-1}k(x)$ and the variance $d_x = \mathcal{K}(x, x) - k'(x)(K + \sigma^2 I)^{-1}k(x)$. The conditional distribution of y_x is Gaussian with the same mean and the variance $\sigma^2 + \mathcal{K}(x, x) - k'(x)(K + \sigma^2 I)^{-1}k(x)$ (see [14], Section 2.2, p. 17).

If we let $a = \sigma^2$, we see that γ_t^{RR} and $a + d_t$ are, respectively, the mean and the variance of the conditional distributions for y_{x_t} given the sample S_t.

Remark 5. Note that in the statement of the theorem there is no assumption that the signals x_t are pairwise different. Some of them may coincide. In the probabilistic picture all xs must be different though, or the corresponding probabilities make no sense. This obstacle may be overcome in the following way. Let us replace the domain X by $X' = X \times \mathbb{N}$, where $\mathbb{N}$ is the set of positive integers $\{1, 2, \ldots\}$, and replace x_t by $x'_t = (x_t, t) \in X'$. For X' there is a Gaussian field with the covariance function $\mathcal{K}'((x_1, t_1), (x_2, t_2)) = \mathcal{K}(x_1, x_2)$. The argument concerning the probabilistic meaning of ridge regression stays for $\mathcal{K}'$ on X'. We can thus assume that all x_t are different.

6 Proof of the Identity

The proof is based on the Gaussian field interpretation. Let us calculate the density of the joint distribution of the variables $(y_{x_1}, y_{x_2}, \ldots, y_{x_T})$ at the point $(y_1, y_2, \ldots, y_T)$. We will do this in three different ways: by decomposing the density into a chain of conditional densities, marginalisation, and, finally, direct calculation. Each method will give us a different expression corresponding to a term in the identity. Since all the three terms express the same density, they must be equal.

6.1 Conditional Probabilities

We have

$$p_{y_{x_1}, y_{x_2}, \ldots, y_{x_T}}(y_1, y_2, \ldots, y_T) = \\ p_{y_{x_T}}(y_T \mid y_{x_1} = y_1, y_{x_2} = y_2, \ldots, y_{x_{T-1}} = y_{T-1}) \cdot \\ p_{y_{x_1}, y_{x_2}, \ldots, y_{x_{T-1}}}(y_1, y_2, \ldots, y_{T-1}) \enspace .$$

Expanding this further yields

$$p_{y_{x_1}, y_{x_2}, \ldots, y_{x_T}}(y_1, y_2, \ldots, y_T) = \\ p_{y_{x_T}}(y_T \mid y_{x_1} = y_1, y_{x_2} = y_2, \ldots, y_{x_{T-1}} = y_{T-1}) \cdot \\ p_{y_{x_{T-1}}}(y_T \mid y_{x_1} = y_1, y_{x_2} = y_2, \ldots, y_{x_{T-1}} = y_{T-2}) \cdots p_{y_{x_1}}(y_1) \enspace .$$

As we have seen before, the distribution for y_{x_t} given that $y_{x_1} = y_1, y_{x_2} = y_2, \ldots, y_{x_{t-1}} = y_{t-1}$ is Gaussian with the mean of γ_t^{RR} and the variance of $d_t + \sigma^2$. Thus

$$p_{y_{x_T}}(y_t \mid y_{x_1} = y_1, y_{x_2} = y_2, \ldots, y_{x_{t-1}} = y_{t-1}) = \frac{1}{\sqrt{2\pi}} \frac{1}{\sqrt{d_t + \sigma^2}} e^{-\frac{1}{2}\frac{(y_t - \gamma_t^{\mathrm{RR}})^2}{d_t + \sigma^2}}$$

and

$$p_{y_{x_1}, y_{x_2}, \ldots, y_{x_T}}(y_1, y_2, \ldots, y_T) = \frac{1}{(2\pi)^{T/2}\sqrt{(d_1 + \sigma^2)(d_2 + \sigma^2)\ldots(d_T + \sigma^2)}} e^{-\frac{1}{2}\sum_{t=1}^{T}\frac{(\gamma_t^{\mathrm{RR}} - y_t)^2}{d_t + \sigma^2}} \ .$$

6.2 Dealing with Singular Kernel Matrix

The expression for the second case looks particularly simple for non-singular K. Let us show that this is sufficient to prove the identity.

All the terms in the identity are in fact some continuous functions of $T(T+1)/2$ values of $\mathcal{K}$ at the pairs of points x_i, x_j, $i, j = 1, 2, \ldots, T$. Indeed, the values of γ_t^{RR} in the left-hand side expression are ridge regression predictions given by respective analytic formula. Note that the coefficients of the inverse matrix are continuous functions of the original matrix.

The optimal function minimising the second expression is in fact $f_{\mathrm{RR}}(x) = \sum_{t=1}^{T} c_t \mathcal{K}(x_t, x)$, where the coefficients c_t are continuous functions of the values of $\mathcal{K}$. The reproducing property implies that

$$\|f_{\mathrm{RR}}\|^2 = \sum_{i,j=1}^{T} c_i c_j \langle \mathcal{K}(x_i, \cdot), \mathcal{K}(x_j, \cdot) \rangle_{\mathcal{F}} = \sum_{i,j=1}^{T} c_i c_j \mathcal{K}(x_i, x_j) \ .$$

We can thus conclude that all the expressions are continuous in the values of $\mathcal{K}$. Consider the kernel $\mathcal{K}_\alpha(x_1, x_2) = \mathcal{K}(x_1, x_2) + \alpha\delta(x_1, x_2)$, where

$$\delta(x_1, x_2) = \begin{cases} 1 & \text{if } x_1 = x_2; \\ 0 & \text{otherwise} \end{cases}$$

and $\alpha > 0$. Clearly, δ is a kernel and thus $\mathcal{K}_\alpha$ is a kernel. If all x_t are different (recall Remark 5), kernel matrix for $\mathcal{K}_\alpha$ equals $K + \alpha I$ and therefore it is nonsingular.

However the values of $\mathcal{K}_\alpha$ tend to the corresponding values of $\mathcal{K}$ as $\alpha \to 0$.

6.3 Marginalisation

The method of marginalisation consists of introducing extra variables to obtain the joint density in some manageable form and then integrating over the extra variables to get rid of them. The variables we are going to consider are $z_{x_1}, z_{x_2}, \ldots, z_{x_T}$.

Given the values of $z_{x_1}, z_{x_2}, \ldots, z_{x_T}$, the density of $y_{x_1}, y_{x_2}, \ldots, y_{x_T}$ is easy to calculate. Indeed, given zs all ys are independent and have the means of corresponding zs and variances of σ^2, i.e.,

$$p_{y_{x_1}, y_{x_2}, \ldots, y_{x_T}}(y_1, y_2, \ldots, y_T \mid z_{x_1} = z_1, z_{x_2} = z_2, \ldots, z_{x_{T-1}} = z_{T-1}) = \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma} e^{-\frac{1}{2}\frac{(y_1 - z_1)^2}{\sigma^2}} \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma} e^{-\frac{1}{2}\frac{(y_2 - z_2)^2}{\sigma^2}} \cdots \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma} e^{-\frac{1}{2}\frac{(y_T - z_T)^2}{\sigma^2}} = \frac{1}{(2\pi)^{T/2}\sigma^T} e^{-\frac{1}{2\sigma^2}\sum_{t=1}^T (y_t - z_t)^2}$$

The density of $z_{x_1}, z_{x_2}, \ldots, z_{x_T}$ is given by

$$p_{z_{x_1}, z_{x_2}, \ldots, z_{x_T}}(z_1, z_2, \ldots, z_T) = \frac{1}{(2\pi)^{T/2}\sqrt{\det K_{T+1}}} e^{-\frac{1}{2} Z' K_{T+1}^{-1} Z} \ ,$$

where $Z = (z_1, z_2, \ldots, z_T)$, provided K_{T+1} is nonsingular.

Using

$$p_{y_{x_1}, y_{x_2}, \ldots, y_{x_T}, z_{x_1}, z_{x_2}, \ldots, z_{x_T}}(y_1, y_2, \ldots, y_T, z_1, z_2, \ldots, z_T) = p_{y_{x_1}, y_{x_2}, \ldots, y_{x_T}}(y_1, y_2, \ldots, y_T \mid z_{x_1} = z_1, z_{x_2} = z_2, \ldots, z_{x_{T-1}} = z_{T-1}) \cdot p_{z_{x_1}, z_{x_2}, \ldots, z_{x_T}}(z_1, z_2, \ldots, z_T)$$

and

$$p_{y_{x_1}, y_{x_2}, \ldots, y_{x_T}}(y_1, y_2, \ldots, y_T) = \int_{\mathbb{R}^T} p_{y_{x_1}, y_{x_2}, \ldots, y_{x_T}, z_{x_1}, z_{x_2}, \ldots, z_{x_T}}(y_1, y_2, \ldots, y_T, z_1, z_2, \ldots, z_T) dZ$$

we get

$$p_{y_{x_1}, y_{x_2}, \ldots, y_{x_T}}(y_1, y_2, \ldots, y_T) = \frac{1}{(2\pi)^{T/2}\sigma^T} \frac{1}{(2\pi)^{T/2}\sqrt{\det K_{T+1}}} \int_{\mathbb{R}^T} e^{-\frac{1}{2\sigma^2}\sum_{t=1}^T (y_t - z_t)^2 - \frac{1}{2} Z' K_{T+1}^{-1} Z} dZ \ .$$

To evaluate the integral we need the following lemma (see [3], Theorem 3 of Chapter 2) .

Lemma 6. *Let $Q(\theta)$ be a quadratic form of $\theta \in \mathbb{R}^n$ with the positive definite quadratic part, i.e., $Q(\theta) = \theta' A \theta + \theta' b + c$, where the matrix A is symmetric positive definite. Then*

$$\int_{\mathbb{R}^n} e^{-Q(\theta)} d\theta = e^{-Q(\theta_0)} \frac{\pi^{n/2}}{\sqrt{\det A}} \ ,$$

where $\theta_0 = \arg\min_{\mathbb{R}^n} Q$.

The quadratic part of the form in our integral has the matrix $\frac{1}{2}K_{T+1}^{-1}+\frac{1}{2\sigma^2}I$ and therefore

$$p_{y_{x_1},y_{x_2},\ldots,y_{x_T}}(y_1,y_2,\ldots,y_T)=\\ \frac{1}{(2\pi)^T\sigma^T\sqrt{\det K_{T+1}}}\frac{\pi^{T/2}}{\sqrt{\det(\frac{1}{2}K_{T+1}^{-1}+\frac{1}{2\sigma^2}I)}}\times\\ e^{-\min_Z\left(\frac{1}{2\sigma^2}\sum_{t=1}^T(y_t-z_t)^2-\frac{1}{2}Z'K_{T+1}^{-1}Z\right)}$$

We have

$$\sqrt{\det K_{T+1}}\sqrt{\det\left(\frac{1}{2}K_{T+1}^{-1}+\frac{1}{2\sigma^2}I\right)}=\sqrt{\det\left(\frac{1}{2}I+\frac{1}{2\sigma^2}K_{T+1}\right)}\\ =\frac{1}{2^{T/2}\sigma^T}\sqrt{\det(K_{T+1}+\sigma^2I)}\ .$$

Let us deal with the minimum. We will link it to

$$M=\min_{f\in\mathcal{F}}\left(\sum_{t=1}^T(f(x_t)-y_t)^2+\sigma^2\|f\|_{\mathcal{F}}^2\right)\ .$$

The representer theorem implies that the minimum in the definition of M is achieved on f from the linear span of $\mathcal{K}(x_1,\cdot),\mathcal{K}(x_2,\cdot),\ldots,\mathcal{K}(x_T,\cdot)$, i.e., on a function of the form $f(x)=\sum_{t=1}^T c_t\mathcal{K}(x_t,\cdot)$. For the column vector $Z(x)=(f(x_1),f(x_2),\ldots,f(x_T))'$ we have $Z(x)=K_{T+1}C$, where $C=(c_1,c_2,\ldots,c_T)'$. Since K_{T+1} is supposed to be non-singular, there is a one-to-one correspondence between C and $Z(x)$; we have $C=K_{T+1}^{-1}Z(x)$ and $\|f\|_{\mathcal{F}}^2=C'K_{T+1}C=Z'(x)K_{T+1}^{-1}Z(x)$. Thus

$$\min_Z\left(\frac{1}{2\sigma^2}\sum_{t=1}^T(y_t-z_t)^2+\frac{1}{2}Z'K_{T+1}^{-1}Z\right)=\frac{1}{2\sigma^2}M\ .$$

For the density we get the expression

$$p_{y_{x_1},y_{x_2},\ldots,y_{x_T}}(y_1,y_2,\ldots,y_T)=\frac{1}{(2\pi)^{T/2}\sqrt{\det(K_{T+1}+\sigma^2I)}}e^{-\frac{1}{2\sigma^2}M}\ .$$

6.4 Direct Calculation

One can easily calculate the covariances of ys:

$$\begin{aligned}\operatorname{cov}(y_{x_1},y_{x_2})&=E(z_{x_1}+\varepsilon_{x_1})(z_{x_2}+\varepsilon_{x_2})\\&=Ez_{x_1}z_{x_2}+E\varepsilon_{x_1}\varepsilon_{x_2}\\&=\mathcal{K}(x_1,x_2)+\sigma^2\delta(x_1,x_2)\ .\end{aligned}$$

Therefore, one can write down the expression

$$p_{y_{x_1},y_{x_2},\ldots,y_{x_T}}(y_1,y_2,\ldots,y_T) = \frac{1}{(2\pi)^{T/2}\sqrt{\det(K_{T+1}+\sigma^2 I)}} e^{-\frac{1}{2}Y'_{T+1}(K_{T+1}+\sigma^2 I)^{-1}Y_{T+1}} \ .$$

6.5 Equating the Terms

It remains to take the logarithms of the densities calculated in different ways. We need the following matrix lemma.

Lemma 7

$$(d_1+\sigma^2)(d_2+\sigma^2)\ldots(d_T+\sigma^2) = \det(K_{T+1}+\sigma^2 I)$$

Proof. The lemma follows from Frobenius's identity (see, e.g., [7]):

$$\det\begin{pmatrix} A & u \\ v' & d\end{pmatrix} = (d - v'A^{-1}u)\det A \ ,$$

where d is a scalar and the submatrix A is non-singular.

We have

$$\begin{aligned}\det(K_{T+1}+\sigma^2 I) &= (\mathcal{K}(x_T,x_T)+\sigma^2 - k'_T(x_T)(K_T+\sigma^2 I)^{-1}k_T(x_T))\cdot \\ &\quad \det(K_T+\sigma^2 I) \\ &= (d_T+\sigma^2)\det(K_T+\sigma^2 I) \\ &= \ldots \\ &= (d_T+\sigma^2)(d_{T-1}+\sigma^2)\ldots(d_2+\sigma^2)(d_1+\sigma^2) \ . \end{aligned}$$ □

We get

$$\sum_{t=1}^{T}\frac{(\gamma_t^{\mathrm{RR}}-y_t)^2}{d_t+\sigma^2} = \frac{1}{\sigma^2}M = Y'(K_{T+1}+\sigma^2 I)^{-1}Y \ .$$

The theorem follows.

7 Alternative Derivations for the Linear Case

In this section we outline alternative ways of obtaining the identity in the linear case.

A Gaussian field z_x with the covariance function x'_1x_2 on $\mathbb{R}^n$ can be obtained as follows. Let θ be an n-dimensional Gaussian random variable with the mean of 0 and the covariance matrix I; let $z_x = \theta' x$ and $y_x = z_x + \varepsilon_x$, where ε_x is independent Gaussian with the mean of 0 and the variance of σ^2 (recall that we let $\sigma^2 = a$). Estimating y_x given a sample of pairs (x_t, y_t) can be thought of as going from the prior distribution for θ to a posterior distribution. The learning process described in Section 5 can thus be thought of as Bayesian estimation. It can be performed in an on-line fashion (the term 'sequential' is more common in Bayesian statistics): the posterior distribution serves as the prior for the next step. This procedure leads to the Gaussian distribution for y with the mean equal to the on-line ridge regression prediction. The linear case is thus a special case of the kernel case.

There is an entirely different way to look at this procedure; it is based on the aggregating algorithm (described, e.g., in [17]). Consider the following game between a predictor and the reality. On step t the reality produces x_t; the predictor sees it and outputs a prediction, which is a Gaussian distribution on $\mathbb{R}$ with the density function p_t. Then the reality announces y_t and the predictor suffers loss $-\ln p_t(y_t)$. Suppose that there is a set of experts who play the same game and we are able to see their predictions before making ours. The aim of aggregating algorithm is to merge experts' predictions so as to suffer cumulative loss comparable to that of the best expert. The game we have described happens to be perfectly mixable, so the merging can be done relatively easily.

Let us consider a pool of experts $\mathcal{E}_\theta$, $\theta \in \mathbb{R}^n$, such that on step t expert $\mathcal{E}_\theta$ outputs the Gaussian distribution with the mean of $\theta' x_t$ and the variance σ^2. The aggregating algorithm requires a prior distribution on the experts. Let us take the Gaussian distribution with the mean of 0 and the covariance matrix I. The distribution is updated on each step; one can show that the update corresponds to the Bayesian update of the distribution for θ. Finally, it is possible to show that the distribution output by the aggregating algorithm on step t is the Gaussian distribution with the mean $\gamma_t = Y_t X_t A_t^{-1} x_t$, i.e., the ridge regression prediction, and the variance $\sigma^2 x_t A_t^{-1} x_t + \sigma^2$, i.e., the conditional variance of y_t in the estimation procedure.

The equality between the first two terms in the identity from Theorem 1 can be derived from a fundamental property of the aggregating algorithm, namely, Lemma 1 in [17], which links the cumulative loss of the predictor to experts' losses. For more details see [18].

An advantage of this approach is that we do not need to consider random fields, estimation, prior and posterior distributions etc. All probabilities are no more than weights or predictions. This is arguably more intuitive.

Acknowledgements

The authors have been supported through the EPSRC grant EP/F002998 'Practical competitive prediction'. The first author has also been supported by an ASPIDA grant from the Cyprus Research Promotion Foundation.

The authors are grateful to Vladimir Vovk and Alexey Chernov for useful discussions and to anonymous COLT and ALT reviewers for detailed comments.

References

[1] Aronszajn, N.: La théorie des noyaux reproduisants et ses applications. Première partie. Proceedings of the Cambridge Philosophical Society 39, 133–153 (1943)

[2] Azoury, K.S., Warmuth, M.K.: Relative loss bounds for on-line density estimation with the exponential family of distributions. Machine Learning 43, 211–246 (2001)

[3] Beckenbach, E.F., Bellman, R.E.: Inequalities. Springer, Heidelberg (1961)

[4] Busuttil, S., Kalnishkan, Y.: Online regression competitive with changing predictors. In: Proceedings of Algorithmic Learning Theory, 18th International Conference, pp. 181–195 (2007)

[5] Cesa-Bianchi, N., Long, P., Warmuth, M.K.: Worst-case quadratic loss bounds for on-line prediction of linear functions by gradient descent. IEEE Transactions on Neural Networks 7, 604–619 (1996)

[6] Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, Cambridge (2006)

[7] Henderson, H.V., Searle, S.R.: On deriving the inverse of a sum of matrices. SIAM Review 23(1) (1981)

[8] Herbster, M., Warmuth, M.K.: Tracking the best linear predictor. Journal of Machine Learning Research 1, 281–309 (2001)

[9] Hoerl, A.E.: Application of ridge analysis to regression problems. Chemical Engineering Progress 58, 54–59 (1962)

[10] Kakade, S.M., Seeger, M.W., Foster, D.P.: Worst-case bounds for Gaussian process models. In: Proceedings of the 19th Annual Conference on Neural Information Processing Systems (2005)

[11] Kivinen, J., Warmuth, M.K.: Exponentiated gradient versus gradient descent for linear predictors. Infornation and Computation 132(1), 1–63 (1997)

[12] Kumon, M., Takemura, A., Takeuchi, K.: Sequential optimizing strategy in multidimensional bounded forecasting games. CoRR abs/0911.3933v1 (2009)

[13] Lamperti, J.: Stochastic Processes: A Survey of the Mathematical Theory. Springer, Heidelberg (1977)

[14] Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press, Cambridge (2006)

[15] Saunders, C., Gammerman, A., Vovk, V.: Ridge regression learning algorithm in dual variables. In: Proceedings of the 15th International Conference on Machine Learning, pp. 515–521 (1998)

[16] Seeger, M.W., Kakade, S.M., Foster, D.P.: Information consistency of nonparametric Gaussian process methods. IEEE Transactions on Information Theory 54(5), 2376–2382 (2008)

[17] Vovk, V.: Competitive on-line statistics. International Statistical Review 69(2), 213–248 (2001)

[18] Zhdanov, F., Vovk, V.: Competing with gaussian linear experts. CoRR abs/0910.4683 (2009)

Author Index

Abernethy, Jacob 270
Alquier, Pierre 35
Angluin, Dana 104, 194
Aspnes, James 104

Bailly, Raphaël 74
Balle, Borja 179
Bartlett, Peter L. 34, 270
Bartók, Gábor 224
Blondel, Vincent D. 89
Bratko, Ivan 32
Buchbinder, Niv 270

Carbonell, Jaime 50
Case, John 285
Castro, Jorge 179
Chernov, Alexey 255
Clark, Alexander 11
Csáji, Balázs Csanád 89

Denis, François 74
Doliwa, Thorsten 209

Eisenstat, David 194

Gao, Wei 59
Gavaldà, Ricard 179

Habrard, Amaury 74
Hanneke, Steve 50
Higgs, Matthew 148
Hirowatari, Eiju 315
Hoi, Steven C.H. 390
Hutter, Marcus 1, 360

Jain, Sanjay 330
Jin, Rong 390
Jungers, Raphaël M. 89

Kalnishkan, Yuri 405
Kinber, Efim 330
Kontorovich, Leonid (Aryeh) 194
Koolen, Wouter M. 239
Kötzing, Timo 285
Kudo, Mineichi 375

Laviolette, François 119
Lever, Guy 119
Lv, Jiawei 163

Moelius III, Samuel E. 300

Nakamura, Atsuyoshi 375

Orseau, Laurent 345

Pál, Dávid 224
Pestov, Vladimir 134

Ramamohanarao, Kotagiri 33
Reyzin, Lev 104, 194
Rooij, Steven de 239

Shawe-Taylor, John 119, 148
Simon, Hans Ulrich 209
Stanton, Isabelle 270
Stephan, Frank 1
Sugiyama, Mahito 315
Sunehag, Peter 360
Szepesvári, Csaba 224

Tsuiki, Hideki 315

Uchiya, Taishi 375

Vovk, Vladimir 1

Wang, Fei 163
Wang, Zheng 163
Warmuth, Manfred K. 31

Yamamoto, Akihiro 315
Yang, Liu 50
Yang, Tianbao 390

Zeugmann, Thomas 1
Zhang, Changshui 163
Zhang, Jianwen 163
Zhdanov, Fedor 255, 405
Zhou, Zhi-Hua 59
Zilles, Sandra 209, 300