NEW FORMULATIONS FOR ACTIVE LEARNING

A Thesis
Presented to
The Academic Faculty

by

Ravi Sastry Ganti Mahapatruni

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Computer Science

Georgia Institute of Technology
May 2014

NEW FORMULATIONS FOR ACTIVE LEARNING

Approved by:

Professor Maria-Florina Balcan,
Committee Chair
School of Computer Science
Georgia Institute of Technology

Professor Alexander Gray, Advisor
School of Computational Science and
Engineering
Georgia Institute of Technology

Professor Alexander Rakhlin
Department of Statistics
University of Pennsylvania, The Wharton School

Professor Le Song
School of Computational Science and
Engineering
Georgia Institute of Technology

Professor Tong Zhang
Department of Statistics
Rutgers University

Date Approved: Nov 27, 2013

To Mummy, Daddy and Vishal

ACKNOWLEDGEMENTS

It has been a pretty long journey, and this would not have been possible without the help of a lot of people. Alex has been one of the most important people in this long endeavour. Alex backed me during my initial years as a graduate student, when it was not clear to me, what I wanted to do, gave me tremendous academic freedom to pursue research in areas of my choice, and never pressurized me to do things that I was not particularly interested in. Last, but not the least he has been a constant source of funding throughout these six plus years. For this I would like to extend a big Thank You! to Alex. I wrote my first research paper with Adam Kalai. Working with Adam was tremendous fun, and a great learning process. It would be great if I could get an opportunity to work with him in the future. Nina's presence in the machine learning group at Georgia Tech has been wonderful. Many a times, we would cross paths in the CSE corridor, and without fail she would ask me how things were going, and what problem I was working on. Nina invited me to attend her reading group, and present papers in the group, all of which have been good learning experiences. Reviewing conference papers with Nina, was a good learning experience, and many a times I had a chance to learn of some interesting results, even before they were published. Unfortunately, we have not had a paper together, but I hope to change that one day. Sasha and Jacob Abernethy's beautifully written paper on online bandit linear optimization got me truly hooked to the world of bandits. Sasha, without any hesitation, agreed to serve on my dissertation committee, was generous with time, and prompt with e-mail replies. If there is one person that I would like to emulate, then I think it is Sasha. Spasiba, Sasha for providing me with a letter of reference when most needed. I would also like to thank Tong Zhang, and Le Song

for agreeing to serve on my thesis committee, and for being generous with time and advice. Thanks to the entire CSE administrative staff, for all their help over these years.

Machine learning is an extremely interdisciplinary field, and in the process I leaned a good deal of Probability, Statistics, and Optimization. I would like to thank Arkadi Nemirovski, Renato Monteiro, Vladimir Koltchinskii, Yuri Bakhtin, and Christian Houdré for being excellent teachers.

My first steps in research happened in my third year of undergraduate studies at IIT Kharagpur, under the wise guidance of Monojit Choudhury. For the help in those early years, big Thank You! to Monojit. I would also like to thank Sudeshna Sarkar, Anupam Basu, and Niloy Ganguly for having provided me with letters of references to grad school, which made this journey possible at the first place. Special thanks to Pallab Dasgupta for teaching beautifully an introductory course in Artificial Intelligence, which ignited in me a fascination for AI.

My internships at MSR-Bing under the guidance of Wei Chu and Lihong Li and at eHarmony under Vaclav Petricek were great learning experiences. Special thanks to Lihong for having provided me with letters of reference during job hunt. I look forward to working with all of them in the future.

My experience as a graduate student benefited tremendously thanks to the camaraderie of FASTlab members. Big Thank You! to all of you. Special thanks to Krishna, Nishant (who was also my roomate for four years), Parikshit and Dongryeol for many research and non-research discussions, and fun times outside of lab. Special thanks go to Krishna and Parikshit for pushing me to look for an internship in the summer of 2013. I think they have this ability of taking excellent decisions about me. Also, Nishant and Krishna take the sole responsibility of running a student seminar on Empirical Processes and Concentration Inequalities, which culminated in a wonderful graduate level course on the same that was taught by the master. Dongryeol

was my first mentor at grad school, and I thank him for that early guidance, and helping me out whenever required. Vanya, I would like to thank you a ton, for having taught me how to drive, and giving me a chance even when I failed my driving test the first time. The next time we meet, we should play football. Spasiba, Vanya!

At IIT Kharagpur I made some wonderful friends, with whom I have managed to stay in touch. Venu has been a constant source of fun, funda all through the years. Big thank you to Venu! Thanks a lot to Nagu, Shaunak for the warm hospitality on my visit to west coast in the spring break of 2010. Thank you to Siva for the fun times, on my visits back to India. Thanks to Venu, Nagu, Sourish, and Shaunak for the friendship and wonderful memories.

My family has been my biggest support all my life. They have been my role models, a never ending source of encouragement, wise guidance, and everything that I could ask for. They continue to take immense pride even in my little achievements. Thank you Ammamma, Tatagaru, Prasad mava, Chinni mava, Lakshmi atta, Aparna atta, all of my cousins, and extended family members for their love and support. Special thanks to Badrinath and Ram for the timely advice, that set me on the right path. Prathyusha, my wife, has been very tolerant of me, and a source of great comfort during the taxing last semester, when I was trying to wrap up my PhD thesis and graduate. She forgave me when I forgot to call her even for a few minutes, and at times has been too nice to me even when I felt I did not deserve it. I hope I can be equally supportive of her in all her future endeavours. I would like to thank my parents-in-law for having raised such a beautiful daughter.

No words can describe the amount of sacrifices that my parents, and my brother took to help me in all of my endeavours. My mom and my brother gave up a comfortable life in Vizag, to move with me to Hyderabad during JEE preparation. My dad has always been generous with time and money. He never flinched back when it came to spending money on buying books, and would always test me with tricky

mathematical questions. I guess those initial tricky questions ignited my interest in Mathematics. This thesis is dedicated to them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

In this thesis we are concerned with the design and analysis of practically efficient pool based and stream based active learning algorithms for the problem of binary classification. The novelty of this work is that we view active learning from the lens of sequential analysis, which allows us to borrow well established tools from stochastic optimization and multi-armed bandits. In Chapter 1 we discuss the importance of active learning in machine learning and survey some important techniques that have been proposed in active learning literature. We briefly review the stochastic mirror descent algorithm in Chapter 2 and state some important results that are used in the future chapters. In Chapter 3, we provide a generic pool based active learning framework, that will form the basis of our pool based algorithms in Chapters 3, 4. In Chapter 3, we provide an efficient pool based active learning algorithm called UPAL for the class of linear hypothesis that uses importance sampling to construct unbiased estimate of the risk of a hypothesis. We show that UPAL learns a hypothesis which is an exponentially weighted average of different linear hypothesis. An experimental evaluation demonstrates good performance of UPAL over other competing pool based active learning algorithms. In the second half of this chapter we introduce another pool based active learning algorithm called SGD-AL that uses the stochastic gradient descent algorithm, followed by Euclidean projections onto a L_2 ball. SGD-AL uses importance sampling to obtain unbiased estimates of the gradient. We establish excess risk guarantees for SGD-AL, which prove that SGD-AL does just as well as a passive learning algorithm for the class of linear hypothesis of bounded L_2 norm.

In Chapter 4, we view the problem of active learning from the multi-armed bandit lens. By making an explicit analogy between active learning and multi-armed bandits,

we are able to devise a pool based active learning algorithm, called LCB-AL, that utilizes lower confidence bounds and self-concordant barrier type regularizer. Experimental evaluation demonstrate better performance of LCB-AL over UPAL and other competing algorithms.

In Chapter 5, we consider the problem of learning a convex aggregation of a given set of classification models. We propose a stream based active learning algorithm, called SMD-AMA, that uses a stochastic mirror descent algorithm, with the entropy regularizer, to minimize an unbiased estimate of the risk of a convex aggregation. We establish excess risk guarantees of our algorithm, and perform an experimental comparison with a passive learner and other competitive active learning algorithms. Our experimental results show that in certain cases, SMD-AMA achieves the same accuracy as a passive learning algorithm but by querying less than 13% of the total available labels.

CHAPTER I

INTRODUCTION

Machine learning techniques have become popular in many fields such as Astronomy, Physics, Biology, Chemistry, Web Search, Finance. With the availability of large amounts of data, and computation, newer machine learning algorithms, and applications have been discovered. A very popular subclass of machine learning problems fall under the category of supervised learning problems. Classification and regression are the two most popular problems in supervised learning, where the learner is provided with labeled data, and is required to predict the labels of unseen points. In the case of classification problems, these labels are discrete, whereas in regression problems the labels are continuous.

Supervised learning critically relies on the presence of labeled data. The cost of obtaining labels for different data points depends on the problem domain. For example, in Astronomy it is easy to get access to tons and tons of unlabeled data. A prime example is the Fermi Gamma-ray Space telescope[1] that intends to study various phenomena in astrophysics by performing an all sky survey. One can easily expect that such a scientific endeavour would require collecting lots of data. However, obtaining labels for the gathered data is usually hard. In biological problems such as gene sequencing, labeling data is tedious and requires Ph.D. level expertise. In problems of speech recognition, while hours of speech data is easily available, labeling speech utterances is tedious, and requires language expertise. For example [95] report that annotating at word level can take more time than the length of the actual audio itself. Similarly, for problems of information extraction from web documents, locating

[1] `http://en.wikipedia.org/wiki/Fermi_Gamma-ray_Space_Telescope`

entities and relations, can take half-hour or more even for simple documents [85, 83]. In such cases, a natural question that arises is, that can we learn with limited supervision?

1.1 Learning With Limited Supervision

Active Learning (AL) and semi-supervised learning are two machine learning paradigms that have emerged in response to the problem of absence of labeled data. In a typical active learning scenario we assume that we have access to a labeling oracle $\mathcal{O}$, which when provided as an input $x \in \mathcal{X}$ sampled from the underlying marginal distribution $\mathcal{D}_{\mathcal{X}}$, provides us with the label $y \in \mathcal{Y}$, sampled from $\mathbb{P}[Y|X = x]$. In the case of binary classification, which is what we will be interested in this thesis, $\mathcal{Y} = \{-1, +1\}$. Various flavours of AL have been proposed in the past, namely membership query (MQ) based algorithms, stream based algorithms and pool based algorithms. All these three kinds of AL algorithms query the oracle $\mathcal{O}$ for the label of the point, but differ in the nature of the queries. In MQ based algorithms the active learner can query for the label of any point in the input space $\mathcal{X}$, but this query might not necessarily be from the support of the marginal distribution $\mathcal{D}_{\mathcal{X}}$. With human annotators, MQ algorithms might work poorly as was demonstrated by Lang and Baum [11]. Lang and Baum studied the problem of MQ based learning for handwritten digit recognition, and observed that the annotators were faced with the awkward situation of labeling semantically meaningless images. Stream based AL algorithms [30, 14, 29] sample a point x from the marginal distribution $\mathcal{D}_{\mathcal{X}}$, and decide on the fly whether to query $\mathcal{O}$ for the label of x. Stream based AL algorithms tend to be computationally efficient, and appropriate when the underlying distribution changes with time. Pool based AL algorithms [68] assume that one has access to a large pool $\mathcal{P} = \{x_1, \ldots, x_n\}$ of unlabeled i.i.d. examples sampled from $\mathcal{D}_{\mathcal{X}}$, and given budget constraints B, the maximum number of points they are allowed to query, query the most informative set

of B points. Both pool based AL algorithms, and stream based AL algorithms overcome the problem of awkward queries, which MQ based algorithms face. In this thesis we shall work in the pool and stream setting for the binary classification problem.

1.2 Active Learning as a Sequential Analysis Problem

A lot of algorithms have been proposed and designed for active learning (see Section 1.3 for a brief overview of past approaches to active learning). To our knowledge, there has been almost no work up until now that has been successful in designing an active learning algorithm, that is computationally efficient, both in theory and practice, and also has provable guarantees. In this thesis, we intend to bridge the gap between theory and practice of active learning for binary classification problems. We do this by making connections between the problem of active learning and sequential analysis. Sequential analysis aims at reducing the number of samples required for reliable statistical inference by using sequentially, and adaptively collected data rather than using "passively" collected data. Classically sequential analysis has encompassed problems such as sequential hypothesis testing [94, 9, 86], sequential estimation [41], stochastic approximation [61], stochastic adaptive control [60] and multi-armed bandits (MAB) [79, 2, 6]. **The aim of this thesis is to show that active learning is also a member of this family, and one could use techniques that have been classically used in sequential analysis to design algorithms for the problem of active learning which are computationally efficient and also have theoretical guarantees.** With the sequential analysis viewpoint in mind, we design four algorithms for active learning in binary classification problems. The advantage of such a sequential viewpoint is that it allows us to build practically efficient algorithms, with provable guarantees on the excess risk of the learned hypothesis, which also have good empirical performance. All of our four proposed algorithms are

1. Easier to implement, and practically more efficient than version space based

approaches, which are considered to be the state-of-art in theoretical active learning literature.

2. Well grounded and principled as they rely on established ideas from literature in sequential analysis such as multi-armed bandits, bandit linear optimization, and stochastic convex optimization.

1.3 Related Work

A lot of active learning algorithms have been proposed in the past, and [83] provides a very comprehensive survey of different techniques. In this thesis, we shall review only the most popular active learning strategies that have been proposed in the literature.

1. **Uncertainty sampling based strategies.** This is perhaps the most well known, and the most commonly used querying strategy [63]. The idea is to query a point whose label we are most uncertain about. For example, when actively learning SVMs, a simple uncertainty sampling based strategy would be to query for the label of a point which is closest to the SVM hyperplane [89]. In the case of active learning with probabilistic models techniques such as margin sampling [82], entropy based sampling [84] have been proposed; which fall into the general framework of uncertainty based sampling strategies. A fundamental problem with most algorithms that fall in this framework is that, the queried samples end up being from a very different distribution than the target distribution. For example, if we were actively learning SVMs, then the strategy of querying points close to the decision boundary would yield a sampled distribution which will be concentrated mostly close to the decision boundary. However, the target distribution can be very different, which can lead to a large generalization error.

2. **Query-By-Committee framework.** The query-by-committee (QBC) framework was initially proposed by Seung et al. [37]. They considered a Bayesian setting where, the target hypothesis is drawn from a known prior distribution. They designed a stream based AL algorithm, which for each unlabeled point in the stream, draws two hypothesis from the posterior distribution over the hypothesis space and queries for the label of the point, only if the two sampled hypothesis disagree on the label of the point. Non-Bayesian approaches to QBC have also been implemented in various forms via the use of ensemble based methods [66, 70], and probabilistic models [69].

3. **Version space based algorithms.** Cohn, Atlas, and Ladner [30] proposed a simple AL algorithm [2] in the stream setting. The algorithm maintains a hypothesis space, that is consistent with all the labels seen till now, and queries for the label of the point, if there exist two hypothesis that do not agree on the label of the current point. For the pool setting, Gonen et al. [43] introduced a version space based active learning algorithm, called ALuMa, for learning half spaces under the margin assumption. They provide a guarantee on the ratio of number of labels queried by ALuMa, to the number of queries that will be required by an optimal active learning algorithm in the worst case. This ratio scales as $O(d \ln(\frac{1}{\gamma}))$, where γ is the true margin. ALuMA is a greedy algorithm, that maintains a version space consistent with previously labeled points, and queries a point that in the worst case, over its label, leads to a new version space of smallest volume. The major drawback of both CAL and ALuMA are that they are both not robust to cases when the data is not separable. Modifications of the CAL algorithm have been proposed that can handle non-separable data gracefully. Roughly speaking, given a hypothesis space $\mathcal{H}$, that needs to be

[2] This algorithm has commonly been called as the CAL algorithm in active learning literature

actively learned, modified CAL type algorithms, henceforth called version space algorithms, maintain a current version space, $\mathcal{H}_i \subseteq \mathcal{H}$ of the hypothesis space, the error rate of whose members is not very large. Calculation of upper bounds on the error rates of different hypothesis can be done via the use of standard VC bounds [93]. They then query points from the disagreement region of $\mathcal{H}_i$, i.e. those set of points in $\mathcal{X}$, on whose label, hypothesis from $\mathcal{H}_i$ disagree on. The algorithms of Dasgupta et al. [31],Hsu [52], Balcan et al. [7], Hanneke [46], and Koltchinskii [59] fall in this category, and proceed via explicitly or implicitly maintaining a version space. Version space type algorithms have yielded strong theoretical guarantees on the excess risk of the learned hypothesis [47], and the number of labels required to be queried to attain this error rate. The first version space based algorithm with label complexity guarantees was the A^2 algorithm. This analysis has been generalized by the pioneering work of Hanneke [44], via the concept of disagreement coefficient.

4. **Importance weighted based active learning strategies.** Importance weighting for sample bias correction is a standard technique in many machine learning applications [6, 87, 88]. For stream based AL, an importance weighted algorithm was first proposed by Beygelzimer et al. [14], and later extended to the pool based setting by Ganti et al. [39, 40]. In a typical importance-weighted strategy in the stream setting, each point that has been sampled from the underlying distribution, is assigned a probability, p, of being queried. If the point is queried, then the importance weight of this point is set to $\frac{1}{p}$, otherwise the importance weight is set to 0. The importance weighted sample is then used for training with a standard passive learning algorithm. One of the most attractive features of importance weighted strategies is that the importance weighted sample provides an unbiased estimator of the loss of a hypothesis, not only for the hypothesis class that was chosen for generating the importance-weighted

sample, but also for any other different hypothesis class that might be used for training in the future, with this importance-weighted sample. This is in contrast to the previously listed strategies where, the queried data is strongly tied to the hypothesis class that was used to actively query the data. As a result an actively queried dataset, generated using one hypothesis class via importance weighted strategies, can now be employed in the future with a different hypothesis class without the problem of sampling bias.

5. **Active learning with convex loss functions.** Beygelzimer et al. [14] introduced the importance-weighting technique for stream based active learning, and designed an active learning algorithm, called IWAL (loss-weighting). IWAL (loss-weighting) uses convex loss functions that are commonly used in passive learning. To study the label complexity of active learning with convex losses, they introduced a new problem dependent quantity called generalized disagreement coefficient, which is a direct generalization of disagreement coefficient, and a property of loss function called loss asymmetry, which for differentiable losses is upper bounded by the ratio of maximum absolute value of derivative of the loss function to the minimum absolute value of derivative of the loss function. The authors analyzed the label complexity of IWAL (loss-weighting) in terms of the generalized disagreement coefficient, and loss asymmetry. However, their analysis is loose, and for certain losses such as hinge loss, squared loss does not yield meaningful results. Hanneke and Yang [45] introduced a version space based algorithm, which uses convex risk minimization procedures, to obtain the intermediate version space. In contrast to typical version space based algorithms (as mentioned in our bullet point 3), the algorithm of Hanneke and Yang, is provably efficient for most reasonable hypothesis spaces, as it solves a convex optimization problem. In practice though, the algorithm can be practically inefficient, as it requires explicitly maintaining the current version space,

via convex inequalities.

6. **Stochastic convex optimization and active learning.** There has been some preliminary work [76, 77] connecting stochastic convex optimization and active learning. The main result has been that the rates for active learning in 1-dimension match those of stochastic convex optimization, if the Tsybakov noise condition [92] is satisfied, and if the marginal distribution is uniform on the interval $[0, 1]$. However, this work is still pretty much preliminary.

7. **Adaptive submodularity and Active Learning.** Golovin and Krause [42] introduced the notion of adaptive submodularity, an extension of the notion of submodularity from sets to adaptive policies. For the problem of pool based active learning, the authors introduced an adaptive submodular objective function, for which they suggested a greedy algorithm, whose computational complexity scales linearly with the size of the hypothesis class. Such ideas, are appropriate when the size of the hypothesis class is finite, and we want to detect the optimal hypothesis. Almost always, the size of our hypothesis class is infinite, and we do not want to precisely detect the best hypothesis. Instead, we want to find a hypothesis whose excess risk w.r.t. the best hypothesis in the hypothesis class is small.

8. **Other Strategies.** Many other strategies, than the ones mentioned above, have been proposed. Settles [83] provides a comprehensive survey of various other strategies, and we encourage the interested reader to take a look at this survey.

1.4 Summary of Contributions

In Chapter 3, we first design a generic pool based active learning framework. An active learning algorithm, operating in this framework proceeds in rounds, and in each round

performs two steps. In the first step, we place a sampling distribution over the pool, from which we sample one point, and query for the label of the point. The sampling distribution, in each round, depends on our current hypothesis and all the points in the pool. By the use of importance weights, we are able to derive importance weighted estimators of the risk of a hypothesis, which are provably unbiased estimators of the risk. In the second step, using these unbiased estimators of risk, we update our current hypothesis.

We introduce two pool based active learning algorithms called UPAL, and SGD-AL, that operate in the generic framework mentioned in the above paragraph. UPAL uses sampling in proportion to the conditional entropy of the label distribution, and a model [3] update procedure that solves an ERM problem minimizing the importance weighted risk over the given hypothesis class. In the case of UPAL, we show that for the class of potentially unbounded norm linear hypothesis, and squared loss, UPAL learns a hypothesis that is similar to an exponentially weighted average hypothesis. We analyze the statistical efficiency of UPAL with the squared loss, for a regression problem, where the response y, for any x, sampled for the underlying marginal distribution belongs to $[-1, 1]$. We make the additional assumption that the underlying regression model is linear. Under these conditions, we analyze the number of labeled and unlabeled samples required for exact recovery of the linear model. To our knowledge UPAL is the first active learning algorithm, proposed in the literature, that uses an importance weighting scheme in the pool based setting. In the second part of the chapter, we consider the problem of pool based active learning, when the hypothesis class is the set of linear hypothesis of bounded norm, i.e. $\mathcal{H} = \{h \in \mathbb{R}^d : ||h|| \leq R\}$. For this class, we design a new active learning algorithm called SGD-AL, that uses stochastic gradient descent, along with projections to update the current hypothesis.

[3] We shall use the words model and hypothesis interchangeably throughout the thesis. Similarly, model class and hypothesis class will be used interchangeably throughout the thesis.

With a carefully designed probability distribution, we show that the final hypothesis outputted by SGD-AL is no worse than a passive learner (see Theorem 8 in Chapter 3).

In Chapter 4, we use ideas from the literature of MAB to design an algorithm that works in the generic pool based AL framework, introduced in Chapter 3. We provide an equivalence between the MAB problem and AL. We do this by identifying what is the conceptual role of the arms, and the loss signal in a MAB problem. We then show that, for the AL problem, one could think of the different hypothesis as arms of a MAB, and the label information obtained by querying the oracle, as providing an implicit reward signal. Once this analogy is made clear, we use standard techniques from MAB literature such as lower-confidence bounds, barrier type regularization to design a pool based active learning algorithm called LCB-AL. Via the Bernstein inequality for martingales, we construct high probability lower confidence bounds on the risk of a hypothesis. We use minimization of the lower confidence bound over the hypothesis space to update our current model, and sample points from the pool as per a sampling distribution, that falls out of analyzing the role of queried labels as an implicit loss signal. We show via experiments, that LCB-AL achieves good accuracy by querying for a lot fewer labels than what passive learning algorithms would do.

In Chapter 5, we consider the problem of convex aggregation of a given set of models. In the classical problem of learning a convex aggregation [71], we are given models $f_1, \ldots, f_M$, and labeled training data. Using this data, one is required to learn a convex aggregation of models, that does as well as the best convex aggregation of models. Precisely, given a margin based loss function $L(\cdot)$, we are interested in procedures which output a convex combination $\hat{\theta} \in \Delta_M$, where $\Delta_M = \{\theta \in \mathbb{R}_+^M : \sum_{j=1}^M \theta_j = 1\}$ is the $M-1$ dimensional probability simplex, such that

$$\mathbb{E}L\left(y\sum_{j=1}^{M}\hat{\theta}_j f_j(x)\right) \leq \min_{\theta\in\Delta_M} \mathbb{E}L\left(y\sum_{j=1}^{M}\theta_j f_j(x)\right) + \delta_{n,M}, \tag{1}$$

where $\delta_{n,M} > 0$ is a small quantity that goes to 0 as $n \to \infty$. We consider an active variant of this problem, where instead of being given fully labeled data, we are allowed to query an oracle for the labels of the data. Working in the streaming setting, we propose a slight variant of the stochastic mirror-descent algorithm, called SMD-AMA that uses uncertainty sampling type strategy to actively learn a convex combination of the given models. We establish excess risk guarantees for the convex aggregate returned by SMD-AMA, to be of the order of $O\left(\sqrt{\frac{\log(M)}{T^{1-\kappa}}}\right)$, where T is the length of the stream, and $\kappa > \frac{1}{2}$ is an algorithm dependent parameter, that trade-off the number of labels queried, and the excess risk. Large κ leads to a smaller lower bound on the number of queries made, but a larger upper bound on the excess risk of the convex aggregate returned by SMD-AMA (see Theorem 12 in Chapter 5). We demonstrate experimentally, that our active learning algorithm, in most cases, has an error rate comparable to that of a passive learning algorithm, even though we query far fewer labels than passive learning does. For certain datasets, the label savings can be as much as 87%.

CHAPTER II

A PRIMER ON MIRROR DESCENT ALGORITHM

Chapters 3 and 5 introduce the SGD-AL and AMA-SMD algorithms respectively, which rely on an implementation of the stochastic mirror descent algorithm. In this chapter we shall introduce the mirror-descent algorithm, and the stochastic mirror descent algorithms, and state a few results which shall be used later on. We intend to keep this chapter as brief as possible, as it is very standard. Some excellent references from where we took most of the material in this chapter are [12] and [19] (See also the numerous references in [19]).

2.1 Mirror Descent Algorithm

The gradient descent algorithm is perhaps the most popular first order optimization techniques in convex optimization. Given a convex function $f : \mathbb{R}^d \to \mathbb{R}$, consider the unconstrained convex optimization problem: $\min f(x)$. Gradient descent is an iterative algorithm, which starts at an arbitrary point x_1 in the domain of f, and performs the following updates for $t \geq 1$

$$x_{t+1} \leftarrow x_t - \eta_t \nabla f(x_t).$$

$\nabla f(x_t)$ is the gradient to the function f at the point x_t. When the function is not differentiable everywhere, then one can use any subgradient in the place of gradient. η_t is called the step size. By using appropriate step sizes, one can guarantee convergence of gradient descent algorithms to an optima x^* of $f(x)$ [17].

For the case of constrained convex optimization problems of the form $\min_{x \in X} f(x)$, where X is a closed convex subset of the domain of f, the gradient descent algorithm along with Euclidean projection onto the constraint set, is a good first order convex

optimization method.

The mirror descent (MD) algorithm, which was first introduced by Nemirovsky and Yudin [73], is a first order algorithm for convex optimization, which can be seen as an extension of the gradient descent algorithm to non-Euclidean geometry. Before we dive into the details of the mirror-descent algorithm we shall need a few definitions.

Definition 1. *Given a convex function $\mathcal{R}$, the conjugate is another function $\mathcal{R}^* : \mathrm{dom}(\mathcal{R}^*) \to \mathbb{R}$ defined as*

$$\mathcal{R}^*(y) = \sup_{x \in X} \langle x, y \rangle - \mathcal{R}(x). \tag{2}$$

Definition 2. *Let $V : X \to \mathbb{R}$ be strictly convex, and continuously differentiable on int(X). The Bregman divergence corresponding to V is a function $D_V : X \times \mathrm{int}(X) \to \mathbb{R}$ defined as*

$$D_V(x_1, x_2) = V(x_1) - V(x_2) - \langle \nabla V(x_2), x_1 - x_2 \rangle \tag{3}$$

Definition 3. *A function $V : X \to \mathbb{R}$ is Legendre if*

1. *V is strictly convex and continuously differentiable on the interior of X.*
2. *$||\nabla V(x)|| \to \infty$ as $x \to \mathrm{bd}(X)$, where $\mathrm{bd}(X)$ is the boundary of X.*

There are two ways to view the mirror descent algorithm. We shall call them primal-dual viewpoint, and the proximal viewpoint.

2.1.1 Primal-Dual Viewpoint

In order to implement a mirror descent algorithm one needs to specify a convex function $\mathcal{R} : X \to \mathbb{R}$, which is strongly convex w.r.t. a chosen norm $||\cdot||$. We assume that the objective function f is Lipschitz continuous w.r.t. the chosen norm. We start with an iterate $x_1 \in \arg\min_{x \in X} \mathcal{R}(x)$. For $t \geq 1$, we generate the following sequence of iterates

1. $\tilde{x}_{t+1} = \nabla \mathcal{R}^*(\nabla \mathcal{R}(x_t) - \eta_t \nabla f(x_t))$.

2. $x_{t+1} = \arg\min_{x \in X} D_{\mathcal{R}}(x, \tilde{x}_{t+1})$.

The second step of the above algorithms guarantees that the next iterate x_{t+1} belongs to the constraint set X.

2.1.2 Proximal Viewpoint

The primal-dual viewpoint that was sketched in the previous section is very unintuitive. The role of the strongly convex function $\mathcal{R}$ is unclear. A very clear understanding of the mirror-descent method is obtained by looking at it from the lens of proximal algorithms. It is well known that the iterates of a gradient descent algorithm for solving a convex optimization problem can be seen as solving the following optimization problem

$$x_{t+1} = \arg\min_{x \in X} \langle \nabla f(x_t), x - x_t \rangle + \frac{1}{2\eta_t} ||x - x_t||^2. \tag{4}$$

By convexity of f, $g(x) \stackrel{\text{def}}{=} f(x_t) + \langle x - x_t, \nabla f(x_t) \rangle$ is an under approximation of the function $f(x)$ everywhere. Equation 4 tries to minimize a lower bound of $f(x)$, by considering the subgradient of f at x_t, without going too far off from the current iterate x_t. An equivalent interpretation for mirror-descent algorithm can be provided as follows. Iterate x_{t+1} is the solution to the optimization problem

$$x_{t+1} = \arg\min_{x \in X} \langle x - x_t, \nabla f(x_t) \rangle + \frac{1}{2\eta_t} D_{\mathcal{R}}(x, x_t). \tag{5}$$

Beck and Teboulle showed that if $\mathcal{R}$ is a strongly convex, Legendre function, then the iterates of MD algorithm are the same as the iterates generated by solving the optimization problem given in Equation 4. This is enlightening as it tells us that mirror-descent is like a gradient descent algorithm, but uses the geometry induced by a strongly convex function $\mathcal{R}$. If $\mathcal{R}(x) = \frac{1}{2}||x||^2$, then we recover the projected gradient descent algorithm. . A common way of using the mirror descent algorithm is to choose an appropriate norm $||\cdot||$, and a regularization function $\mathcal{R}$ which is strongly convex w.r.t. the chosen norm. For example,

1. If $X = \Delta_M$, the $M-1$ dimensional simplex, then a popular choice of $\mathcal{R}$ is the entropy regularizer, defined as $\mathcal{R}(x) = -\sum_{i=1}^{M} x_i \log(x_i)$. $\mathcal{R}(x)$ is 1-strongly convex w.r.t. $||\cdot||_1$ norm.

2. Consider the following example [19]. Suppose $X = \{Q \in \mathbb{R}^{d\times d} : Q \succ 0\}$. Define the norm $||\cdot||$ as $||x|| = \sqrt{x^T Q x}$, and take the regularizer $\mathcal{R}(x) = \frac{1}{2} x^T Q x$. $\mathcal{R}(x)$ is 1-strongly convex w.r.t. the defined norm.

A very popular way of using the mirror descent algorithm is with specific kinds of regularization functions which are barriers, and Legendre type. A barrier type regularization function means that $\mathcal{R}(x) = \infty$, for $x \notin X$. Strongly convex, Legendre type regularizers have some nice properties. These properties are well known(e.g. see pages 297-298 of [26]).

Lemma 1. *If $\mathcal{R} : X \to X$ is a strongly convex, Legendre function, then*

1. $(\nabla\mathcal{R})^{-1} = \nabla\mathcal{R}^*$.

2. $D_{\mathcal{R}}(x_1, x_2) = D_{\mathcal{R}^*}(\nabla\mathcal{R}(x_2), \nabla\mathcal{R}(x_1))$.

3. $\mathcal{R}^{**} = \mathcal{R}$, *i.e. the conjugate of the conjugate is the function itself.*

4. *For all $x \in X$ and $y, z \in \text{int}(X)$, we have*

$$D_{\mathcal{R}}(x, y) + D_{\mathcal{R}}(y, z) = D_R(x, z) + \langle x - y, \nabla R(z) - \nabla R(y)\rangle. \tag{6}$$

This result has been called as the three point equality.

2.1.3 Stochastic Mirror Descent Algorithm

The Mirror descent algorithm has also been extended to solve stochastic optimization problems [72], and online learning problems [19]. Given a stochastic optimization problem: $\min_{x\in X}\{f(x) \stackrel{\text{def}}{=} \mathbb{E}_\omega F(x;\omega)\}$, stochastic mirror descent algorithm assumes that we have access to a stochastic gradient oracle, which when provided with a point

$x \in X$, provides us with $g(x;\omega)$, which is an unbiased estimate of the gradient of f at x, i.e, $\mathbb{E}_\omega g(x;\omega) = \nabla f(x)$. Mirror descent now proceeds in similar fashion to one shown in Section 2.1.1 , but, in iteration t, now uses $g(x_t;\omega)$ instead of $\nabla f(x_t)$. In the case of online optimization, $g(x_t;\omega)$ is built by using the current data sample that we see during the online learning process.

CHAPTER III

UNBIASED POOL BASED ACTIVE LEARNING ALGORITHMS

3.1 Introduction

In the problem of binary classification one has a distribution $\mathcal{D}$ on the domain $\mathcal{X} \times \mathcal{Y} \subseteq \mathbb{R}^d \times \{-1, +1\}$, and access to a sampling oracle, which provides us with i.i.d. labeled samples $\mathcal{S} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$. The task is to learn a classifier, which predicts well on unseen points. In this chapter we address the problem of pool based active learning, where instead of being given a collection of labeled examples, $\mathcal{S}$, we are now given an unlabeled pool $\mathcal{P}$ of examples sampled from the underlying marginal distribution, $\mathcal{D}_{\mathcal{X}}$, on $\mathcal{X}$. We also have access to a labeling oracle $\mathcal{O}$, which when provided as an input a point $x_i \in \mathcal{P}$, provides a label $y_i \in \{-1, +1\}$ sampled from $\mathbb{P}[y_i|x_i]$. In the pool based active learning scenario it is common to think of being given a budget, B, of the maximum number of points for which we can query oracle $\mathcal{O}$ for the labels of points. As mentioned in Section 1.1 of Chapter 1 pool based active learning, and stream based active learning are perhaps the most natural settings under which active learning problems can be studied.

3.1.1 Contributions and outline of this chapter.

1. We introduce a generic pool based active learning framework, and introduce two algorithms namely UPAL and SGD-AL that operate in this framework.

2. UPAL uses a) a sampling distribution which is in proportion to the conditional entropy of the label distribution, and b) an empirical risk minimization procedure that minimizes an importance weighted estimate of the risk over a linear

hypothesis class to update the model. While unbiased estimators of risk have been used in stream based AL algorithms [14], no such estimators have been introduced for pool based AL algorithms. We provide an unbiased estimator of the risk of a hypothesis, by using the idea of importance weights introduced for AL in Beygelzimer et al. [14].

3. In Theorem 2 (Section 3.3.1) we show that, for the squared loss, and when $\mathcal{H} = \mathbb{R}^d$, UPAL outputs a hypothesis that is equivalent to an exponentially weighted average of all the hypothesis in the hypothesis class. Such exponentially weighted average techniques have been utilized in both online learning with experts [26], and in the design of the EXP4 algorithm for the MAB problem with expert advice [6]. Hence, UPAL can be seen as pruning the hypothesis space, in a soft manner, by placing a probability distribution that is determined by the importance weighted loss of each classifier on the currently labeled part of the pool.

4. In Section 3.4, we analyze the UPAL algorithm, with squared loss, for a regression problem, where the response y, for any x, sampled from the underlying marginal distribution belongs to $[-1, 1]$. We make the additional assumption that the underlying regression model is linear. Under these conditions, we analyze the number of labeled and unlabeled samples required for exact recovery. Our proof employs some results from random matrix theory regarding eigenvalues of sums of random matrices [50, 51, 91].

5. In Section 3.7, we provide a thorough empirical analysis of UPAL comparing it to a batch mode active learning algorithm, which we shall call as BMAL [48], and a simple active learning algorithm, that in each round chooses a random point from the pool, and queries its label. We also empirically demonstrate the scalability of UPAL over BMAL on the MNIST dataset. When provided with

a large budget, UPAL is up to seven times faster than BMAL.

6. While the hypothesis class, $\mathcal{H} = \mathbb{R}^d$, is powerful, it is less widely used in practice. Often times, one implements a learning algorithm with some sort of regularizer, e.g. SVMs minimize empirical hinge loss with the squared norm regularizer, or one explicitly constraints the hypothesis space to be a bounded subset of a metric space. When $\mathcal{H} = \{h \in \mathbb{R}^d : ||h|| \leq R\}$, we propose in Section 3.9, a different pool based active learning algorithm called SGD-AL, which performs active learning via stochastic gradient descent, followed by Euclidean projections onto an L_2 ball. For SGD-AL, we establish excess risk guarantees and show that SGD-AL is no worse than a passive empirical risk minimization procedure.

3.2 A Generic Pool Based Active Learning Framework

We introduce a generic pool based active learning framework, which will be used to design pool based active learning algorithms. A correct active learning algorithm needs to take into account the fact that the points it has queried might not reflect the true underlying marginal distribution. This problem is similar to the problem of dataset shift [75], where the train and test distributions are potentially different, and the learner needs to take into account this bias during the learning process. One approach to this problem is to use importance weights, where during the training process instead of weighing all the points equally the algorithm weighs different points differently. In order to use importance weights in our framework, in each round we perform two steps. In the first step, we put a sampling distribution over the pool. This distribution depends on the current model, and the points in the pool. In the second step, we sample a single point from the distribution with replacement, and query for the label of the point, and using importance weighted estimators of the risk, update our current model. As mentioned in Section 1.3, importance weights

help create unbiased estimators of the risk of a hypothesis, and allow reuse of data with a different hypothesis class.

Algorithm 1 Generic Pool Based Active Learning Framework (Input: $\mathcal{P} = \{x_1, \ldots, x_n, \}$, Budget B, Labeling Oracle $\mathcal{O}$.)

1. Start with an initial hypothesis.

while Not run out of budget **do**

2. Place a sampling distribution over $\mathcal{P}$. This sampling distribution depends on the current hypothesis, and the points in $\mathcal{P}$.

3. Sample a point from this distribution, with replacement, query for the label of the sampled point, and update the current hypothesis using importance weighted estimators.

end while

3.3 Design of the UPAL Algorithm

UPAL algorithm operates in the above framework, by proceeding in rounds, where in each round t, we put a probability distribution $\{p_i^t\}_{i=1}^n$ on the entire pool $\mathcal{P}$, and sample one point from this distribution. As mentioned before, we shall sample with replacement, where a point x_i, if queried in some round, is put back in the pool $\mathcal{P}$, and is available for requerying in the future rounds. If the point, sampled in round t, was queried in one of the previous rounds $1, \ldots, t-1$ then its queried label from the previous round is reused, else the oracle $\mathcal{O}$ is queried for the label of the point. One could avoid requerying points, but this leads to somewhat more complicated expressions for our risk estimator. Hence, for simplicity of exposition we shall allow re-querying of points in this paper. Denote by $Q_i^t \in \{0, 1\}$ a random variable that takes the value 1 if the point x_i was queried for its label in round t and 0 otherwise. In order to guarantee that our estimate of the error rate of a hypothesis $h \in \mathcal{H}$ is unbiased we use importance weighting, where a point $x_i \in \mathcal{P}$ in round t gets an importance weight of $\frac{Q_i^t}{p_i^t}$. Notice that by definition $\mathbb{E}[\frac{Q_i^t}{p_i^t}|p_i^t] = 1$. We formally prove that importance weighted risk is an unbiased estimator of the true risk. Let $\mathcal{D}_n$ denote a product distribution on $(x_1, y_1), \ldots, (x_n, y_n)$. Also denote by $Q_{1:n}^{1:t}$ the collection of

random variables $Q_1^1, \ldots, Q_n^1, \ldots, Q_n^t$. Let $Z_i^t \stackrel{\text{def}}{=} y_i Q_i^t$. Let $R(h) \stackrel{\text{def}}{=} \mathbb{E}_{x,y} L(yh^T x)$ We shall make the following independence assumption:

Assumption 1. *If x_i has not been queried up until the start of round t, then*

$$p_i^t \perp\!\!\!\perp y_i | x_{1:n}, Z_{1:n}^{1:t-1}.$$

Theorem 1. *Let $\hat{L}_t(h) \stackrel{\text{def}}{=} \frac{1}{nt} \sum_{i=1}^n \sum_{\tau=1}^t \frac{Q_i^\tau}{p_i^\tau} L(y_i h^T x_i)$, where $p_i^\tau > 0$ for all $\tau = 1, \ldots, t$. Then, for any $t \geq 1$*

$$\mathbb{E}_{Z_{1:n}^{1:t}, x_{1:n}} \hat{L}_t(h) = R(h). \tag{7}$$

Proof.

$$\begin{aligned}
\mathbb{E}_{Z_{1:n}^{1:t}, x_{1:n}} \hat{L}_t(h) &= \mathbb{E}_{Z_{1:n}^{1:t}, x_{1:n}} \frac{1}{nt} \sum_{i=1}^n \sum_{\tau=1}^t \frac{Q_i^\tau}{p_i^\tau} L(y_i h^T x_i) \\
&= \mathbb{E}_{Z_{1:n}^{1:t}, x_{1:n}} \frac{1}{nt} \sum_{i=1}^n \sum_{\tau=1}^t \mathbb{E}_{Q_i^\tau, y_i | Z_{1:n}^{1:\tau-1}, x_{1:n}} \frac{Q_i^\tau}{p_i^\tau} L(y_i h^T x_i) \qquad (8) \\
&= \mathbb{E}_{Z_{1:n}^{1:t}, x_{1:n}} \frac{1}{nt} \sum_{i=1}^n \sum_{\tau=1}^t L(y_i h^T x_i) \qquad (9) \\
&= R(h). \qquad \square
\end{aligned}$$

In the above proof, we used Assumption 1 along with the fact that $\mathbb{E}[\frac{Q_i^t}{p_i^t} | p_i^t] = 1$ to get to Equation 9, from Equation 8. The theorem guarantees that as long as the probability of querying any point in the pool in any round is non-zero, $\hat{L}_t(h)$ will be an unbiased estimator of $R(h)$. The critical question to answer is, how does one come up with a probability distribution on $\mathcal{P}$ in round t? To solve this problem we resort to probabilistic uncertainty sampling, where the point whose label is most uncertain as per the current hypothesis, $h_{A,t-1}$, gets a higher probability mass. The current hypothesis is simply the minimizer of the importance weighted risk in $\mathcal{H}$, i.e. $h_{A,t-1} = \arg\min_{h \in \mathcal{H}} \hat{L}_{t-1}(h)$. For any point $x_i \in \mathcal{P}$, to calculate the uncertainty of the label y_i of x_i, we first estimate $\eta(x_i) \stackrel{\text{def}}{=} \mathbb{P}[y_i = 1 | x_i]$ using $h_{A,t-1}$, and then use the

entropy, $H(p) \stackrel{\text{def}}{=} -p\ln(p) - (1-p)\ln(1-p)$ of the label distribution of x_i to calculate the probability of querying x_i. The estimate of $\eta(\cdot)$ in round t, depends both on the current active learner $h_{A,t-1}$, and the loss function. In general it is not possible to estimate $\eta(\cdot)$ with arbitrary convex loss functions. Hence, to estimate $\eta(\cdot)$ we use properties of the loss function. It is well known that standard loss functions such as exponential loss, logistic loss, squared loss, modified squared loss, Huber loss which are used in classification are also proper losses for probability estimation. Steps 4, 11 of Algorithm 2 depend on the loss function $L(\cdot)$ being used. If we use the logistic loss i.e., if $L(yh^Tx) = \ln(1+\exp(-yh^Tx))$ then $\hat{\eta}_t(x) = \frac{1}{1+\exp(-h^Tx)}$. In case of squared loss, $\hat{\eta}_t(yh^Tx) = \min\{\max\{0, h^Tx\}, 1\}$. Similar expressions can be derived for other losses too (see, for example, Section 4 in [96]). Since the loss function is convex, and the constraint set $\mathcal{H}$ is convex, the minimization problem in Step 11 of the Algorithm 2 is a convex optimization problem.

3.3.1 The case of squared loss

It is interesting to look at the behaviour of UPAL in the case of squared loss where $L(yh^Tx) = (1 - yh^Tx)^2$. If not mentioned, we shall denote by h_A the hypothesis returned by UPAL at the end of T rounds. We now show that the prediction of h_A on any x, is simply the exponentially weighted average of predictions of all h in $\mathcal{H}$, on x.

Theorem 2. *Let,*

$$z_i \stackrel{\text{def}}{=} \sum_{t=1}^{T} \frac{Q_i^t}{p_i^t} \qquad \hat{\Sigma}_z \stackrel{\text{def}}{=} \sum_{i=1}^{n} z_i x_i x_i^T$$

$$v_z \stackrel{\text{def}}{=} \sum_{i=1}^{n} z_i y_i x_i \qquad c \stackrel{\text{def}}{=} \sum_{i=1}^{n} z_i.$$

Define $w \in \mathbb{R}^d$ as

$$w = \frac{\int_{\mathbb{R}^d} \exp(-\hat{L}_T(h)) h \ \mathrm{d}h}{\int_{\mathbb{R}^d} \exp(-\hat{L}_T(h)) \ \mathrm{d}h}. \tag{10}$$

Assuming $\hat{\Sigma}_z$ is invertible we have for any $x_0 \in \mathbb{R}^d$, $w^Tx_0 = h_A^Tx_0$.

Algorithm 2 UPAL (Input: $\mathcal{P} = \{x_1, \ldots, x_n, \}$, Margin based loss function $L(\cdot)$, Budget B, Labeling Oracle $\mathcal{O}$, $\kappa \geq 0$)

1. Set unique_queries=0, $h_{A,0} = 0$, $t = 1$.
while unique_queries $\leq B$ **do**
2. Set $Q_i^t = 0$ for all $i = 1, \ldots, n$.
for $x_1, \ldots, x_n \in \mathcal{P}$ **do**
3. Set $p_{\min}^t = \frac{1}{nt^\kappa}$.
4. Calculate $\hat{\eta}_t(x_i) = \mathbb{P}[y = +1 | x_i, h_{A,t-1}]$.
5. $p_i^t \stackrel{\text{def}}{=} p_{\min}^t + (1 - np_{\min}^t) \frac{H(\hat{\eta}_t(x_i))}{\sum_{j=1}^n H(\hat{\eta}_t(x_j))}$.
end for
6. Sample a point (say x_j) from p^t.
if x_j was queried previously **then**
7. Reuse its previously queried label y_j.
else
8. Query oracle $\mathcal{O}$ for its label y_j.
9. unique_queries $\leftarrow$ unique_queries+1.
end if
10. Set $Q_j^t = 1$.
11. $h_{A,t} = \arg\min_{h \in \mathcal{H}} \sum_{i=1}^n \sum_{\tau=1}^t \frac{Q_i^\tau}{p_i^\tau} L(y_i h^T x_i)$.
12. $t \leftarrow t + 1$.
end while
13. Return $h_A \stackrel{\text{def}}{=} h_{A,t}$

Proof. By elementary linear algebra one can establish that

$$h_A = \hat{\Sigma}_z^{-1} v_z \tag{11}$$

$$\hat{L}_T(h) = (h - \hat{\Sigma}_z^{-1} v_z)^T \hat{\Sigma}_z (h - \hat{\Sigma}_z^{-1} v_z). \tag{12}$$

Using standard integrals we get

$$Z \stackrel{\text{def}}{=} \int_{\mathbb{R}^d} \exp(-\hat{L}_T(h)) \, \mathrm{d}h = \exp(-c - v_z^T \hat{\Sigma}_z^{-1} v_z) \sqrt{\pi^d} \sqrt{\det(\hat{\Sigma}_z^{-1})}. \tag{13}$$

In order to calculate $w^T x_0$, it is now enough to calculate the integral

$$I \stackrel{\text{def}}{=} \int_{\mathbb{R}^d} \exp(-\hat{L}_T(h)) \; h^T x_0 \, \mathrm{d}w.$$

To solve this integral we proceed as follows. Define $I_1 = \int_{\mathbb{R}^d} \exp(-\hat{L}_T(h)) \; h^T x_0 \, \mathrm{d}h$.

By simple algebra we get

$$I = \int_{\mathbb{R}^d} \exp(-h^T \hat{\Sigma}_z h + 2h^T v_z - c)\ h^T x_0\ \mathrm{d}h \tag{14}$$

$$= \exp(-c - v_z^T \hat{\Sigma}_z^{-1} v_z) I_1. \tag{15}$$

Let $a = h - \hat{\Sigma}_z^{-1} v_z$. We then get

$$\begin{aligned} I_1 &= \int_{\mathbb{R}^d} h^T x_0 \exp\left(-(h - \hat{\Sigma}_z^{-1} v_z)^T \hat{\Sigma}_z (h - \hat{\Sigma}_z^{-1} v_z)\right)\ \mathrm{d}h \\ &= \int_{\mathbb{R}^d} (a^T x_0 + v_z^T \hat{\Sigma}_z^{-1} x_0) \exp(-a^T \hat{\Sigma}_z a)\ \mathrm{d}a \\ &= \underbrace{\int_{\mathbb{R}^d} (a^T x_0) \exp(-a^T \hat{\Sigma}_z a)\ \mathrm{d}a}_{I_2} + \underbrace{\int_{\mathbb{R}^d} v_z^T \hat{\Sigma}_z^{-1} x_0 \exp(-a^T \hat{\Sigma}_z a)\ \mathrm{d}a}_{I_3}. \end{aligned}$$

Clearly I_2 being the integrand of an odd function over the entire space evaluates to 0. To calculate I_3 we shall substitute $\hat{\Sigma}_z = SS^T$, where $S \succ 0$. Such a decomposition is possible since $\hat{\Sigma}_z \succ 0$. Now define $z = S^T a$. We get

$$I_3 = v_z^T \hat{\Sigma}_z^{-1} x_0 \int \exp(-z^T z)\ \det(S^{-1})\ \mathrm{d}z \tag{16}$$

$$= v_z^T \hat{\Sigma}_z^{-1} x_0 \det(S^{-1}) \sqrt{\pi^d}. \tag{17}$$

Using equations (15, 16, 17) we get

$$I = (\sqrt{\pi})^d v_z^T \hat{\Sigma}_z^{-1} x_0\ \det(S^{-1}) \exp(-c - v_z^T \hat{\Sigma}_z^{-1} v_z). \tag{18}$$

Hence we get

$$w^T x_0 = v_z^T \hat{\Sigma}_z^{-1} x_0 \frac{\det(S^{-1})}{\sqrt{\det(\hat{\Sigma}_z^{-1})}} = v_z^T \hat{\Sigma}_z^{-1} x_0 = h_A^T x_0,$$

where the penultimate equality follows from the fact that $\det(\hat{\Sigma}_z^{-1}) = 1/\det(\hat{\Sigma}_z) = 1/(\det(SS^T)) = 1/(\det(S))^2$, and the last equality follows from equation 11. □

Theorem 2 is instructive. It tells us that assuming that the matrix $\hat{\Sigma}_z$ is invertible, h_A is the same as an exponentially weighted average of all the hypothesis in $\mathcal{H}$. This also allows us to interpret UPAL as pruning the hypothesis space in a soft way via

exponential weighting, where the hypothesis that has suffered larger cumulative loss gets lesser weight. Exponential weighted average techniques have been utilized in both online learning with experts [26], and in the design of the EXP4 algorithm for the MAB problem with expert advice [6]. Another point worth noting is that optimization problem as shown in step 11 of Algorithm 2 is over the entire hypothesis space, and is a convex optimization problem as our loss function is convex. This leads to a computationally efficient procedure. In contrast, implicit and explicit version spaced algorithms, rely either on restricting hypothesis spaces, or the use of error-minimization oracles. Both these procedures, are in theory, computationally inefficient in the worst case, as they require minimization of 0-1 loss, which is known to be a computationally hard problem [58].

3.4 Exact Recovery By UPAL for Certain Regression Problems

In this section we perform a statistical analysis of UPAL for certain regression problems. Specifically, we shall analyze UPAL when run with squared loss on a noiseless regression problem, where our oracle $\mathcal{O}$, will now instead return the response value y for any query x chosen from the pool $\mathcal{P}$. We shall make the following additional assumptions

Assumption 2. $\Sigma \stackrel{\text{def}}{=} \mathbb{E}[xx^T]$ *is invertible.*

Assumption 3. $||x_i|| \leq X$ *a.s.*

Assumption 4. $y = \beta^T x$ *a.s., and* $y \in [-1, +1]$.

Assumption 2 is required to guarantee that there is a unique minimizer of the expected squared loss. Assumption 3 is just a boundedness assumption of the input domain. Assumption 4, says that the underlying model is linear with the response values bounded in the range of $[-1, +1]$.

The motivation for using the squared loss is that it leads to closed form solution for h_A, which can then be easily analyzed using some results from random matrix theory [50, 51, 91].

Our main result is that under Assumptions 2-4, given enough unlabeled data, and if UPAL is run with the squared loss, and $\kappa = 1/2$, then with high probability over the sample and the randomness in sampling, $h_A = \beta$.

Theorem 3. *Suppose Assumptions 1- 4 hold. Then for $T \geq T_{0,\delta}$, $n \geq \max(n_{0,\delta}, n_{1,\delta})$, $\kappa = 1/2$, with probability at least $1 - 5\delta$, UPAL exactly recovers the vector β.*

Before we dive into the details of the proof, we would like to present a sketch of the proof of theorem.

Proof Sketch.

1. We first establish in Lemma 2 that conditioned on the matrices $\hat{\Sigma}_z, \hat{\Sigma}$ being invertible ($\hat{\Sigma}$ is the empirical covariance matrix) the hypothesis h_A returned by UPAL is β.

2. Once we have established this simple result, in Lemmas 3, 4, we establish conditions for the matrices $\hat{\Sigma}_z, \hat{\Sigma}$ to be invertible

We will require the following notation in addition to what has been used in Theorem 2

$$\gamma_0 \stackrel{\text{def}}{=} \max\left(\frac{X}{\sqrt{\lambda_{\min}(\Sigma)d}}, 1\right).$$

$$n_{1,\delta} \stackrel{\text{def}}{=} 4608d^2\gamma_0^4(d\ln(5) + \ln(2/\delta)) + \frac{6d\gamma_0^2}{d\ln(5) + \ln(2/\delta)}$$

$$n_{0,\delta} \stackrel{\text{def}}{=} 8\gamma_0^2 d\ln(d/\delta).$$

$$T_{0,\delta} \stackrel{\text{def}}{=} \frac{324X^8\log^2(d/\delta)}{(\lambda_{\min}(\Sigma))^4} + \frac{18\lambda_{\max}(\Sigma)}{\lambda_{\min}(\Sigma)}\ln(d/\delta).$$

Also denote by $\mathbb{E}_t[\cdot] \stackrel{\text{def}}{=} \mathbb{E}_{Q^t_{1:n}|Z^{1:t-1}_{1:n}, x_{1:n}}[\cdot]$, and $\hat{\Sigma} \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^n x_i x_i^T$ be the empirical covariance matrix.

Lemma 2. *Suppose Assumptions 1-4 hold. Then, if matrix $\hat{\Sigma}_z$ is invertible, we get $h_A = \beta$.*

Proof.

$$h_A = \hat{\Sigma}_z^{-1} v_z = \hat{\Sigma}_z^{-1} \sum_{i=1}^{n} z_i y_i x_i = \hat{\Sigma}_z^{-1} \hat{\Sigma}_z \beta = \beta,$$

where in the first step we used the fact that h_A is the active learner outputted by the algorithm after T rounds, and in the third equality we used Assumption 4. □

Lemma 2 makes use of the assumption that $\hat{\Sigma}_z^{-1}$ is well defined. The next lemma establishes conditions under which this matrix is invertible. The key tool is a result regarding the spectra of random matrices. In particular we shall be using the matrix Bernstein bound and the matrix Chernoff bound, which are stated next.

Theorem 4. *(Matrix Bernstein bound [50]) Let $M_1 \ldots, M_n$ be symmetric valued random matrices. Suppose there exist $\bar{b}, \bar{\sigma}$ such that for all $i = 1, \ldots, n$*

$$\mathbb{E}_i[M_i] = 0$$

$$\lambda_{max}(M_i) \leq \bar{b}$$

$$\lambda_{max}\left(\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_i[M_i^2]\right) \leq \bar{\sigma}^2,$$

almost surely, then

$$\mathbb{P}\left[\lambda_{max}\left(\frac{1}{n}\sum_{i=1}^{n} M_i\right) > \sqrt{\frac{2\bar{\sigma}^2 \ln(d/\delta)}{n}} + \frac{\bar{b}\ln(d/\delta)}{3n}\right] \leq \delta.$$

Theorem 5. *(Matrix chernoff bound [91, 50]) Let $v_1, \ldots v_n$ be random vectors such that, for some $b \geq 0$*

$$\mathbb{E}[||v_i||^2 | v_1, \ldots, v_{i-1}] \geq 1, \text{ and } ||v_i|| \leq b,$$

for all $i = 1, \ldots, n$, almost surely. For all $\delta \in (0, 1)$,

$$\mathbb{P}\left[\lambda_{min}\left(\frac{1}{n}\sum_{i=1}^{n} v_i v_i^T\right) < 1 - \sqrt{\frac{2b^2}{n}\ln(d/\delta)}\right] \leq \delta. \tag{19}$$

Theorem 6. *(Eigenvalue bounds of a sum of rank-1 matrices [64]) Let $r_1, \dots r_n$ be random vectors in $\mathbb{R}^d$ such that, for some $\gamma > 0$,*

$$\mathbb{E}[r_i r_i^T | r_1, \dots, r_{i-1}] = I$$
$$\mathbb{E}[\exp(\alpha^T r_i) | r_1, \dots, r_{i-1}] \leq \exp(||\alpha||^2 \gamma/2) \ \forall \alpha \in \mathbb{R}^d.$$

For all $\delta \in (0, 1)$,

$$\mathbb{P}\left[\lambda_{max}\left(\frac{1}{n}\sum_{i=1}^{n} r_i r_i^T\right) > 1 + 2\epsilon_{\delta,n} \vee \lambda_{min}\left(\frac{1}{n}\sum_{i=1}^{n} r_i r_i^T\right) < 1 - 2\epsilon_{\delta,n}\right] \leq \delta,$$

where

$$\epsilon_{\delta,n} = \gamma\left(\sqrt{\frac{32(d\ \ln(5) + \ln(2/\delta))}{n}} + \frac{2(d\ln(5) + \ln(2/\delta))}{n}\right).$$

We will also need Weyl's inequalities (see Chapter 3 of [49]).

Theorem 7. *Let A, B be positive semi-definite matrices. Then*

$$\lambda_{max}(A) + \lambda_{min}(B) \leq \lambda_{max}(A + B) \leq \lambda_{max}(A) + \lambda_{max}(B)$$

Proposition 1. *For any arbitrary $\alpha \in \mathbb{R}^d$, under assumption A1 we have*

$$\mathbb{E}[\exp(\alpha^T \Sigma^{-1/2} x)] \leq 5 \exp\left(\frac{3d\gamma_0^2 ||\alpha||^2}{2}\right). \tag{20}$$

Proof. From Cauchy-Schwarz inequality, we get

$$-||\alpha||\gamma_0\sqrt{d} \leq -||\alpha||\ ||\Sigma^{-1/2}x|| \leq \alpha^T\Sigma^{-1/2}x \leq ||\alpha||\ ||\Sigma^{-1/2}x|| \leq ||\alpha||\gamma_0\sqrt{d}. \tag{21}$$

Also $\mathbb{E}[\alpha^T\Sigma^{-1/2}x] \leq ||\alpha||\gamma_0\sqrt{d}$. Using Hoeffding's lemma we get

$$\mathbb{E}[\exp(\alpha^T\Sigma^{-1/2}x)] \leq \exp\left(||\alpha||\gamma_0\sqrt{d} + \frac{||\alpha||^2 d\gamma_0^2}{2}\right) \tag{22}$$
$$\leq 5\exp(3||\alpha||^2 d\gamma_0^2/2). \qquad \square$$

Lemma 3. *With probability at least $1 - \delta$, each the following two inequalities hold*

1. *$\lambda_{min}(\hat{\Sigma}) \geq \frac{1}{2}\lambda_{min}(\Sigma) > 0$ for $n \geq n_{0,\delta}$.*

2. $\lambda_{max}(\hat{\Sigma}) \leq \frac{3}{2}\lambda_{max}(\Sigma)$ *for* $n \geq n_{1,\delta}$.

Proof. Let $J \stackrel{\text{def}}{=} \sum_{i=1}^{n} \Sigma^{-1/2} x_i x_i^T \Sigma^{-1/2}$. To prove the first part we shall now use the matrix Chernoff inequality. In order to do so, we first need an upper bound on the quantity $||\Sigma^{-1/2}x||$. By the definition of matrix norms, $||\Sigma^{-1/2}x|| \leq X||\Sigma^{-1/2}|| \leq \frac{X}{\sqrt{\lambda_{\min}(\Sigma)}} = \gamma_0\sqrt{d}$. We then get with probability at least $1-\delta$

$$\lambda_{\min}(J/n) \geq 1 - \sqrt{\frac{2d\gamma_0^2 \ln(d/\delta)}{n}} \geq 1/2 \tag{23}$$

for $n \geq n_{0,\delta}$. Now by the definition of $J = n\Sigma^{-1/2}\hat{\Sigma}\Sigma^{-1/2}$, we get

$$\begin{aligned}
\lambda_{\min}(\hat{\Sigma}) &= \frac{1}{\lambda_{\max}(\hat{\Sigma}^{-1})} \\
&= \frac{1}{n\lambda_{\max}(\Sigma^{-1/2}J^{-1}\Sigma^{-1/2})} \\
&= \frac{1}{n||\Sigma^{-1/2}J^{-1}\Sigma^{-1/2}||} \\
&\geq \frac{1}{n||\Sigma^{-1/2}||\ ||J^{-1}||\ ||\Sigma^{-1/2}||} \\
&= \frac{1}{n}\lambda_{\min}(\Sigma)\lambda_{\min}(J) \\
&\geq \frac{\lambda_{\min}(\Sigma)}{2},
\end{aligned}$$

where in the last line we used equation 23. To prove the second part we proceed as follows: Notice that $\mathbb{E}[\Sigma^{-1/2}x_i x_i^T \Sigma^{-1/2}] = I$. From Proposition 1 we have $\mathbb{E}[\exp(\alpha^T\Sigma^{-1/2}x)] \leq 5\exp(3||\alpha||^2 d\gamma_0^2/2)$. By using Theorem 6 we get with probability at least $1-\delta$:

$$\begin{aligned}
\lambda_{\max}(J) &= \lambda_{\max}\left(\frac{1}{n}\sum_{i=1}^{n}(\Sigma^{-1/2}x_i)(\Sigma^{-1/2}x_i)^T\right) \\
&\leq 1 + 6d\gamma_0^2\left[\sqrt{\frac{32(d\ln(5)+\ln(2/\delta))}{n}} + \frac{2(d\ln(5)+\ln(2/\delta))}{n}\right].
\end{aligned} \tag{24}$$

For $n \geq n_{0,\delta}$, we get

$$\begin{aligned}\lambda_{\max}(\hat{\Sigma}) &= \frac{1}{n}\lambda_{\max}(\Sigma^{1/2} J \Sigma^{1/2}) \\ &\leq \frac{1}{n}\lambda_{\max}(\Sigma)\lambda_{\max}(J) \\ &\leq \frac{3n\lambda_{\max}(\Sigma)}{2n}\end{aligned}$$

where in the last step we used Equation 24. Therefore, $\lambda_{\max}(\hat{\Sigma}) \leq \frac{3\lambda_{\max}(\Sigma)}{2}$. This finishes our proof. □

We are now ready to establish conditions for the inverse of $\hat{\Sigma}_z$ to exist.

Lemma 4. *For $T \geq T_{0,\delta}$, $n \geq \max\{n_{0,\delta}, n_{1,\delta}\}$, and $\kappa = 1/2$, with probability at least $1 - 4\delta$ we have $\lambda_{min}(\hat{\Sigma}_z) \geq nT\lambda_{min}(\Sigma)/12 > 0$, and hence $\hat{\Sigma}_z$ is invertible.*

Proof. The idea is to use the matrix Bernstein bound to get a lower bound on $\lambda_{\min}(\hat{\Sigma}_z)$. Let $M'_t \stackrel{\text{def}}{=} \sum_{i=1}^{n} \frac{Q_i^t}{p_i^t} x_i x_i^T$, so that $\hat{\Sigma}_z = \sum_{t=1}^{T} M'_t$. Now $\mathbb{E}_t M'_t = n\hat{\Sigma}$. Define $R'_t \stackrel{\text{def}}{=} n\hat{\Sigma} - M'_t$, so that $\mathbb{E}_t R'_t = 0$. We shall apply the matrix Bernstein inequality to the random matrix $\sum R'_t$. To do so we need upper bounds on $\lambda_{\max}(R'_t)$ and $\lambda_{\max}(\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_t R_t'^2)$. Let $n \geq n_{1,\delta}$. Using Lemma 3 we get with probability at least $1 - \delta$,

$$\begin{aligned}\lambda_{\max}(R'_t) &= \lambda_{\max}(n\hat{\Sigma} - M'_t) \\ &\leq \lambda_{\max}(n\hat{\Sigma}) \\ &\leq \frac{3n\lambda_{\max}(\Sigma)}{2} \stackrel{\text{def}}{=} b_2. \end{aligned} \tag{25}$$

$$\lambda_{\max}\left[\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_t R_t'^2\right] = \frac{1}{T}\lambda_{\max}[\sum_{t=1}^{T}\mathbb{E}_t(n\hat{\Sigma} - M_t')^2] \tag{26}$$

$$= \frac{1}{T}\lambda_{\max}(-n^2T\hat{\Sigma}^2 + \sum_{t=1}^{T}\mathbb{E}_t\sum_{i=1}^{n}\frac{Q_i^t}{(p_i^t)^2}(x_i x_i^T)^2) \tag{27}$$

$$= \frac{1}{T}\lambda_{\max}(-n^2T\hat{\Sigma}^2 + \sum_{t=1}^{T}\sum_{i=1}^{n}\frac{1}{p_i^t}(x_i x_i^T)^2) \tag{28}$$

$$\leq \frac{1}{T}\lambda_{\max}(\sum_{i=1}^{n}\sum_{t=1}^{T}\frac{1}{p_i^t}(x_i x_i^T)^2) \tag{29}$$

$$\leq n\sqrt{T}\lambda_{\max}(\sum_{i=1}^{n}(x_i x_i^T)^2) \tag{30}$$

$$\leq n\sqrt{T}\sum_{i=1}^{n}\lambda_{\max}^2(x_i x_i^T) \tag{31}$$

$$\leq n^2\sqrt{T}X^4 \stackrel{\text{def}}{=} \sigma_2^2. \tag{32}$$

Equation 27 follows from Equation 26 by the definition of M_t' and the fact that in any given round only one point is queried, i.e for a given t and $i \neq j$ we get $Q_i^t Q_j^t = 0$. Equation 28 follows from Equation 27 by using the fact that $E_t[Q_i^t | p_i^t] = 1$. Equation 29 follows from 28 by Weyl's inequality and the fact that $\hat{\Sigma} \succeq 0$. To obtain Equation 30 from Equation 29 we substituted $p_{\min}^t \stackrel{\text{def}}{=} \frac{1}{n\sqrt{t}}$ in place of p_i^t. Finally the remaining set of inequalities follow because of Assumption 3 , and the fact that if p is a vector then $\lambda_{\max}(pp^T) = ||p||^2$.

Using Theorem 4 we get with probability at least $1 - \delta$,

$$\lambda_{\max}(\frac{1}{T}\sum_{t=1}^{T}R_t') \leq \sqrt{\frac{2\sigma_2^2\ln(d/\delta)}{T}} + \frac{b_2\ln(d/\delta)}{T}. \tag{33}$$

This implies that,

$$\lambda_{\max}(n\hat{\Sigma} - \frac{1}{T}\sum_{t=1}^{T}M_t') \leq \sqrt{\frac{2\sigma_2^2\ln(d/\delta)}{T}} + \frac{b_2\ln(d/\delta)}{T}, \tag{34}$$

and hence by Theorem 7

$$\lambda_{\min}(n\hat{\Sigma}) - \frac{1}{T}\lambda_{\min}\left(\sum_{t=1}^{T}M_t'\right) \leq \sqrt{\frac{2\sigma_2^2\ln(d/\delta)}{T}} + \frac{b_2\ln(d/\delta)}{T}. \tag{35}$$

Rearranging the inequality and substituting for σ_2, b_2 as calculated in equations 25, 32, we get

$$\lambda_{\min}(\sum_{t=1}^{T} M_t') \geq nT\lambda_{\min}(\hat{\Sigma}) - \sqrt{2T^{3/2}n^2X^4\ln(d/\delta)} - \frac{3n\lambda_{\max}(\Sigma)\ln(d/\delta)}{2}. \quad (36)$$

By union bound the above stochastic inequality holds with probability at least $1 - 3\delta$. Finally using Lemma 3 to stochastically lower bound the quantity $\lambda_{\min}(\hat{\Sigma})$ by $\lambda_{\min}(\Sigma)/2$, and applying union bound once again we get the desired result. □

3.4.1 Proof of Theorem 3

For $n \geq n_{0,\delta}$ from Lemma 3, with probability $1-\delta$, $\hat{\Sigma}$ is invertible. For $T \geq T_{0,\delta}, n \geq n_{1,\delta}$, from Lemma 4 the matrix $\hat{\Sigma}_z$ becomes invertible with probability at least $1-4\delta$. Conditioned on the invertibility of $\hat{\Sigma}, \hat{\Sigma}_z$, we know from Lemma 2, that we can recover β exactly. Summing up all the failure probabilities via union bound we get the desired result.

3.5 Discussion of Theorem 3

From Theorem 3, we have the following data requirements

1. UPAL needs at least $\max(n_{1,\delta}, n_{0,\delta})$ unlabeled samples.
2. UPAL needs at least $T_{0,\delta} = O\left(\frac{X^8\log^2(d/\delta)}{\lambda^4_{\min}(\Sigma)} + \frac{\lambda_{\max}(\Sigma)}{\lambda_{\min}(\Sigma)}\log(d/\delta)\right)$ labeled samples.
3. A passive learning algorithm, such as an ERM procedure that solves the problem $\min_w \sum_{i=1}^{n}(y_i - w^T x_i)^2$, would need $n_{0,\delta}$ labeled samples. This is because, we know that for an ERM procedure, exact recovery is achieved once $\hat{\Sigma}$ is invertible. From Lemma 3, we know that $\hat{\Sigma}$ is invertible, with high probability, once we have at least $n_{0,\delta}$ samples.

We shall now make appropriate assumptions, and see how the label complexity of UPAL matches with that of a passive learning algorithm. Suppose, the following two

conditions are satisfied

$$\frac{\lambda_{\max}(\Sigma)\lambda^3_{\min}(\Sigma)}{\log(d/\delta)} = O(1). \tag{37}$$

$$\frac{X}{\sqrt{\lambda_{\min}(\Sigma)d}} \leq 1. \tag{38}$$

Under the second condition, we have $\gamma_0 = 1$. Now, let us examine the ratio $\frac{T_{0,\delta}}{n_{0,\delta}}$. This ratio tells us how many more labeled samples are required by UPAL when compared to an ERM procedure. Under the conditions stated in Equations 37, 38, we get

$$\begin{aligned}\frac{T_{0,\delta}}{n_{0,\delta}} &= O\left(\frac{\log(d/\delta)}{d\gamma_0^2\lambda^4_{\min}(\Sigma)}\right) \\ &= O\left(\frac{\log(d/\delta)}{d\lambda^4_{\min}(\Sigma)}\right) \\ &= O(d\log(d/\delta)).\end{aligned}$$

Hence under conditions given by Equations 37, 38, the ratio of the number of labels queried by UPAL to that of an ERM procedure is bounded from above by $O(d\log(d/\delta))$. This result is somewhat disappointing, as it says that UPAL might end up requiring more labels than a passive learning algorithm. In Section 3.9 we shall show how one can derive a stochastic gradient descent inspired active learning algorithms, which performs no worse than a passive learning algorithm. One last comment worth mentioning is that UPAL needs $\max(n_{1,\delta}, n_{0,\delta})$ unlabeled samples. It is easy to see that $n_{1,\delta} > n_{0,\delta}$. This basically means that we need a larger number of unlabeled samples than an ERM procedure would need. The need for extra unlabeled samples can be explained as follows. Notice that in the proof of Theorem 3, we needed to lower bound $\lambda_{\min}(\hat{\Sigma}_z)$. Establishing a lower bound on $\hat{\Sigma}_z$ in turn requires us to establish an upper bound on $\lambda_{\min}(\hat{\Sigma})$ (see Lemmas 3, 4). The second part of Lemma 3 shows that if we have $n_{1,\delta}$ unlabeled samples, then the empirical covariance matrix $\hat{\Sigma}$, satisfies $\lambda_{\max}(\hat{\Sigma}) \leq \frac{3}{2}\lambda_{\max}(\Sigma)$. This is the precise reason why our unlabeled sample complexity is larger than the unlabeled sample complexity of an ERM procedure. [1]

[1]For the ERM procedure that we described above, the unlabeled sample complexity is same as the labeled sample complexity, as all the sampled points come with their labels.

3.6 Related Work

Zhao et al. [97] consider the problem of active learning with expert advice, for the binary classification problem. Given a sequence of unlabeled examples, $x_1, x_2, \ldots$, each of the experts makes a prediction in $[0, 1]$ for each example. The forecaster, which is the active learning algorithm, combines the predictions of these experts to make a forecast on the current point. After making a forecast, the forecaster has a choice of whether to query for the label of the example. Zhao et al., analyze two forecasters, namely a greedy forecaster, and an exponentially weighted average forecaster, and provided upper bounds on the regret measured only on those rounds in which the forecaster makes a query. However, such regret bounds are not very useful, as they do not consider into account those rounds, where the active forecaster did not issue a query.

An interesting line of work [32, 74, 27] known as selective sampling, deals with active learning in the adversarial setting, where the unlabeled points are generated by an adversary, and the label may also be adversarial, or may be stochastic. All the above three cited papers, assume a linear model, and use the regularized least squares estimator as the base learning algorithm. They then estimate the margin of the current point as per the regularized least squares estimate and query a point if the estimated margin is small.

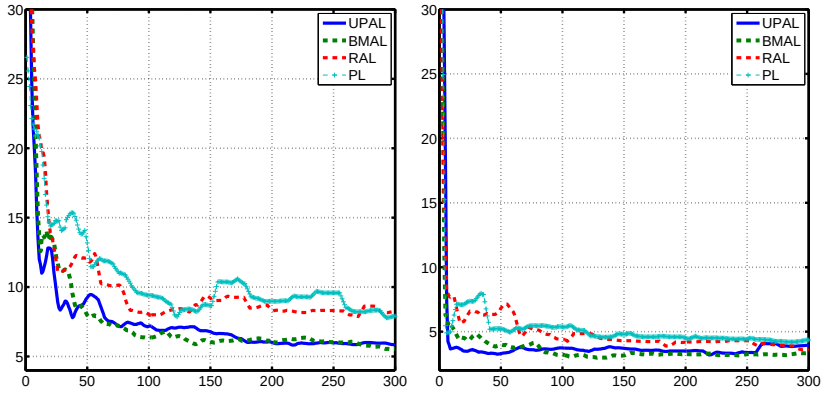
3.7 Experimental results

We implemented UPAL, a standard passive learning (PL) algorithm, a variant of UPAL called RAL (in short for random active learning), and a batch model active learning algorithm described in [48], which we shall call as BMAL, all using logistic loss, in MATLAB. The choice of logistic loss was motivated by the fact that BMAL was designed for logistic loss. Our matlab codes were vectorized to the maximum possible extent so as to be as efficient as possible. RAL is similar to UPAL, but in

each round samples a point uniformly at random from the currently unqueried pool. However, it does not use importance weights to calculate an estimate of the risk of the classifier. The purpose of implementing RAL was to demonstrate the potential effect of using unbiased estimators, and to check if the strategy of randomly querying points helps in active learning.

BMAL algorithm was introduced by Hoi et al. [48]. In their paper, they were able to show superior empirical performance of BMAL over other competing pool based active learning algorithms, and this is the primary motivation for choosing BMAL as a competitor pool based AL algorithm for our experimental study. BMAL, like UPAL, also proceeds in rounds and in each iteration selects k examples by minimizing the Fisher information ratio between the current unqueried pool and the queried pool. However, a point once queried by BMAL is never requeried. In order to tackle the high computational complexity of optimally choosing a set of k points in each round, the authors suggested a monotonic submodular approximation to the original Fisher ratio objective, which is then optimized by a greedy algorithm. At the start of round $t+1$, when BMAL has already queried t points in the previous rounds, BMAL, in order to decide which point to query next has to calculate for each potential new query a dot product with all the queried points. Such a calculation when done for all possible potential new queries takes $O((n-t)t)$ time. Hence if our budget is B, then the total computational complexity of BMAL is $\sum_{t=1}^{B} O(t(n-t)) = O(nB^2)$. Note that this calculation does not take into account the complexity of solving a regularized empirical risk minimization problem in each round after having queried a point. In order to further reduce the computational complexity of BMAL in each round we restrict our search, for the next query, to a subsample of the current set of unqueried points. We set the value of $p_{\min}$ in step 3 of algorithm 1 to $\frac{1}{nt}$. In order to avoid numerical problems we implemented a regularized version of UPAL where the term $\lambda||w||^2$ was added to the optimization problem shown in step 11 of Algorithm

1. The value of λ is allowed to change as per the current importance weight of the pool. We ran all our experiments on the MNIST dataset (3 Vs 5), henceforth called MNIST [2], and datasets from UCI repository namely Statlog, Abalone, Whitewine. All the datasets were scaled to be in the box $[-1, 1]^d$. Figure 1 shows the performance of all the algorithms on the first 300 queried points. On the MNIST dataset, on an



(a) MNIST (b) Statlog

(c) Abalone (d) Whitewine

Figure 1: Empirical performance of passive and active learning algorithms.The x-axis represents the number of points queried, and the y-axis represents the test error of the classifier. The subsample size for approximate BMAL implementation was fixed at 300.

average, the performance of BMAL is very similar to UPAL, and there is a noticeable gap in the performance of BMAL and UPAL over PL, and RAL. Similar results were

[2]The dataset can be obtained from `http://cs.nyu.edu/~roweis/data.html`. We first performed PCA to reduce the dimensions to 25 from 784.

also seen in the case of Statlog dataset, though towards the end the performance of UPAL slightly worsens when compared to BMAL. However, UPAL is still better than PL, and RAL.

Active learning is not always helpful and the success story of AL depends on the match between the marginal distribution and the hypothesis class. This is clearly reflected in Abalone where the performance of PL is better than UPAL at least in the initial stages and is never significantly worse. UPAL is uniformly better than BMAL, though the difference in error rates is not significant. However, the performance of RAL is significantly worse. Similar results were also seen in the case of Whitewine dataset, where PL outperforms all AL algorithms. UPAL is better than BMAL most of the times. Even here one can witness a huge gap in the performance of RAL over PL, BMAL and UPAL.

The uniformly poor performance of RAL signifies that querying uniformly at random does not help. On the whole UPAL and BMAL perform equally well, and we show via our next set of experiments that UPAL has significantly better scalability, especially when one has a relatively large budget B.

3.7.1 Scalability results

Each round of UPAL takes $O(n)$ plus the time to solve the optimization problem shown in step 11 in Algorithm 1. A similar optimization problem is also solved in the BMAL problem. If the cost of solving this optimization problem in step t is $c_{opt,t}$, then the complexity of UPAL is $O(nB + \sum_{t=1}^{T} c_{opt,t})$, where $T \approx B$, while BMAL takes $O(nB^2 + \sum_{t=1}^{B} c'_{t,opt})$ where $c'_{t,opt}$ is the complexity of solving the optimization problem in BMAL in round t. For the approximate implementation of BMAL that we described if the subsample size is $|S|$, then the complexity is $O(\sum_{t=1}^{B} |S|t + \sum_{t=1}^{B} c'_{t,opt}) = O(|S|B^2 + \sum_{t=1}^{B} c'_{t,opt})$. For our implementations, in order to get accurate results, we let the size of $|S|$ grow in proportion to n.

Table 1: Comparison of UPAL and BMAL on MNIST data-set of varying training sizes, and with the budget being fixed at 300. The error rate is in percentage, and the time is in seconds.

Sample size	UPAL		BMAL	
	Time	Error	Time	Error
1200	65	7.27	60	5.67
2400	100	6.25	152	6.05
4800	159	6.83	295	6.25
10000	478	5.85	643.17	5.85

In our first set of experiments we fix the budget B to 300, and calculate the test error and the combined training and testing time of both BMAL and UPAL for varying sizes of the training set. All the experiments were performed on the MNIST dataset. Table 1 shows that with increasing sample size UPAL tends to be more efficient than BMAL, though the gain in speed that we observed was at most a factor of 1.8.

In the second set of scalability experiments we fixed the training set size to 10000, and studied the effect of increasing budget. We found out that with increasing budget size the speedup of UPAL over BMAL increases. In particular when the *budget was 2000, UPAL is approximately 7 times faster than BMAL.* All our experiments were run on a dual core machine with 3 GB memory.

Table 2: Comparison of UPAL on the entire MNIST dataset for varying budget size. All the times are in seconds unless stated, and error rates in percentage.

Budget	UPAL		BMAL		Speedup
	Time	Error	Time	Error	
500	859	5.79	1973	5.33	2.3
1000	1919	6.43	7505	5.70	3.9
2000	4676	5.82	32186	5.59	6.9

3.8 Pool Based Active Learning via a Stochastic Gradient Descent Algorithm

In Section 3.3, we introduced the UPAL algorithm, that utilizes unbiased estimate of the risk, via importance weighting, to perform active learning. In UPAL, our

hypothesis class is a large class of unbounded linear hypothesis, i.e. $\mathcal{H} = \mathbb{R}^d$. In this section, we look at the problem of active learning hypothesis of bounded L_2 norm. Our hypothesis class is $\mathcal{H} = \{h \in \mathbb{R}^d : ||h|| \leq R\}$. For this hypothesis class, we introduce a somewhat different active learning algorithm, SGD-AL [3]. SGD-AL operates in the pool based active learning framework that was introduced in Section 3.2. Working in this framework, the model updates are done via a stochastic gradient descent step procedure, on the objective function $\mathbb{E}L(y\langle h, x\rangle)$, followed by a projection onto the L_2 ball. The probability distribution, in each round, is chosen in such a way, that the sum of divergences between consecutive iterates obtained during the stochastic gradient descent procedure stays small. This guarantees that, the excess risk of the hypothesis, outputted by the algorithm at the end of T rounds, w.r.t. any hypothesis in $\mathcal{H}$ is no worser than a passive learning algorithm, that performs one pass of stochastic gradient descent over T labeled examples. The updates of SGD-AL are given by the formula

$$h_{t+1} \leftarrow \Pi_R\Big(h_t - \eta \sum_{i=1}^{n} \frac{Q_i^t}{p_i^t} L'(y_i\langle h_t, x_i\rangle) y_i x_i\Big),$$

where $\Pi_R(v)$ is the projection of vector v onto an origin centered L_2 ball of radius R.

3.9 Excess risk bounds for SGD-AL

We now establish an excess risk guarantee for the hypothesis outputted by SGD-AL, after T rounds.

Theorem 8. *Suppose $|L'(z)| \leq G_{max} < \infty$, for all $z \in [-X, X]$. Let $0 \leq p_{\min} < \frac{1}{n}$ be chosen such that, for all $i = 1, \ldots, n, t = 1, \ldots, T, p_i^t > p_{\min}$. If we run Algorithm 3 with $\eta = \frac{2R}{nXG_{max}}\sqrt{\frac{(1-np_{\min})}{T}}$, and for h_A returned by SGD-AL after T rounds, for any*

[3] SGD-AL stands for Stochastic Gradient Descent based Active Learning

Algorithm 3 SGD-AL Input: $\mathcal{P} = \{x_1, \ldots, x_n\}$, Loss function $L(\cdot)$, Rounds of algorithm T, Labeling Oracle $\mathcal{O}$, $p_{\min} \geq 0, \eta > 0$.

1: Set $h_1 = 0$.
2: **for** $t = 1 : T$ **do**
3: **for** $x_i \in \mathcal{P}$ **do**
4:

$$\bar{y}_i = \begin{cases} y_i \text{ if } x_i \text{ was queried in any one of the rounds } 1, \ldots, t-1 \\ \arg \max_{y \in \{-1,+1\}} |L'(y\langle h_t, x_i \rangle)| \text{ otherwise} \end{cases} \tag{39}$$

5: **end for**
6: Assign probability distribution: $p_i^t = p_{\min} + \frac{(1-np_{\min})|L'(\bar{y}_i\langle h_t, x_i\rangle)|}{\sum_{j=1}^n |L'(\bar{y}_j\langle h_t, x_j\rangle)|}$, and sample a point, say x, from the distribution p^t.
7: **if** x was not queried in the past **then**
8: Query $\mathcal{O}$ for the label y of x.
9: **else**
10: Reuse the label of x.
11: **end if**
12: $g_t \leftarrow \sum_{i=1}^n \frac{Q_i^t}{p_i^t} L'(y_i\langle h_t, x_i\rangle) y_i x_i$
13: $h_{t+1} \leftarrow \Pi_R(h_t - \eta g_t)$.
14: **end for**
15: Return $h_A = \frac{h_1 + \ldots + h_T}{T}$.

$h \in \mathcal{H}$, *we have,*

$$\mathbb{E}L(yh_A(x)) \leq \mathbb{E}L(y\langle h, x\rangle) + 2XRG_{max}\sqrt{\frac{1}{T(1-np_{\min})}} + \frac{2XRG_{max}\sqrt{2}}{\sqrt{n}} + \frac{3}{\sqrt{n}}.$$

Notice that SGD-AL in step 6, puts a probability distribution on the pool in proportion to the absolute value of the derivative of the loss. If $p_{\min} = 0$, then if for a certain point $x_i \in \mathcal{P}$, $L'(\bar{y}_i\langle h_t, x_i\rangle) = 0$, then $p_i^t = 0$. This is problematic, because our theorem uses the unbiased property of certain importance weighted estimators (see lemma 5), and this unbiased property is not satisfied if $p_i^t = 0$. To get around this situation we use $p_{\min} > 0$, so that any point in the pool is queried with a probability of at least $p_{\min}$. However, for certain losses such as logistic loss, and exponential loss, $|L'(\bar{y}_i\langle h_t, x_i\rangle|$ is always greater than 0. For such losses we can take $p_{\min} = 0$. In turn, the impact of $p_{\min}$ is minimal in Theorem 8. One could choose an extremely small

$p_{\min}$, so that $\frac{1}{1-np_{\min}} \approx 1$. We need the following piece of notation.

$$\Delta_{n,\mathcal{H}} \stackrel{\text{def}}{=} \mathbb{E} \sup_{h \in \mathcal{H}} |\mathbb{E}L(y\langle h, x\rangle) - \frac{1}{n}\sum_{i=1}^{n} L(y_i\langle h, x_i\rangle)|$$

Lemma 5. *For h_A returned by SGD-AL after T rounds, and for any $h \in \mathcal{H}$, we have*

$$\begin{aligned}\mathbb{E}L(yh_A(x)) - \mathbb{E}L(y\langle h, x\rangle) &\leq \frac{1}{nT}\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbb{E}\frac{Q_i^t}{p_i^t}\Big(L(y_i\langle h_t, x_i\rangle) - L(y_i\langle h, x_i\rangle)\Big) + \Delta_{n,\mathcal{H}} \\ &\leq \frac{1}{nT}\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbb{E}\frac{Q_i^t}{p_i^t}\langle L'(y_i\langle h_t, x_i\rangle)y_i x_i, h_t - h\rangle + \Delta_{n,\mathcal{H}},\end{aligned}$$

where the expectation is w.r.t. all the random variables.

Proof. Since $\mathcal{H}$ is convex, and $h_1, \ldots, h_T \in \mathcal{H}$, hence, $h_A = \frac{h_1+\ldots+h_T}{T}$ also belongs to $\mathcal{H}$. Hence, our algorithm is a proper learning algorithm. Because of the convexity of $|\cdot|$ function, we get

$$\begin{aligned}|\mathbb{E}L(y\langle h_t, x\rangle) - \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}L(y_i\langle h_t, x_i\rangle)| &\leq \mathbb{E}|\mathbb{E}L(y\langle h_t, x\rangle) - \frac{1}{n}\sum_{i=1}^{n} L(y_i\langle h_t, x_i\rangle| \\ &\leq \mathbb{E}\sup_{h\in\mathcal{H}}|\mathbb{E}L(y\langle h, x\rangle) - \frac{1}{n}\sum_{i=1}^{n} L(y_i\langle h, x_i\rangle)| \\ &\stackrel{\text{def}}{=} \Delta_{n,\mathcal{H}}. \end{aligned} \tag{40}$$

With this we can bound $\mathbb{E}L(yh_A(x))$ as

$$\begin{aligned}\mathbb{E}L(yh_A(x)) &\stackrel{\text{(a)}}{\leq} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}L(y\langle h_t, x\rangle) \\ &\stackrel{\text{(b)}}{\leq} \frac{1}{nT}\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbb{E}L(y_i\langle h_t, x_i\rangle) + \Delta_{n,\mathcal{H}} \\ &\stackrel{\text{(c)}}{=} \frac{1}{nT}\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbb{E}\frac{Q_i^t}{p_i^t}L(y_i\langle h_t, x_i\rangle) + \Delta_{n,\mathcal{H}}. \end{aligned} \tag{41}$$

Inequality (a) follows from Jensen's inequality, inequality (b) follows from equation 40, and equality (c) follows from the fact that $\mathbb{E}_t\frac{Q_i^t}{p_i^t} = 1$.

For any fixed hypothesis h in $\mathcal{H}$, if $p_i^t > 0$, then by our Assumption 1, we have the following unbiasedness property

$$\mathbb{E}L(y\langle h, x\rangle) = \frac{1}{nT}\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbb{E}\frac{Q_i^t}{p_i^t}L(y_i\langle h, x_i\rangle). \tag{42}$$

Putting together Equations 41, 42 we get

$$\mathbb{E}L(yh_A(x)) - \mathbb{E}L(y\langle h, x\rangle) \leq \frac{1}{nT}\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbb{E}\frac{Q_i^t}{p_i^t}\Big(L(y_i\langle h_t, x_i\rangle) - L(y_i\langle h, x_i\rangle)\Big) + \Delta_{n,\mathcal{H}}.$$

With this we obtain the first inequality in the statement of lemma 5. To obtain the second inequality, we simply use the fact that our loss function L is convex, and hence the subgradient at a point is an under-estimate of the function. This gets us,

$$\sum_{i=1}^{n}\frac{Q_i^t}{p_i^t}\Big(L(y_i\langle h, x_i\rangle) - L(y_i\langle h_t, x_i\rangle)\Big) \leq \sum_{i=1}^{n}\frac{Q_i^t}{p_i^t}\langle L'(y_i\langle h_t, x_i\rangle)y_i x_i, h_t - h\rangle. \qquad \square$$

Lemma 5 bounds the expected excess risk as a sum of two terms. The rest of the proof for Theorem 8 requires us to upper bound each of these terms individually. Bounding $\Delta_{n,\mathcal{H}}$ is fairly straightforward and utilizes standard techniques from empirical process theory, such as symmetrization and Talagrand's contraction lemma.

Lemma 6.

$$\Delta_{n,\mathcal{H}} \leq \frac{2}{\sqrt{n}} + \frac{2XRG_{max}\sqrt{2}}{\sqrt{n}}.$$

Proof. Let $\tilde{L}(\cdot) = L(\cdot) - 1$. Note that $\tilde{L}(0) = 0$, and $\tilde{L}$ is Lipschitz with Lipschitz constant $G_{\max}$. Let $\epsilon_1, \ldots, \epsilon_n$ be i.i.d. Rademacher random variables independent of

our data, and sampling scheme. We have,

$$\begin{aligned}
\Delta_{n\mathcal{H}} &\stackrel{\text{def}}{=} \mathbb{E}\sup_{h\in\mathcal{H}}|\mathbb{E}L(y\langle h,x\rangle) - \frac{1}{n}\sum_{i=1}^{n} L(y_i\langle h,x_i\rangle)| \\
&\stackrel{\text{(a)}}{\leq} \frac{2}{n}\mathbb{E}\sup_{h\in\mathcal{H}}|\sum_{i=1}^{n}\epsilon_i L(y_i\langle h,x_i\rangle)| \\
&= \frac{2}{n}\mathbb{E}\sup_{h\in\mathcal{H}}|\sum_{i=1}^{n}\epsilon_i \tilde{L}(y_i\langle h,x_i\rangle) + \epsilon_i| \\
&\stackrel{\text{(b)}}{\leq} \frac{2}{n}\mathbb{E}\sup_{h\in\mathcal{H}}|\sum_{i=1}^{n}\epsilon_i \tilde{L}(y_i\langle h,x_i\rangle)| + \frac{2}{n}\mathbb{E}\sum_{i=1}^{n}|\epsilon_i| \\
&\stackrel{\text{(c)}}{\leq} \frac{2}{n}\mathbb{E}\sup_{h\in\mathcal{H}}|\sum_{i=1}^{n}\epsilon_i \tilde{L}(y_i\langle h,x_i\rangle)| + \frac{2}{n}\sqrt{E(\sum_{i=1}^{n}\epsilon_i)^2} \\
&\stackrel{\text{(d)}}{=} \frac{2}{n}\mathbb{E}\sup_{h\in\mathcal{H}}|\sum_{i=1}^{n}\epsilon_i \tilde{L}(y_i\langle h,x_i\rangle)| + \frac{2}{\sqrt{n}} \\
&\stackrel{\text{(e)}}{\leq} \frac{2G_{\max}}{n}\mathbb{E}\sup_{h\in\mathcal{H}}|\sum_{i=1}^{n}\epsilon_i y_i\langle h,x_i\rangle| + \frac{2}{\sqrt{n}} \\
&\stackrel{\text{(f)}}{\leq} \frac{2XRG_{\max}\sqrt{2}}{\sqrt{n}} + \frac{2}{\sqrt{n}}
\end{aligned}$$

where inequality (a) follows from standard symmetrization arguments, inequality (b) follows from triangle inequality, inequality (c) follows from the simple fact that for any random variable X, $\mathbb{E}|X| \leq \sqrt{\mathbb{E}X^2}$. Equality (d) follows from the fact that $\mathbb{E}\epsilon_i^2 = 1, \mathbb{E}\epsilon_i\epsilon_j = 0$, for $i \neq j$. Inequality (e) follows by Talagrand's contraction lemma, and finally inequality (f) uses an upper bound on the Rademacher process indexed by functions in $\mathcal{H}$ (see Theorem 1 in [55]). □

Lemma 7. *For h_A returned by SGD-AL, and for any $h \in \mathcal{H}$, we have*

$$\mathbb{E}L(yh_A(x)) - \mathbb{E}L(y\langle h,x\rangle) \leq \frac{1}{nT\eta}\Big(2R^2 + \frac{\eta^2}{2}\sum_{t=1}^{T}||g_t||^2\Big) + \Delta_{n,\mathcal{H}}.$$

Proof. Since $h_{t+1} = \Pi_R(h_t - \eta g_t)$, hence we get

$$||h_{t+1} - h||^2 \leq ||h_t - \eta g_t - h||^2 = ||h_t - h||^2 + \eta^2||g_t||^2 - 2\eta\langle h_t - h, g_t\rangle. \tag{43}$$

Rearranging the above equation, we get

$$\langle h_t - h, g_t\rangle \leq \frac{1}{2\eta}||h_t - h||^2 - \frac{1}{2\eta}||h_{t+1} - h||^2 + \frac{\eta^2}{2\eta}||g_t||^2. \tag{44}$$

Replacing for the definition of g_t, and summing up over all $t = 1 \ldots, T$, we get

$$\sum_{i=1}^{n}\sum_{t=1}^{T}\langle h_t - h, \frac{Q_i^t}{p_i^t}L'(y_i\langle h_t, x_i\rangle)y_i x_i\rangle \leq \frac{1}{2\eta}||h_1 - h||^2 + \frac{\eta}{2}\sum_{t=1}^{T}||g_t||^2. \tag{45}$$

Using Lemma 5 and Equation 45 we get

$$\begin{aligned}\mathbb{E}L(yh_A(x)) - \mathbb{E}L(y\langle h, x\rangle) &\leq \frac{1}{nT}\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbb{E}\frac{Q_i^t}{p_i^t}\langle L'(y_i\langle h_t, x_i\rangle)y_i x_i, h_t - h\rangle + \Delta_{n,\mathcal{H}} \\ &\leq \frac{1}{n\eta T}\Big(2R^2 + \frac{\eta^2}{2}\mathbb{E}\sum_{t=1}^{T}||g_t||^2\Big) + \Delta_{n,\mathcal{H}}. \qquad \square\end{aligned}$$

Lemma 8. $\sum_{t=1}^{T}\frac{\eta^2}{2}\mathbb{E}||g_t||^2 \leq \frac{X^2\eta^2 G_{max}^2 n^2 T}{2(1-np_{\min})}$.

Proof. By definition of g_t, we get

$$\eta^2\mathbb{E}||g_t||^2 \overset{(a)}{=} \eta^2\mathbb{E}\Big[\sum_{i=1}^{n}||\frac{Q_i^t}{p_i^t}L'(y_i\langle h_t, x_i\rangle)y_i x_i||^2 + \tag{46}$$

$$\cancelto{0}{\sum_{i\neq j}\frac{Q_i^t Q_j^t}{p_i^t p_j^t}L'(y_i\langle h_t, x_i\rangle)L'(y_j\langle h_t, x_j\rangle)\langle x_i, x_j\rangle}\Big] \tag{47}$$

To obtain (a), we used the fact that, in any round t, only one point can be queried, and hence for all $i \neq j$, $Q_i^t Q_j^t = 0$. By substituting the expression of p_i^t as suggested in Algorithm 3, in Equation 47, we get

$$\begin{aligned}\eta^2\mathbb{E}||g_t||^2 &\leq \eta^2\mathbb{E}\sum_{i=1}^{n}\frac{(L'(y_i\langle h_t, x_i\rangle))^2||x_i||^2}{p_{\min} + \frac{(1-np_{\min})|L'(\bar{y}_i\langle h_t, x_i\rangle)|}{\sum_{j=1}^{n}|L'(\bar{y}_j\langle h_t, x_j\rangle)|}} \\ &\leq \eta^2\mathbb{E}\sum_{i=1}^{n}\frac{(L'(y_i\langle h_t, x_i\rangle))^2||x_i||^2}{\frac{(1-np_{\min})|L'(\bar{y}_i\langle h_t, x_i\rangle)|}{\sum_{j=1}^{n}|L'(\bar{y}_j\langle h_t, x_j\rangle)|}} \\ &= \eta^2 X^2\mathbb{E}\sum_{i=1}^{n}\frac{(L'(y_i\langle h_t, x_i\rangle))^2}{(1-np_{\min})|L'(\bar{y}_i\langle h_t, x_i\rangle)|}\sum_{j=1}^{n}|L'(\bar{y}_j\langle h_t, x_j\rangle)| \\ &\overset{(d)}{\leq} \frac{\eta^2 X^2 n^2 G_{\max}^2}{1-np_{\min}}. \qquad \square\end{aligned}$$

Summing up over all $t = 1, 2, \ldots$, and dividing by two, we get the desired result.

3.9.1 Proof of Theorem 8

Proof. From Lemma 7, and Lemma 8 we know that

$$\mathbb{E}L(yh_A(x)) - \mathbb{E}L(y\langle h, x\rangle) \leq \frac{2R^2}{n\eta T} + \frac{\eta n X^2 G_{\max}^2}{2(1-np_{\min})} + \Delta_{n,\mathcal{H}}. \tag{48}$$

Set $\eta = \frac{2R}{nXG_{\max}}\sqrt{\frac{1-np_{\min}}{T}}$, to get

$$\mathbb{E}L(yh_A(x)) - \mathbb{E}L(y\langle h, x\rangle) = \frac{2RXG}{\sqrt{T(1-np_{\min})}} + \Delta_{n,\mathcal{H}}.$$

Replacing for the expression of $\Delta_{n,\mathcal{H}}$ as provided by Lemma 6, we get the required result. □

3.9.2 Comparison to passive learning.

From standard results in statistical learning theory [16], we know that excess risk of a passive, empirical risk minimization (ERM) procedure, that gets to see the labels of all n points, is $\Delta_{n,\mathcal{H}}$, whose order is provided by Lemma 6 to be $O(\frac{2}{\sqrt{n}} + \frac{2XRG_{\max}\sqrt{2}}{\sqrt{n}})$. This basically shows that SGD-AL, when run for $T = n$ rounds, achieves the same excess risk as an ERM. This guarantees that we are at least as good as passive learning.

3.10 Discussion

In this chapter we introduced a general pool based active learning framework that proceeds in rounds, sampling points from the pool, and uses importance weights to update the current hypothesis. We proposed two algorithms in this framework, namely UPAL and SGD-AL. For UPAL we showed good empirical performance and its ability to scale both with higher budget constraints and larger dataset sizes. We analyzed UPAL under certain statistical assumptions, and established both labeled, and unlabeled sample complexity of the algorithm. In the second part of the chapter, we established excess risk guarantees for SGD-AL, and showed that the excess risk is no worser than a passive learning algorithm.

An important open problem is to be able to establish excess risk guarantees for UPAL under much weaker conditions. A potential approach is to exploit the exponential weighted average interpretation of UPAL, as shown in Section 3.3.1. This interpretation allows us to use PAC-Bayes type of inequalities for the hypothesis returned by UPAL. However, we would need some special PAC-Bayesian inequalities in

order to handle the fact that the loss of a hypothesis at a certain point could be potentially infinite, if the norm of the hypothesis is infinite. To our knowledge there has not been any work yet on deriving PAC-Bayesian inequalities for unbounded random variables. Another theoretically interesting question is to calculate how many unique queries are made after T rounds of UPAL. This problem is similar to calculating the number of non-empty bins in the balls-and-bins model commonly used in the field of randomized algorithms, when there are n bins and T balls, with the different points in the pool being the bins, and the process of throwing a ball in each round being equivalent to querying a point in each round. However since each round is, unlike standard balls-and-bins, dependent on the previous round we expect the analysis to be more involved than a standard balls-and-bins analysis. For SGD-AL it would be useful to provide label complexity guarantees.

3.11 Bibliographic Notes

The UPAL algorithm and its analysis was published at AISTATS 2011 [39] under the title of "UPAL: Unbiased Pool Based Active Learning".

CHAPTER IV

ACTIVE LEARNING FROM THE MULTI-ARMED BANDIT LENS

4.1 Introduction

In Chapter 3, we introduced a generic pool based active learning framework, and proposed two learning algorithms, namely UPAL and SGD-AL, in this framework. In this chapter, we build on the proposed framework, and propose an AL algorithm by building a bridge between the multi-armed bandit world and active learning world. By carefully constructing an analogy between active learning (AL) and multi-armed bandits (MAB), we utilize ideas such as lower confidence bounds and barrier type regularization, commonly used in the multi-armed bandit and bandit online optimization community, to design a new AL algorithm called LCB-AL [1].

4.2 Motivation

In Chapter 3, we showed that the UPAL algorithm under certain conditions, learns a hypothesis, which is an exponentially weighted average of the different hypothesis in the hypothesis class. Hence, UPAL algorithm can be seen as an algorithm that learns with expert advice, where the experts are the different hypothesis. **In this chapter we view the problem of active learning from the lens of exploration-exploitation trade-off**. The concept of exploration-exploitation is central to various problems in decision making under uncertainty. This concept is perhaps best illustrated in the problem of multi-armed bandits [13]. The MAB problem is a B round game, where in a generic round t, the player has to pull one among k arms of a

[1] LCB-AL stands for Lower Confidence Bounds based Active Learning

multi-armed bandit. On doing so the player suffers a loss L_t. The player does not get to know the loss he would have suffered if he had pulled a different arm. The goal of the player is to minimize the cumulative loss suffered over B rounds. In each round the player needs to resolve the dilemma of whether to explore an arm which has not been pulled in the past, or whether to exploit the knowledge of the cumulative losses of the arms that have been pulled in the past. We provide a pool based, sequential AL algorithm called LCB-AL, which is motivated by applying algorithmic ideas from the problem of multi-armed bandits to the problem of AL. In order to do so we build a bridge between the MAB problem and AL problem, providing an equivalence between the arms of a MAB problem, and the hypothesis in $\mathcal{H}$, and mitigating the problem of absence of an explicit loss signal in AL. Establishing this analogy is not very straightforward, but once done allows us to readily use tools such as lower confidence bounds [5], and self-concordant barrier type regularization [1, 18] in the design of LCB-AL. To our knowledge, our work is one of the first in trying to use bandit type ideas for active learning, and we strongly believe that one can build extremely practical, yet very simple and scalable algorithms by understanding the interplay between multi-armed bandits and active learning.

4.2.1 Contributions and Outline of this chapter

1. In Section 4.3, we take the first steps towards building an analogy between MAB and AL. This inspires us to use a very successful algorithm from the MAB literature, based on confidence bound, for the problem of AL. In Section 4.3.2 we show how our algorithm overcomes the problem of absence of an explicit loss signal in active learning problems.

2. The lower confidence bound algorithm requires us to build a lower confidence bound on the risk of any hypothesis in the hypothesis class. We accomplish this, in Section 4.4, via Bernstein type inequalities for martingales.

3. In Section 4.7, we compare LCB-AL to other active learning algorithms on various datasets.

Notation. Let $\mathcal{H} = \{h \in \mathbb{R}^d : ||h|| \leq R\}$ be a convex set in $\mathbb{R}^d$. By $h(x)$, we shall mean the inner product $\langle h, x \rangle$. Let $L : \mathbb{R} \to \mathbb{R}_+$ be a continuous, convex loss function of the margin $yh(x)$. We shall assume that our domain $\mathcal{X}$ is bounded, in the sense that, for all x in $\mathcal{X}$, $||x|| \leq X$. This in turn guarantees that, for all h in $\mathcal{H}$, x in $\mathcal{X}$, $L(\cdot)$ is at most some $L_{\max} < \infty$. For example, for squared loss, $L_{\max} = (1+XR)^2$, for logistic loss $L_{\max} = \log(1 + e^{RX})$, for hinge loss $L_{\max} = 1 + XR$, and for exponential loss $L_{\max} = e^{XR}$. Let $R(h) \stackrel{\text{def}}{=} \mathbb{E}L(yh(x))$ be the risk of a hypothesis $h \in \mathcal{H}$.

4.3 Towards an analogy between Multi-armed bandits and Active Learning

In active learning, the goal is to find a hypothesis $h \in \mathcal{H}$ with low risk by using as little labeled data as possible. In other words, we want to quickly estimate the risk of different hypothesis, and discard suboptimal hypothesis. In MAB, the goal of the player is to design a strategy, that minimizes the cumulative loss suffered by the player over the length of the game. If the player knew the arms with the smallest possible cumulative loss, then the optimal strategy would be to pull this arm in each and every round. Hence, in MAB the player wants to quickly detect the (near) optimal arm to pull. Looking from the lens of MAB, it is now natural to think of AL problem as a MAB problem, where the arms of the MAB are the different hypothesis in $\mathcal{H}$. While this is a good connection to start with, there are two issues that still need to be resolved, which we shall state now.

1. In the MAB problem, in each round we pull an arm of the MAB. If the different hypothesis in $\mathcal{H}$, were thought to be equivalent to the different arms of the MAB, then how do we decide which "hypothesis to pull".

2. In the MAB problem, the player gets to see an explicit loss signal at the end of each round. However, in AL there is no such explicit loss signal, instead the feedback that is received is the label of the queried point x. Hence, the next question that arises is how could one use the label information as some kind of a loss signal? The following subsections attempt to resolve these issues.

4.3.1 Which hypothesis to pull?

A very popular approach in MAB to mitigate the exploration-exploitation trade-off is via the use of lower confidence bounds (LCB) [6, 5, 3, 20] [2]. In the LCB approach, at the end of round t, for each arm a in the set of arms, we build a lower confidence bound, $\text{LCB}_t(a)$ for the cumulative loss the player would have suffered, in hindsight, had he pulled arm a for the first t rounds. The choice of arm a_{t+1} to be pulled in the next round, i.e. round $t+1$ is the solution to the optimization problem $a_{t+1} \in \arg\min \text{LCB}_t(a)$. Such lower confidence bounds can be derived via concentration inequalities [6, 4], and are generally expressed as $LCB_t(a) \stackrel{\text{def}}{=} \hat{L}_t(a) - U(\hat{L}_t(a))$, where $\hat{L}_t(a)$ is an estimate of the cumulative loss of arm a, the player would have suffered had he pulled a each time for the first t rounds, and $U(\hat{L}_t(a))$ is some measure of uncertainty (typically variance) of the cumulative loss of a, at the end of round t. The reason behind the success of confidence bounds in the MAB problem can be explained by the fact that $\text{LCB}_t(a)$ captures both the knowledge of the cumulative loss, via $\hat{L}_t(a)$, as well as the uncertainty in this estimate, via $U(\hat{L}_t(a_t))$. By pulling the arm a_{t+1} in round $t+1$ of our MAB algorithm, and by updating our estimate of the cumulative loss of arm a_{t+1}, our updated estimate $\hat{L}_{t+1}(a_{t+1})$ is a better estimator, as $U(\hat{L}_{t+1}(a_{t+1}))$ is potentially smaller than $U(\hat{L}_t(a_{t+1}))$.

One could use a similar technique even in AL. If one had some kind of a LCB

[2] Traditionally in the bandit literature, researchers have used upper confidence bounds. Since, we are dealing with losses and not rewards, it is useful for our purpose, to rename this as lower confidence bounds.

on the risk of each hypothesis, then we could equate pulling a hypothesis as solving the optimization problem $h_{t+1} \in \arg\min_{h \in \mathcal{H}} \mathrm{LCB}_t(h)$, where $\mathrm{LCB}_t(h)$ is the lower confidence bound on the risk of $h \in \mathcal{H}$. An LCB for $R(h)$ can be obtained by utilizing the labeled data gathered over the run of the algorithm. We shall show how to do this in Section 4.4.

4.3.2 Absence of a loss signal in AL

When an arm is pulled in the MAB setting, the player suffers a loss, and this loss is used to update the LCB of the chosen arm. However, in AL there is no such explicit loss signal. One might come up with a proxy loss signal for the active learning problem, which can then be used to update the lower confidence bound of all the hypothesis in $\mathcal{H}$. However, by examining the conceptual role played by the loss signal, we avoid having to come up with a proxy for loss signal. The utility of the loss signal when the arm a_t is pulled in round t of the MAB problem is two folds. Firstly, to update the cumulative loss of a_t, and secondly to decrease the uncertainty in the estimate of the cumulative loss of a_t. In AL, when a certain point is queried for its label, then this label information can be utilized to improve the error estimate of h_t, as well as other hypothesis. Hence, it makes sense to query $\mathcal{O}$ for the label of some point x in $\mathcal{P}$, such that its label information maximally reduces the variance of the estimate of risk of h_t. Hence, by conceptually viewing label information as a mechanism to reduce the variance of the risk estimate of different hypothesis, we have a disciplined way of deciding which points to query. Table 3 summarizes the analogy between AL and MAB.

Table 3: The analogy between MAB and AL that is used as a guiding principle for the design of LCB-AL.

MAB	AL
Arms.	Hypothesis.
Loss signal on pulling an arm helps improve cumulative loss estimates.	Sampling distribution designed to reduce variance of risk estimates of hypothesis.

4.3.3 Rough outline of LCB-AL.

With the ideas developed in the previous section, we now introduce a new pool based active learning algorithm called LCB-AL (see Algorithm 4). LCB-AL is designed in the generic framework that was introduced in Section 3.2. In this generic framework, we update our hypothesis by minimizing the lower confidence bound over the hypothesis space. In order to construct lower confidence bounds on the risk of h, we use importance weighting along with Bernstein type inequalities for martingales. The problem with such importance weighted estimators is that they have very high variance. In order to tackle the high variance of importance weighted estimators, we use self-concordant barrier type regularization [1, 18]. As a result, in each round (see step 14 of algorithm 4) we solve the optimization problem

$$h_{t+1} \in \arg\min_{h \in \mathcal{H}} LCB_t(h) + \mathcal{R}(h),$$

where $\mathcal{R}(h)$ is a self-concordant barrier type regularization of $\mathcal{H}$. For our choice of $\mathcal{H} = \{h \in \mathbb{R}^d : ||h|| \leq R\}$, $\mathcal{R}(h) = -\log(R^2 - ||h||^2)$. Using h_t we induce a sampling distribution over the pool $\mathcal{P}$, at the start of round t (see step 4 of algorithm 4). As discussed in section 4.3.2, the probability distribution is such that it minimizes the (conditional) variance of the estimate of risk of h_t, given the information gathered from the previous $t-1$ rounds. We shall make this step clear in Section 4.5.

4.4 Risk Estimates and Confidence Bounds

We begin with the notation that will be required to develop our confidence bounds. Let p_i^t be the probability of querying x_i in round t, and $Q_i^t \in \{0, 1\}$ be the random variable which takes the value 1, if x_i was queried in round t, and 0 otherwise. Hence $\mathbb{E}[Q_i^t | p_i^t] = 1$. For convenience, we shall denote by $Q_{1:n}^{1:t}$ the collection of random variables $Q_1^1, \ldots Q_1^t, \ldots, Q_n^1, \ldots, Q_n^t$. Let $Z_i^t \stackrel{\text{def}}{=} y_i Q_i^t$. Denote by $x_{1:n}$ the collection of random variables $x_1, \ldots, x_n$. Also let $[x]_+ = \max\{x, 0\}$.

Algorithm 4 LCB-AL Input: $\mathcal{P} = \{x_1, \ldots, x_n\}$, Loss function $L(\cdot)$, Budget B, Labeling Oracle $\mathcal{O}$, $p_{\min} \in (0, \frac{1}{n})$

Set $h_1 = 0, t = 1$.
while num_queried $\leq B$ **do**
  for $x_i \in \mathcal{P}$ **do**

$$\bar{y}_i = \begin{cases} y_i & \text{if } x_i \text{ was queried in one of the previous rounds} \\ \operatorname{sgn}(h_t(x_i)) & \text{otherwise} \end{cases} \tag{49}$$

    $p_i^t \leftarrow p_{\min} + (1 - np_{\min}) \frac{L(\bar{y}_i h_t(x_i))}{\sum_{x_i \in \mathcal{P}} L(\bar{y}_i h_t(x_i))}$.
  end for
  Sample a point (say x) from the probability vector p^t.
  if x was not queried in the past **then**
    Query $\mathcal{O}$ for the label y of x.
    num_queried $\leftarrow$ num_queried $+ 1$
  else
    Reuse the label of x.
  end if
  Solve: $h_{t+1} = \arg\min_{h \in \mathcal{H}} \mathrm{LCB}_t(h) + \lambda_t \mathcal{R}(h)$.
  $t \leftarrow t + 1$
end while
Return h_B.

We shall make the following independence assumption, that we made in Chapter 3 (Assumption 1).

Assumption. *If x_i has not been queried up until the start of round t, then $p_i^t \perp\!\!\!\perp y_i$ given $x_{1:n}, Z_{1:n}^{1:t-1}$.*

4.4.1 Constructing lower confidence bounds

For any hypothesis $h \in \mathcal{H}$, define $\hat{L}_t(h) \stackrel{\text{def}}{=} \frac{1}{nt} \sum_{i=1}^{n} \sum_{\tau=1}^{t} \frac{Q_i^\tau}{p_i^\tau} L(y_i h(x_i))$. $\hat{L}_t(h)$ is an unbiased estimator of the risk of the hypothesis h, as shown in Chapter 3. Let $\mathcal{Q}_t \stackrel{\text{def}}{=} \{x_i \in \mathcal{P} | \sum_{\tau=1}^{t} Q_i^\tau > 0\}$, and $\mathcal{F}_\tau \stackrel{\text{def}}{=} \sigma(x_{1:n}, Z_{1:n}^{1:\tau})$ be the smallest sigma algebra that makes the random variables $x_{1:n}, Z_{1:n}^{1:\tau}$ measurable. Clearly $\mathcal{F}_1 \subset \ldots \subset \mathcal{F}_t$ form a filtration. Utilizing the unbiased estimator $\hat{L}_t(h)$, along with a Bernstein type inequality for martingales allows us to construct lower confidence bounds for $R(h)$. For simplicity we shall assume that $|\mathcal{H}| < \infty$. While this is definitely not true for the

setting where $\mathcal{H}$ is the space of linear hypothesis with bounded L_2 norm, one could in practice, obtain a "good" approximation, $\mathcal{H}'$ from $\mathcal{H}$, by taking $\mathcal{H}'$ to be a very fine grained cover of $\mathcal{H}$. With this assumption, the following theorem establishes a lower confidence bound on the risk of hypotheses in $\mathcal{H}$.

Theorem 9. *Let $|\mathcal{H}| < \infty$. With probability at least $1 - |\mathcal{H}|\delta(2 + T\log(T/e))$, for all $h \in \mathcal{H}, 4 \leq t \leq T$, and $\delta < 1/e$, we have*

$$R(h) \geq \Big[\hat{L}_t(h) - \frac{2}{t}\log(1/\delta)L_{\max}\left(1 + \frac{1}{np_{\min}}\right) - \frac{4}{nt}\sqrt{V_t \log(1/\delta)} - \sqrt{\frac{L_{\max}^2 \log(1/\delta)}{2n}}\Big]_+$$

where,

$$V_t \stackrel{\text{def}}{=} \sum_{\substack{i=1:n \\ \tau=1:t}} \frac{Q_i^\tau}{(p_i^\tau)^2} L^2(y_i h(x_i)) - \Big(\sum_{\mathcal{Q}_t} L(y_i h(x_i))\Big)^2 + \frac{L_{\max}^2 \sqrt{2t\log(1/\delta)(n-1)}}{\sqrt{p_{\min}}}. \quad (50)$$

In order to prove Theorem 9, we need the Azuma-Hoeffding inequality, and Bernstein inequality for martingale difference sequences.

Theorem 10. *[Azuma-Hoeffding inequality] Let $M_1, M_2, \ldots$ be a martingale difference sequence w.r.t a filtration $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots$. If for each $i \geq 1$, $|M_i| \leq c_i$. Then,*

$$\mathbb{P}\Big[|\sum_{i=1}^n M_i| \geq \epsilon\Big] \leq 2\exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^n c_i^2}\right).$$

Theorem 11 (Bernstein inequality [10])**.** *Let $M_1, \ldots, M_t$ be a martingale difference sequence (MDS), w.r.t. the filtration $\mathcal{F}_1 \subset \ldots \subset \mathcal{F}_t$, with $|M_\tau| \leq b$. Let $\mathbb{V}_\tau M_\tau \stackrel{\text{def}}{=} \mathbb{V}(M_\tau|\mathcal{F}_{\tau-1})$, and $\sigma^2 \stackrel{\text{def}}{=} \sum_{\tau=1}^t \mathbb{V}_\tau M_\tau$. Then we have, for any $\delta < 1/e$, and $t \geq 4$, with probability at least $1 - \delta\log(t)$*

$$\sum_{\tau=1}^t M_\tau < 2\max\{2\sigma, b\sqrt{\log(1/\delta)}\}\sqrt{log(1/\delta)}.$$

4.4.2 Proof of Theorem 9

The proof proceeds via application of Theorems 10 and 11 to appropriately defined martingale difference sequences. Let,

$$M_\tau \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^n \frac{Q_i^\tau}{p_i^\tau} L(y_i h(x_i)) - \frac{1}{n}\sum_{i=1}^n L(y_i h(x_i)). \quad (51)$$

Utilizing the independence assumption, it is easy to see that $\mathbb{E}[M_\tau|\mathcal{F}_{\tau-1}] = 0$. Hence $M_1, \ldots, M_t$ form a martingale difference sequence w.r.t. the filtration $\mathcal{F}_1, \ldots, \mathcal{F}_t$, where $\mathcal{F}_\tau \stackrel{\text{def}}{=} \sigma(x_{1:n}, Z_{1:n}^{1:\tau})$, is the smallest sigma algebra that makes random variables $x_{1:n}, Z_{1:n}^{1:\tau}$ measurable. In order to apply Theorem 11, to the above martingale difference sequence, we need estimates for the sum of conditional variances, and the range of $|M_\tau|$. From Equation 51 and the triangle inequality we get

$$\begin{aligned} |M_\tau| &\leq \frac{1}{n}|\sum_{i=1}^{n} \frac{Q_i^\tau}{p_i^\tau} L(y_i h(x_i))| + \frac{1}{n}|\sum_{i=1}^{n} L(y_i h(x_i))| \\ &\leq L_{\max}\left(1 + \frac{1}{np_{\min}}\right). \end{aligned} \tag{52}$$

To bound the sum of conditional variances we proceed as follows. Let,

$$\begin{aligned} \sigma^2 &\stackrel{\text{def}}{=} \sum_{\tau=1}^{t} \mathbb{E}[M_\tau^2|\mathcal{F}_{\tau-1}] \\ &\stackrel{\text{(a)}}{=} \sum_{\tau=1}^{t} \frac{1}{n^2}\mathbb{E}\Big[\frac{Q_i^\tau}{(p_i^\tau)^2} L^2(y_i h(x_i)) + \underbrace{\frac{2}{n}\sum_{i\neq j} \frac{Q_i^\tau Q_j^\tau}{p_i^\tau p_j^\tau} L(y_i h(x_i)) L(y_j h(x_j))}_{=0} \\ &\qquad - \Big(\sum_{i=1}^{n} L(y_i h(x_i))\Big)^2 |\mathcal{F}_{\tau-1}\Big] \\ &= \underbrace{\frac{1}{n^2}\sum_{\tau=1}^{t}\sum_{i=1}^{n} \frac{L^2(y_i h(x_i))}{p_i^\tau}}_{I_1} - \underbrace{\frac{1}{n^2}\Big(\sum_{i=1}^{n} L(y_i h(x_i))\Big)^2}_{I_2}. \end{aligned} \tag{53}$$

A simple lower bound on I_2 is

$$\hat{I}_2 \stackrel{\text{def}}{=} \frac{1}{n^2}\Big(\sum_{\mathcal{Q}_t} L(y_i h(x_i)\Big)^2. \tag{54}$$

We now proceed to construct an upper bound on I_1. Let,

$$\hat{I}_1 \stackrel{\text{def}}{=} \frac{1}{n^2}\sum_{\tau=1}^{t}\sum_{i=1}^{n} \frac{Q_i^\tau}{(p_i^\tau)^2} L^2(y_i h(x_i)).$$

Define

$$J_\tau \stackrel{\text{def}}{=} \frac{1}{n^2}\sum_{i=1}^{n} \frac{Q_i^\tau}{(p_i^\tau)^2} L^2(y_i h(x_i)) - \frac{1}{n^2}\sum_{i=1}^{n} \frac{1}{p_i^\tau} L^2(y_i h(x_i))$$

Once again utilizing our independence assumption, we conclude that $J_1, \dots J_t$ form an MDS w.r.t. the filtration $\mathcal{F}_1, \dots, \mathcal{F}_t$. Applying Theorem 10 to this MDS, we get with probability at least $1-\delta$

$$|\sum_{\tau=1}^{t} J_\tau| \leq \frac{L_{\max}^2 \sqrt{2t\log(1/\delta)}}{n^2}\sqrt{\frac{n-1}{p_{\min}}}. \tag{55}$$

Hence, from equations 54 and 55, we get with probability at least $1-\delta$,

$$\sigma^2 \leq \frac{1}{n^2}\sum_{\substack{i=1:n\\ \tau=1:t}} \frac{Q_i^\tau}{(p_i^\tau)^2} L^2(y_i h(x_i)) - \Big(\sum_{\mathcal{Q}_t} L(y_i h(x_i))\Big)^2 + \frac{L_{\max}^2\sqrt{2t\log(1/\delta)(n-1)}}{\sqrt{p_{\min}}}. \tag{56}$$

For any fixed $h \in \mathcal{H}$, and fixed $t \leq T$, we can apply Theorem 11, to our martingale difference sequence M_t. Finally via Hoeffding inequality we get an upper bound on $|\frac{1}{n}\sum_{i=1}^n L(y_i h(x_i)) - R(h)|$ for a fixed $h \in \mathcal{H}, t \leq T$. Applying the union bound over all hypothesis in $\mathcal{H}$, and $t \leq T$ we get the desired result.

Specification of $\mathrm{LCB}_t(h)$. Theorem 9 provides us with an expression for $\mathrm{LCB}_t(h)$. This allows us to set

$$\mathrm{LCB}_t(h) \stackrel{\text{def}}{=} \Big[\hat{L}_t(h) - \frac{4}{nt}\sqrt{\log(1/\delta')V_t} - \frac{2}{t}\log(1/\delta')L_{\max}\left(1+\frac{1}{np_{\min}}\right) - \sqrt{\frac{L_{\max}^2\log(1/\delta')}{2n}}\Big]_+, \tag{57}$$

where V_t is shown in Equation 50, and $\delta' = \frac{\delta}{|\mathcal{H}|(2+T\log(T/e))}$.

4.5 Query probability distribution in each round of LCB-AL

The only thing that is left to be motivated in LCB-AL, is the choice of probability distribution in steps 3-5. As explained in section 4.3.2, we want to use a sampling distribution, such that the conditional variance of the risk estimate $\hat{L}_t(h_t)$ is minimized. We shall now show how the sampling distribution should be designed in order to achieve this goal. Let $\Delta_n \subset \mathbb{R}_+^n$ be the $n-1$ dimensional probability simplex. Let $V_t(\cdot)$ denote the variance, conditioned on $x_{1:n}, Z_{1:n}^{1:t-1}$. Let $p^t \stackrel{\text{def}}{=} (p_1^t, \dots, p_n^t) \in \Delta_n$. At

the start of round t, the desired sampling distribution, p^t should satisfy

$$p^t = \arg\min_{\tilde{p}^t \in \Delta_n} \mathbb{V}_t \Big[\underbrace{\frac{1}{nt} \sum_{\substack{i=1:n \\ \tau=1:t}} \frac{Q_i^\tau}{\tilde{p}_i^\tau} L(y_i h_t(x_i))}_{\hat{L}_t(h_t)} \Big]$$
$$= \arg\min_{\tilde{p}^t \in \Delta_n} \sum_{i=1}^{n} \frac{\mathbb{E}_t L^2(y_i h_t(x_i))}{\tilde{p}_i^\tau}$$

Solving the above optimization problem yields the simple solution $p_i^t \propto \sqrt{\mathbb{E}_t L^2(y_i h_t(x_i))}$. If $x_i \in \mathcal{Q}_{t-1}$, then the label y_i is known and hence, we let $p_i^t \propto L(y_i h_t(x_i))$. If $x_i \notin \mathcal{Q}_{t-1}$, then since y_i is yet unknown, we let $p_i^t \propto L(|h_t(x_i)|)$. This is equivalent to taking y_i to be equal to $\text{sgn}(h_t(x_i))$ (see steps 3-5 of algorithm 4). This scheme encourages querying points which have small margin w.r.t the current classifier, h_t, or points which have already been queried for their label, but on which the current hypothesis, h_t suffers a large loss. In any round, the minimum probability of querying any point is $p_{\min}$. This guarantees that $\hat{L}_t(h)$ is an unbiased estimator of risk of h.

4.6 Related Work

To our knowledge, there have been few papers bridging the world of active learning and MAB. [8] proposed a meta-active learning algorithm called COMB. COMB was an implementation of the EXP4 algorithm for MAB with expert advice, where the different active learning algorithms are the various "experts" and the different points in the pool are the arms of the MAB. Briefly, in each round, each of the experts suggest a sampling distribution on the pool. COMB maintains an estimate of the error rate of each expert, takes into account the expert suggestions, and uses exponential weighting to come up with a sampling distribution on the pool. In order to estimate the error rate of each of the experts, the authors proposed a proxy reward function of querying a point in terms of the entropy of label distribution of the unlabeled pool, induced by the classifier obtained on the labeled dataset gathered by COMB till the current iteration. In a way, the concept of reward seems inevitable in their formulation

because the unlabeled points in the pool are treated as arms of the MAB. In contrast, we think of the arms of the bandit as the different hypothesis, and querying a data point, as the process of improving our estimate of the risk of the different hypothesis. Hence, we bypass the need for an explicit reward signal, yet utilize MAB ideas for AL.

The problem of active learning in multi-armed bandits was investigated by Carpentier et al. [23]. Here the authors assume that they are in the stochastic multi-armed bandit scenario, and on pulling a certain arm, we get a loss sampled from the loss distribution of the arm. The authors suggest a lower confidence bound algorithm to estimate, in a sample efficient way, uniformly well the average loss of all the arms.

4.7 Experiments

We implemented LCB-AL in MATLAB, and compared it with UPAL, BMAL [48], and a passive learning (PL) algorithm that minimizes the squared norm regularized logistic loss. As mentioned before, our hypothesis set is $\mathcal{H} = \{h \in \mathbb{R}^d : ||h|| \leq R\}$, whose self-concordant barrier is $\mathcal{R}(h) = -\log(R^2 - ||h||^2)$ [1], where $R > 0$ was provided as an input to LCB-AL. For our implementation of LCB-AL, we used a slightly different definition for $\text{LCB}_t(h)$, than the one proposed in equation 57. Let

$$\text{LCB}'_t(h) \stackrel{\text{def}}{=} \hat{L}_t(h) - C_t\sqrt{V'_t}, \tag{58}$$

where

$$V'_t \stackrel{\text{def}}{=} \sum_{\substack{i=1:n \\ \tau=1:t}} \frac{Q_i^\tau}{(p_i^\tau)^2} L^2(y_i h(x_i)) - \Big(\sum_{\mathcal{Q}_t} L(y_i h(x_i))\Big)^2.$$

The definition of $\text{LCB}'_t(h)$ is almost similar to the one suggested by Equation 57 except that the terms $\frac{2}{t}\log(1/\delta')L_{\max}(1+\frac{1}{np_{\min}})$, and $\sqrt{\frac{L^2_{\max}\log(1/\delta')}{2n}}$ in $\text{LCB}_t(h)$ were dropped (as they are independent of h), and the term $\frac{L^2_{\max}\sqrt{2t\log(1/\delta)(n-1)}}{p_{\min}}$, was dropped from V_t, as it is independent of h. Note that $\text{LCB}_t(h)$ is larger than $\text{LCB}'_t(h)$, and hence

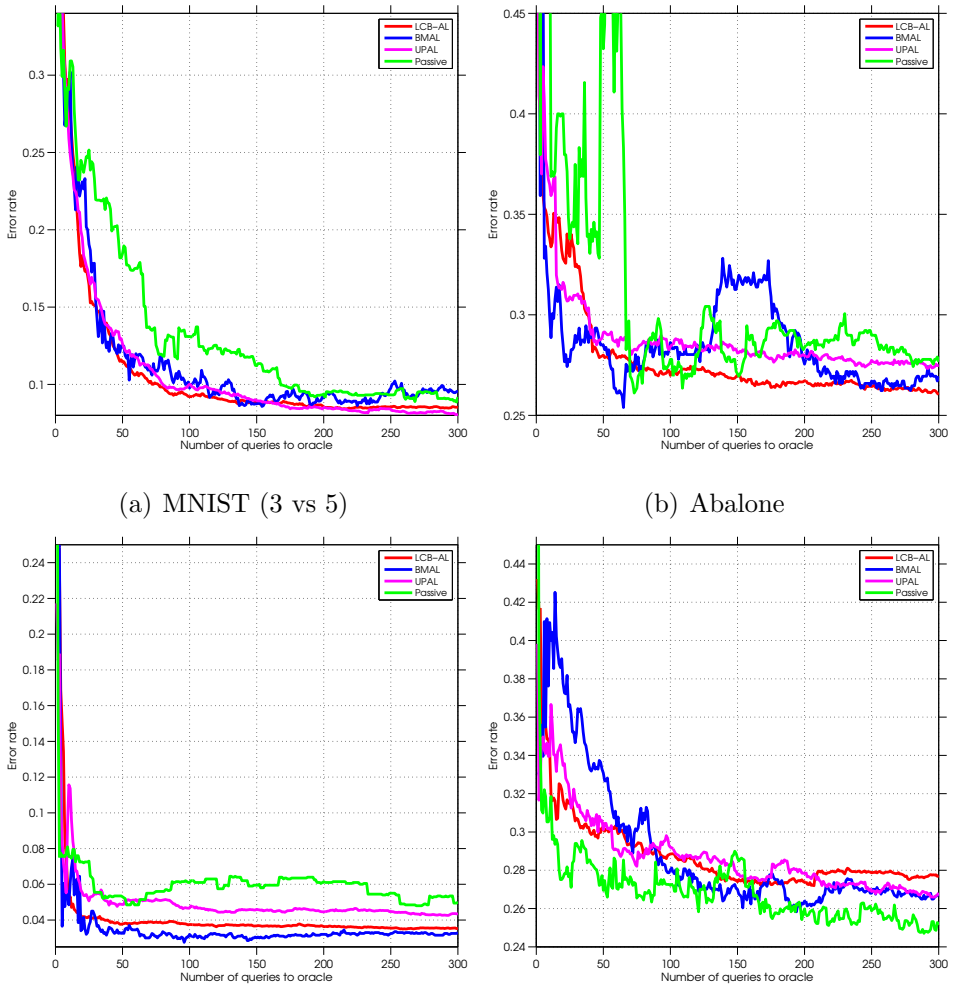
$\text{LCB}'_t(h)$ is a valid lower confidence bound of $R(h)$. Finally, motivated by Sauer-Shelah lemma, we replace $|\mathcal{H}|$ in δ' term by n^d. C_t, and λ_t in all our experiments were set to $\frac{\sqrt{\log(t)}}{10}$, and $\frac{100nt}{\left(\sum_{i=1}^{n}\sum_{\tau=1}^{t-1}\frac{Q_i^\tau}{p_i^\tau}\right)^{1/3}}$ respectively. We used minFunc [3] to solve all of our optimization problems. We used the same datasets that were used in Chapter 3, namely MNIST (3 Vs 5), Statlog, Whitewine, Abalone. Since, UPAL and LCB-AL are randomized algorithms, on each dataset, we ran them 10 times each, and report averaged measurements. For all of our experiments, we used a separate held-out test dataset, to calculate test error. For our passive learner, we provided the learning algorithm with incremental data, trained the learner and report the results on the held out test dataset.

4.7.1 Experimental comparisons of different algorithms

Figure 2 shows the test error of the hypothesis obtained, corresponding to the number of unique queries made to the oracle, by each algorithm. Table 4 shows the error rate on the test set, of each algorithm, once the budget is exhausted. On three of the datasets, namely MNIST, Abalone, and Statlog, utilizing an active learner is better than a passive learner. On MNIST, the performance of LCB-AL and UPAL are nearly equal as far as the final test error goes, and both are better than BMAL. On Abalone, LCB-AL is better than both BMAL and UPAL, while on Statlog the final error achieved by BMAL is better than UPAL, and also LCB-AL, though the difference between LCB-AL and BMAL is pretty narrow. On Whitewine, passive learner is better than any of the active learners. In order to gain an insight into how well each of the learning algorithm learns with each query to the oracle, we also report the cumulative error rate of each algorithm, summed over all the queries in Table 4. Even on this measure, LCB-AL and UPAL are better than BMAL on MNIST, Abalone and Whitewine datasets, with an largest difference being in the case

[3] minFunc can be downloaded from `http://www.di.ens.fr/~mschmidt/Software/minFunc.html`

of Abalone and Statlog.



(a) MNIST (3 vs 5)

(b) Abalone

(c) Statlog

(d) Whitewine

Figure 2: Error rate of different learning algorithms with the number of queries made to the oracle.

4.7.2 Comparing UPAL with LCB-AL

From our first set of experiments it looks like UPAL is just as good as LCB-AL if not any better. e.g. on the MNIST dataset, there is almost no difference between LCB-AL and UPAL. Since, both LCB-AL and UPAL are randomized algorithms it makes sense to measure the fluctuations in the performance of both the algorithms. Table 5 gives the standard deviation of the cumulative error rate over all the runs for both LCB-AL and UPAL. It is clear that the standard deviation of the cumulative

Table 4: Comparison of various active learning algorithms and passive learner on various datasets.In this table we report both the error rate of each learner after it has exhausted its budget, as well as the cumulative error rate for each learning algorithm.

Dataset	LCB-AL		UPAL		BMAL		PL	
MNIST	0.0808	33.27	0.0809	32.75	0.0958	34.89	0.0918	40.08
Abalone	0.2604	83.49	0.2747	86.60	0.2695	86.21	0.2766	93.60
Statlog	0.0354	12.59	0.0433	14.97	0.0330	11.33	0.05	18.06
Whitewine	0.2771	86.30	0.2682	86.21	0.2665	86.95	0.2517	80.94

Table 5: Standard deviation of the cumulative error rate of LCB-AL and UPAL on different datasets.

Dataset	LCB-AL	UPAL
MNIST	3.8604	5.0132
Abalone	2.6512	2.6869
Statlog	0.7944	1.6691
Whitewine	2.4097	2.9992

error rate for LCB-AL is uniformly smaller than that of UPAL over all datasets, and the difference in the standard deviations is largest for the MNIST dataset. This can be explained by the fact that, the unbiased estimator of risk used in UPAL is a high variance estimator, and hence not a reliable estimator of the risk of a hypothesis. In LCB-AL, by utilizing lower confidence bounds, and a self-concordant barrier type regularizer, we are able to tackle the high variance of our estimator, and at the same time harness the variance for exploration in the hypothesis space. In fact, a similar phenomenon occurs even in the MAB setting, where algorithms built only on unbiased estimators, such as EXP3 [6], achieve optimal performance only on an average, whereas algorithm using confidence bounds such as EXP3.P achieve optimal performance with high probability.

4.8 Conclusions and Discussion

We proposed LCB-AL a multi-armed bandit inspired pool based active learning algorithm. By viewing the problem of active learning as quickly detecting the hypothesis with (near) optimal risk, we view the problem of active learning as similar to a MAB

problem with the arms being the different hypothesis. By building lower confidence bounds on the risk of each hypothesis we are able to perform exploration in the hypothesis space. By conceptually investigating the role of a loss signal in MAB, we are able to design a sampling distribution from which we sample the points to be queried. Experimental results suggest that our algorithm is both more accurate, and also more stable than competing active learning algorithms.

An important property of LCB-AL that is worth investigating is that what is the excess risk of the hypothesis returned by LCB-AL as a function of n, B? This would also imply an upper bound on the budget, B that would be required in order to guarantee an excess risk of ϵ. The hard part of this analysis is the fact that, because of the use of lower confidence bounds, the resulting optimization problem that we solve at each stage of LCB-AL, is a non-convex optimization problem. One could relax the non-convex expression for $\text{LCB}_t(h)$ as provided by Equation 57, to another non-convex expression, but one which is a difference of convex functions, as follows

$$\text{LCB}_t^{''}(h) = \hat{L}_t(h) - \frac{4}{nt}\sqrt{\log(1/\delta')} \sum_{\substack{i=1:n \\ \tau=1:t}} \frac{Q_i^\tau}{(p_i^\tau)} L(y_i h(x_i)) - \frac{4}{nt}\log(1/\delta')\Big(\sum_{\mathcal{Q}_t} L(y_i h(x_i))\Big),$$

where $\delta' = \frac{\delta}{|\mathcal{H}|(2+T\log(T/e))}$. Using the fact that $z_+ \geq z$, and $\sqrt{\cdot}$ is sub-additive, we can easily show that $\text{LCB}_t(h) \geq \text{LCB}_t^{''}(h)$, and hence $\text{LCB}_t^{''}(h)$ is a valid lower confidence bound on $R(h)$, with probability at least $1 - \delta'$. As mentioned, a nice property of $\text{LCB}_t^{''}(h)$ is that it is a difference of convex functions, and hence efficient algorithms exist [62] that guarantee convergence to a critical point.

Another extension of this work could be to investigate how different concentration inequalities can be utilized to give different lower confidence bounds for the risk of a hypothesis. This has proved to be an attractive idea in the MAB setting and it is generally accepted that tighter concentration inequalities lead to better algorithms for MAB [3, 80]. We expect something similar to happen even in active learning problems.

4.9 Bibliographic Notes

This chapter was published at UAI 2013 [40], under the title of "Building Bridges: Viewing Active Learning from the Multi-Armed Bandit Lens".

CHAPTER V

ACTIVE MODEL AGGREGATION VIA STOCHASTIC MIRROR DESCENT

5.1 Introduction

In Chapters 3 and 4, we primarily looked at active learning in the setting when we are given a class of models $\mathcal{H}$, and we are required to return one model from this class. In this chapter, we consider active learning in the ensemble framework, where we are given a collection of models, and we want to actively learn a model, which does not necessarily belong to the model class. To make this precise, we assume that we are given a collection of models, $\mathcal{B} = \{b_1, \ldots, b_M\}$, and we want to learn a model in the convex hull of $\mathcal{B}$. Learning in the convex hull of models has been of interest both in machine learning and approximation theory. Ensemble methods [33, 81] are methods that combine a large number of simple models to learn a single powerful model. Boosting algorithms such as AdaBoost [35], and LogitBoost [38] can be viewed as performing aggregation via functional gradient descent [67]. The key idea here is minimization of a convex loss, exponential loss in the case of AdaBoost, and logistic loss in the case of LogitBoost, via a sequential aggregation of models in $\mathcal{F}$. In the case of boosting, the set of base models are weak learners, and by aggregating the models we aim to boost the learning capabilities of the final aggregated model. The problem of model aggregation for regression models was first proposed by Nemirovski et al. [71]. Given a collection of models, $\mathcal{B} = \{b_1, \ldots, b_M\}$, Nemirovski [71] outlined three problems of model aggregation, namely, model selection, convex aggregation and linear aggregation. **In this chapter we are interested in actively learning a convex aggregation of models for the binary classification problem.** Given

a convex, margin based loss function $L : \mathbb{R} \to \mathbb{R}_+$, we want a procedure that outputs a model, $f(x) = \sum_{j=1}^{M} \beta_j b_j(x)$ in the convex hull of $\mathcal{B}$, whose excess risk, when compared to the best model in the convex hull of $\mathcal{B}$ satisfies the inequality

$$\mathbb{E}L(y\langle \beta, b(x)\rangle) \leq \min_{\theta \in \Delta_M} \mathbb{E}L(y\langle \theta, b(x)\rangle) + \delta_{T,M},$$

where $\langle , \rangle$ is used to denote the dot product, $b(x) \stackrel{\text{def}}{=} [b_1(x), \ldots, b_M(x)] \in \{-1, +1\}^M$, and $\delta_{T,M} > 0$ is a small remainder term that goes to 0 as $T \to \infty$, and the expectation is w.r.t. all the random variables involved. In order to construct such a β vector we assume that we have access to an unlabeled stream of examples $x_1, x_2, \ldots$, drawn i.i.d. from the underlying distribution P defined on $\mathcal{X}$.

Juditsky et al. [54] studied the above problem of learning the best convex aggregation of models, under the assumption that one has access to a stream of labeled examples sampled i.i.d. from the underlying distribution. They introduced an online, stochastic mirror descent algorithm for the problem of learning the best convex aggregation of models. We shall call their method SMD-PMA [1]. They showed that by making one pass of the stochastic mirror descent algorithm, and by averaging the iterates obtained after each step of the algorithm, the resulting convex aggregate has excess risk of $O\left(\sqrt{\frac{\log(M)}{T}}\right)$, where M is the number of models being aggregated, and T is the number of samples seen in the stream. Essentially, SMD-PMA is a slight modification of the stochastic mirror descent algorithm, applied to the stochastic optimization problem

$$\min_{\theta \in \Delta_M} \mathbb{E}L(y\langle \theta, b(x)\rangle).$$

Since the constraint set is a simplex, the entropy regularizer was used in SMD-PMA. The stochastic mirror descent procedure is followed by an averaging step, that allows the authors to obtain excess risk bounds.

[1] SMD-PMA stands for Passive Model Aggregation via Stochastic Mirror Descent

5.1.1 Contributions.

In this chapter, we are interested in learning convex aggregation of models, with the help of actively labeled data. We consider a streaming setting, where we are given unlabeled points $x_1, x_2, \ldots$ and an oracle $\mathcal{O}$. The oracle $\mathcal{O}$, when provided as an input x in $\mathcal{P}$, returns a label $y \in \{-1, +1\} \sim \mathbb{P}[Y|X = x]$. We present an algorithm which is essentially a one-pass, stochastic mirror-descent based active learning algorithm, called SMD-AMA [2], which solves the stochastic optimization problem $\min_{\theta \in \Delta_M} \mathbb{E}_{x,y}[L(y\langle \theta, b(x)\rangle)]$. Since we are dealing with simplex constraints, we use the entropy function as the regularization function in our stochastic mirror descent algorithm. In round t of SMD-AMA, we query for the label of point x_t, with probability p_t. This allows us to construct an unbiased stochastic sub-gradient of the objective function at the current iterate. If the length of the stream is T points, then SMD-AMA returns the hypothesis $\hat{b}_T \stackrel{\text{def}}{=} \langle \hat{\theta}_T, b\rangle$, where $b : \mathcal{X} \to \{-1, +1\}$ is defined as $b(x) = [b_1(x), \ldots, b_M(x)]^T$. We show that the excess risk of the hypothesis $\hat{b}_T$, w.r.t. the best convex aggregate of models, scales with the number of models as $\sqrt{\log(M)}$, and decays with the number of points, T, as $\frac{1}{T^{1-\kappa}}$, where $\kappa > 1/2$ is an algorithmic parameter. The mild dependence on the number of models, M, allows us to use a large number of models, which is desirable when we are learning convex aggregation of models. This chapter is organized as follows. In Section 5.2, we introduce our algorithm SMD-AMA. In Section 5.3 we present an excess risk bound for the hypothesis returned by our active learning algorithms. Section 5.4 reviews related work, and Section 5.5 compares our proposed algorithm to a passive learning algorithm, and a previously proposed ensemble based active learning algorithm.

[2] Stochastic Mirror Descent Based Active Model Aggregation

5.2 Algorithm Design

Before we can jump into the details of SMD-AMA, we shall build some ground work leading to the algorithm.

5.2.1 Preliminaries

SMD-AMA is essentially a slight variant of SMD algorithm. In order to fully specify a SMD algorithm, we need to know a way to calculate unbiased estimates of the gradient to the stochastic objective function, and we need to specify a Legendre, barrier regularization function corresponding to the constraint set. As we mentioned before our stochastic optimization problem is

$$\min_{\theta \in \Delta_M} \mathbb{E}L(\langle \theta, b(x) \rangle). \tag{59}$$

Standard analysis of stochastic mirror descent algorithm, assumes that we have access to a stochastic (sub)gradient oracle which provides an unbiased estimate of the gradient of the objective function at any point in the domain. A naive application of the stochastic mirror descent method to our optimization problem 59 would require, in each iteration, to obtain a stochastic subgradient of $\mathbb{E}L(y\langle \theta, b(x) \rangle)$. In iteration t, if the current iterate is θ_{t-1}, then a stochastic subgradient of $f(\theta)$ at $\theta = \theta_{t-1}$ is given by

$$\nabla f(\theta_{t-1}) = L'(y_t \langle \theta_{t-1}, b(x_t) \rangle) y_t b(x_t), \tag{60}$$

where $L'(\cdot)$ is the subderivative of L at the given argument. If in round t, we decided to query for the label of the point x_t, then one can calculate the stochastic subgradient using Equation 60. However, if we decided not to query for the label of x_t, then the stochastic subgradient, which depends on the unknown label y_t of x_t, cannot be calculated. While one could, in such a case, consider the stochastic subgradient to be the zero vector, this is no longer an unbiased estimate of the subgradient. This is problematic, as the classical analysis of stochastic mirror descent, assumes that one

has access to unbiased estimates of the subgradient of the objective function. In order to counter this problem we use the idea of importance sampling.

Importance weighted subgradient estimates: In order to use importance weights, we assume that in round t, a point x_t is queried with probability p_t. Suppose Q_t is a $\{0, 1\}$ random variable, which takes the value 1, if x_t was queried, and takes the value 0 if x_t was not queried. Let $Z_t \stackrel{\text{def}}{=} y_t Q_t$. Let $Z_{1:t-1}$ be the collection of random variables $Z_1, \ldots, Z_{t-1}$. We shall make the following independence assumption, similar to Assumption 1 first presented in Chapter 3.

Assumption 5. $p_t \perp\!\!\!\perp y_t | x_t, x_{1:t-1}, Z_{1:t-1}$,

Consider the following importance-weighted stochastic subgradient

$$g_t \stackrel{\text{def}}{=} \frac{Q_t}{p_t} L'(y_t \langle \theta_{t-1}, b(x_t) \rangle) y_t b(x_t).$$

We have the following fairly simple proposition.

Proposition 2. *Under Assumption 5 we have,*

$$\mathbb{E}_{x_t, y_t}[g_t | x_{1:t-1}, Z_{1:t-1}] = \nabla \mathbb{E}_{x,y}[L(y \langle \theta_{t-1}, b(x) \rangle) | x_{1:t-1}, Z_{1:t-1}] \tag{61}$$

Proof.

$$\begin{aligned}
\mathbb{E}_{x_t, y_t}[g_t | x_{1:t-1}, Z_{1:t-1}] &= \mathbb{E}_{x_t, y_t}\left[\frac{Q_t}{p_t} L'(y_t \langle \theta_{t-1}, b(x_t) \rangle) y_t b(x_t) | x_{1:t-1}, Z_{1:t-1}\right] \\
&= \mathbb{E}_{x_t} \mathbb{E}_{Q_t, y_t}\Big[\frac{Q_t}{p_t} L'(y_t \langle \theta_{t-1}, b(x_t) \rangle) y_t b(x_t) | x_{1:t-1}, Z_{1:t-1}, x_t\Big] \\
&\stackrel{\text{(a)}}{=} \mathbb{E} L'(y_t \langle \theta_{t-1}, b(x_t) \rangle) y_t b(x_t) \\
&\stackrel{\text{(b)}}{=} \nabla \mathbb{E}_{x,y}[L(y \langle \theta_{t-1}, b(x) \rangle) | x_{1:t-1}, Z_{1:t-1}].
\end{aligned}$$

□

In Equality (a) we used our Assumption 5, and the fact that $\mathbb{E}_{Q_t | x_{1:t-1}, Z_{1:t-1}, x_t} \frac{Q_t}{p_t} = 1$. In Equality (b) we used the fact that our data is i.i.d., and hence x_t, y_t is independent of $x_{1:t-1}, y_{1:t-1}$. Proposition 2 says that g_t provided by Equation 5.2.1 provides an unbiased estimate of the gradient of the objective function in Equation 59.

Before we dive into the details of SMD-AMA we shall need a few notations, and terminology, that are relevant for the explanation of the our algorithm. Most of this exposition is standard and is taken from [54]. Let $E = \ell_1^M$, be the space of $\mathbb{R}^M$ equipped with ℓ_1 norm. Let $E^* = \ell_\infty^M$ be the corresponding dual space, equipped with the ℓ_∞ norm.

Definition 4. *Let $\Delta_M \subset E$, be the simplex, and let $V : \Delta_M \to \mathbb{R}$ be a convex function. For a given parameter β, the β conjugate dual of V is the convex function $V_\beta^* : E^* \to \mathbb{R}$, defined as*

$$V_\beta^*(\xi) = \sup_{\theta \in \Delta_M} [\langle \xi, \theta \rangle - \beta V(\theta)].$$

This definition is the same as the definition of conjugate function that was first stated in Chapter 2, except with an additional parameter β. The role of β will be clarified in Section 5.2.3. Finally, in order to fully specify a SMD algorithm, we need a regularization function. Since our constraint set is a simplex, we shall use the entropy function defined as

$$\mathcal{R}(\theta) = \begin{cases} -\sum_{j=1}^M \theta_j \log(\theta_j) \text{ if } \theta \in \Delta_M \\ \infty \text{ otherwise} \end{cases}$$

as our regularization function.

5.2.2 Design of SMD-AMA.

We now have all the ingredients of our algorithm in place. The algorithm proceeds in a streaming fashion, looking at one unlabeled data point at a time. Step 4 of SMD-AMA, calculates the probability of the point being labeled +1 by the current convex aggregate, and this calculation is used in Step 5 to calculate the probability of querying the label of x_t. Notice that the probability of querying a point, in round t, is always at least $\epsilon_t > 0$. Value of ϵ_t is set in Step 3. Step 7 calculates the importance weighted gradient, which is used in Step 9 to calculate the new iterate

θ_t. By straightforward Calculus, one can show that Step 9, leads to the following iteration

$$\theta_{t,j} \propto \exp(-\xi_{t,j}/\beta),$$

where $\xi_{t,j}$ is the j^{th} component of the vector ξ_t. In Step 4, we use properties of the loss function in order to estimate p_t^+. It is well known that standard loss functions such as exponential, logistic, squared loss, modified squared loss, Huber loss, which are used in classification, are also proper losses for probability estimation [96, 21, 78]. Hence, given the loss function, via standard formulae it is easy to estimate the conditional probability $\mathbb{P}[Y_t = 1|X_t = 1]$. For instance, if one were to use the squared loss $L(yz) = (1 - yz)^2$, then our estimate for p_t^+ is given by the formula $\max\left(0, \min\left(\frac{1+z}{2}, 1\right)\right)$. Similarly for other losses, estimates for the probability of label being $+1$ can be calculated (e.g see Section 3 in [96]). In our case, the value of z, in round t, is given by $\langle \theta_{t-1}, b(x_t)\rangle$, and hence if we were to use the squared loss, then $p_t^+ = \max\left(0, \min\left(\frac{1+\langle \theta_{t-1}, b(x_t)\rangle}{2}, 1\right)\right)$.

Algorithm 5 SMD-AMA (Input: A margin based loss function L, Labeling Oracle $\mathcal{O}$, Parameters $1 \geq \kappa > \frac{1}{2}$, $\beta_0 > 0$)

1. Initialize $\theta_0 = [\frac{1}{M}, \ldots, \frac{1}{M}]^T, \xi_0 = [0, \ldots, 0], t = 1$

for t=1,... **do**

2. Receive x_t.
3. Set $\epsilon_t = t^{1-2\kappa}$
4. Estimate $p_t^+ = \mathbb{P}[Y_t = 1|X = x_t, \theta_{t-1}]$
5. Calculate $p_t = 4p_t^+(1 - p_t^+)(1 - \epsilon_t) + \epsilon_t$.
6. Query the label of x_t with probability p_t.
7. Set $g_t = \frac{Q_t}{p_t} L'(y_t \langle \theta_{t-1}, b(x_t)\rangle) y_t b(x_t)$.
8. Set $\xi_t \leftarrow \xi_{t-1} - g_t$.
8. Update $\beta_t = \beta_0(t + 1)^\kappa$.
9. Calculate $\theta_t = \nabla \mathcal{R}^*_{\beta_t}(\xi_t)$.
10 $t \leftarrow t + 1$.

end for

11. Return $\hat{\theta}_T = \frac{\sum_{t=1}^T \theta_t}{T}$.

SMD-AMA, like stochastic mirror descent, performs gradient descent in the dual space, and at each round generates the primal variable via the β conjugate of $\mathcal{R}$. If

p_t is set to 1, then we recover the SMD-PMA algorithm of Juditsky et al.

5.2.3 Difference Between Stochastic Mirror Descent and SMD-PMA

The primary difference between SMD-PMA and stochastic mirror descent is in the way the primal variables are generated. While in SMD-PMA β_t changes with t, in stochastic mirror descent, $\beta_0 = \beta_1 = \ldots = \beta_t \ldots$ is fixed to a constant that depends on the length of the data stream. As we saw in Chapter 2, in the standard implementation of stochastic mirror descent, β parameter is not used. Instead a step size η is used. In contrast, SMD-AMA, never uses a step size, but instead uses a β parameter, that roughly behaves as inverse of step size. The advantage of using a variable β_t, is that we are able to obtain excess risk guarantees without even knowing the length of the data stream.

5.3 Excess Risk analysis

Theorem 12. *Let $\mathcal{B} = \{b_1, \ldots, b_M\}$ be a collection of basis models, where for each $x \in \mathcal{X}$, $b_j(x) \in \{-1, +1\}$. For any x in $\mathcal{X}$, let $b(x) = [b_1(x), \ldots, b_M(x)]^T$. Let, $R(\theta) \stackrel{\text{def}}{=} \mathbb{E}L(y\langle\theta, b(x)\rangle)$. Then, for any $T \geq 1$, and any $\theta \in \Delta_M$, when SMD-AMA is run with the parameter $\beta_0^2 = \frac{L_\phi^2}{\log(M)2^{\kappa+1}(\kappa)}$, where $\kappa > 1/2$, then the convex aggregation, $\hat{\theta}_T$, returned by the SMD-AMA algorithm after T rounds, satisfies the following excess risk inequality*

$$\mathbb{E}R(\hat{\theta}_T) \leq R(\theta) + T^{\kappa-1}\left(\sqrt{\frac{L_\phi^2 2^{\kappa+1}\log(M)}{\kappa}}\right).$$

Proof of Theorem 12 proceeds via the following lemma.

Lemma 9. *For any $\theta \in \Delta_M$, and for any $T \geq 1$, we have*

$$\sum_{t=1}^{T}\langle\theta_{t-1} - \theta, \nabla\mathcal{R}(\theta_{t-1}) \leq \beta_T\mathcal{R}(\theta) + \sum_{t=1}^{T}\frac{||g_t||_\infty^2}{2\beta_{t-1}} - \sum_{t=1}^{T}\langle\theta_{t-1} - \theta_t, \Delta_t(\theta_{t-1})\rangle.$$

The above lemma was first established in [54]. We shall not provide the proof, as it follows almost verbatim from the proof of Proposition 2 in [54]).

5.3.1 Proof of Theorem 12

Let $R(\theta) \stackrel{\text{def}}{=} \mathbb{E}L(y\langle\theta, b(x)\rangle)$. By definition,

$$\hat{\theta}_T = \frac{\sum_{t=1}^{T} \theta_{t-1}}{T}.$$

Since L is a convex function, hence by Jensen's inequality

$$\mathbb{E}R(\hat{\theta}_T) - \mathbb{E}R(\theta) \leq \frac{\sum_{t=1}^{T} \mathbb{E}R(\theta_{t-1}) - \mathbb{E}R(\theta)}{T} \tag{62}$$

Since $R(\theta)$ is a convex function of θ, hence we can use the subgradient to build an under-approximation to get

$$R(\theta_{t-1}) - R(\theta) \leq \langle \nabla\mathcal{R}(\theta_{t-1}), -\theta + \theta_{t-1}\rangle. \tag{63}$$

Putting together Equations 62, 63 we then get

$$\begin{aligned}
\mathbb{E}R(\hat{\theta}_T) - \mathbb{E}R(\theta) &\leq \frac{\sum_{t=1}^{T} \mathbb{E}\langle \nabla\mathcal{R}(\theta_{t-1}), \theta_{t-1} - \theta\rangle}{T} \\
&\leq \frac{1}{T}\Big[\beta_T \log(M) - \mathbb{E}\sum_{t=1}^{T}\langle \theta_{t-1} - \theta, g_t - \nabla R(\theta_{t-1})\rangle + \mathbb{E}\sum_{t=1}^{T}\frac{||g_t||_\infty^2}{2\beta_{t-1}}\Big].
\end{aligned}$$

In the above set of inequalities, to obtain the second inequality from the first we used Lemma 9. Since our data stream is drawn i.i.d. from the input distribution, hence $\mathbb{E}g_t - \nabla R(\theta_{t-1}) = 0$. This gets us

$$\begin{aligned}
\mathbb{E}R(\hat{\theta}_T) - \mathbb{E}R(\theta) &\leq \frac{\mathbb{E}\langle \nabla\mathcal{R}(\theta_{t-1}), \theta_{t-1} - \theta\rangle}{T} \\
&\leq \frac{1}{T}\Big[\beta_T \log(M) + \mathbb{E}\sum_{t=1}^{T}\frac{||g_t||_\infty^2}{2\beta_{t-1}}\Big] \\
&\stackrel{(a)}{\leq} \frac{1}{T}\left[\beta_T \log(M) + \sum_{t=1}^{T}\mathbb{E}\frac{L_\phi^2}{2p_t\beta_{t-1}}\right] \\
&\stackrel{(b)}{\leq} \frac{\beta_T \log(M)}{T} + \frac{L_\phi^2}{2T\beta_0}\sum_{t=1}^{T}\frac{1}{t^{1-\kappa}} \\
&\stackrel{(c)}{\leq} \frac{\beta_0(T+1)^\kappa \log(M)}{T} + \frac{L_\phi^2 T^{\kappa-1}}{2\beta_0\kappa} \\
&\stackrel{(d)}{\leq} \frac{2^\kappa\beta_0 T^\kappa \log(M)}{T} + \frac{L_\phi^2 T^{\kappa-1}}{2\beta_0\kappa} \\
&\leq 2T^{\kappa-1}\sqrt{\frac{L_\phi^2 2^{\kappa+1}\log(M)}{\kappa}}.
\end{aligned}$$

Inequality (a) was obtained by using the fact that $\mathbb{E}[\frac{Q_t}{p_t}] = 1$, and by approximating $|L'(y_i\langle\theta, b(x_i)\rangle)| \leq L_\phi$. Inequality (b) was obtained by substituting for β_{t-1} the expression $\beta_0 t^\kappa$, and upper bounding $\frac{1}{p_t}$ with $\frac{1}{\epsilon_t}$. This completes the proof.

5.4 Related Work

Madani et al. [65] considered the problem of active model aggregation, where given many models one has to choose a single best model from the collection. They model the problem as the coins problem, where a player is provided with a certain number of flips, and is allowed to flip coins until the budget runs out, after which the player has to report the coin which has the highest probability of turning up heads. A reinforcement learning approach, to the same problem, was taken in [57]. In this chapter, we are dealing with a more complicated problem of choosing the best model from the convex hull of a given set of models. Mamituska and Abe [66] combine the ideas of Query-by-committee, and boosting to come up with an active learning algorithm, where the current weighted majority is used to decide which point to query next. In order to obtain a weighted-majority of hypothesis, the authors suggest using ensemble techniques such as boosting and bagging. Trapeznikov et al. [90] introduced the ActBoost algorithm, which is an active learning algorithm in the boosting framework. Particularly, ActBoost works under the weak learning assumption [36], which assumes that there is a hypothesis with zero error rate, in the convex hull of the base classifiers. Under this assumption, the authors suggest a version space based algorithm, that maintains all the possible convex combinations of the base hypothesis that are consistent with the current data, and queries the labels of the points, on which two hypothesis in the current convex hull, disagree. By design, the ActBoost algorithm is very brittle. In contrast, we do not make any weak learning assumptions, and hence avoid the problems that ActBoost might face when weak learning assumption is not satisfied. Active learning algorithms, in the boosting framework, have also been

suggested by [53], but they do not admit any guarantees, and are somewhat heuristic.

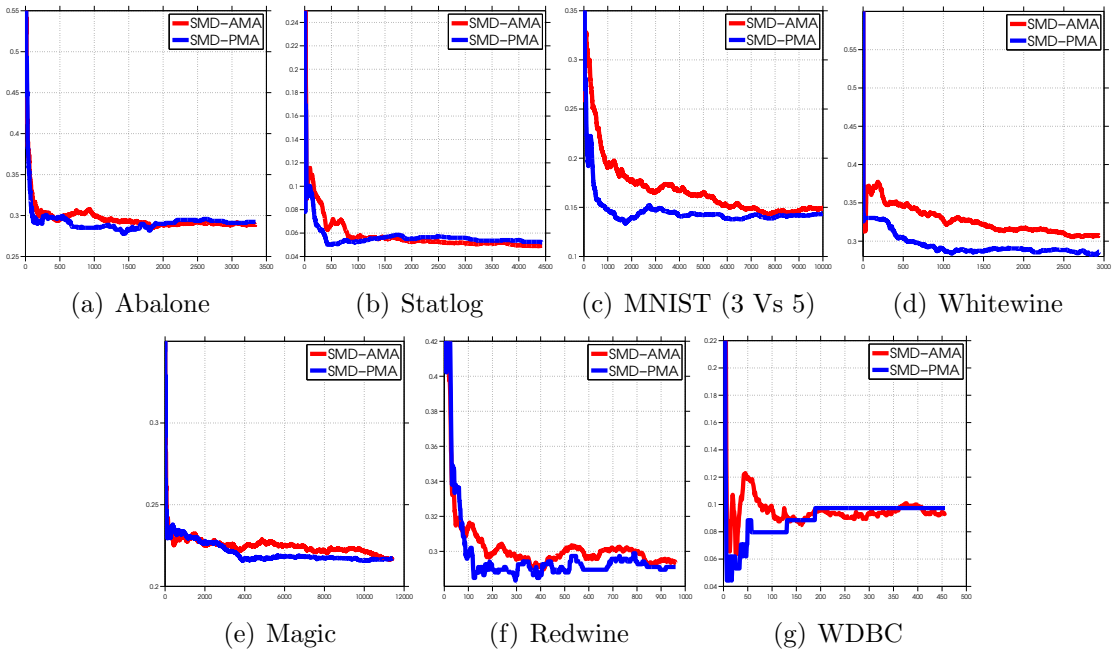
5.5 Experimental Results

We implemented SMD-AMA, along with SMD-PMA, and the QBB algorithm [66]. As mentioned before, QBB is an ensemble based active learning algorithm, which builds a committee via the AdaBoost algorithm. QBB works in an iterative fashion. In round t, QBB runs the AdaBoost algorithm, on the currently labeled dataset S_t, with the collection of models $\mathcal{B}$, to get a boosted model h_t. A boosted model is of the form $h_t(x) = \sum_{j=1}^{M} \alpha_{j,t} b_j(x)$, where $\alpha_{j,t} > 0$, for all $j = 1, \ldots, M$. To choose the next point to be queried, QBB generates a random sample of R points from the current set of unqueried points. Suppose this random sample is C_t. To choose the next point to be queried, we look for that point in C_t, whose margin w.r.t. h_t is the smallest. We then query for the label of this point. This process is repeated until some condition is satisfied (typically until a budget is exhausted).

5.5.1 Experimental Setup.

We used decision stumps along different dimensions, and with different thresholds, to form our set of basis models $\mathcal{B}$. A decision stump is a weak classifier that is characterized by a dimension j, and a threshold θ, and classifies a point $x \in \mathbb{R}^d$ as $\text{sgn}(x_j - \theta)$, where x_j is the j^{th} dimension of x. For all our experiments, we used 80 decision stumps along each dimension [3]. We make our set, $\mathcal{B}$, symmetric by adding $-b$ to $\mathcal{B}$, if $b \in \mathcal{B}$. Unless otherwise mentioned, the choice of κ is set to 0.65 for all of our SMD-AMA experiments. We report results on some standard UCI datasets.

[3] Using more decision stumps yielded insignificant improvement in test error, but increased computational complexity by a large amount



(a) Abalone (b) Statlog (c) MNIST (3 Vs 5) (d) Whitewine

(e) Magic (f) Redwine (g) WDBC

Figure 3: Comparison of the test error between SMD-AMA and SMD-PMA with the number of points seen. All results are reported by averaging over 10 repetitions of our experiments. The loss function used is the squared loss.

5.5.2 Comparison with Passive Learning

Our first set of experiments compare SMD-PMA to SMD-AMA. We run both SMD-AMA and SMD-PMA on our datasets, and use the hypothesis outputted by these algorithms, at the end of each round, to classify on a test dataset. In Figure 3, we plot the test error rate of both the algorithms with the number of points seen in the stream. Note that while SMD-PMA gets to see the label of each and every point, SMD-AMA gets to see only those labels which it queries. We used the squared loss function for SMD-AMA, and SMD-PMA. Finally, since SMD-AMA is a randomized algorithm, we report results averaged over 10 iterations. From Table 6, it is clear that for all datasets but Whitewine, both SMD-AMA and SMD-PMA attain almost the same error rate, after finishing a single pass through the stream. Figure 3 shows how the test error changes for SMD-PMA and SMD-AMA with the number of points seen in the stream. While in the case of Abalone, and Statlog, SMD-AMA quickly

Table 6: Comparison of the test error between SMD-AMA and SMD-PMA, and number of queries made by SMD-AMA on different datasets. All results are for the hypothesis returned by the algorithms at the end of the stream. All results are reported by averaging over 10 repetitions of our experiments.

Dataset	SMD-AMA	SMD-PMA	Number of Queries	Fraction Queried
Abalone	0.2889	0.2922	440.2	0.1317
Statlog	0.0491	0.0520	2984	0.6728
MNIST	0.1496	0.1442	931.6	0.0932
Whitewine	0.3075	0.2864	406.8	0.1351
Magic	0.2166	0.2171	2450	0.2146
WDBC	0.0938	0.0973	112	0.2469
Redwine	0.2933	0.2911	540	0.5625

Table 7: Comparison of the test loss between SMD-AMA and SMD-PMA for the hypothesis returned by the algorithms at the end of the stream. All results are reported by averaging over 10 repetitions of our experiments. The loss function used is the squared loss.

Dataset	SMD-AMA	SMD-PMA
Abalone	0.7992	0.7611
Statlog	0.3242	0.3396
MNIST	0.5517	0.5381
Whitewine	0.7894	0.7361
Maigc	0.6315	0.6305
WDBC	0.3178	0.2933
Redwine	0.7455	0.7660

catches up with SMD-PMA (and in fact slightly surpasses SMD-PMA), in the case of MNIST, the difference between SMD-AMA and SMD-PMA closes only after having seen about 80% of the stream. In the case of Whitewine, SMD-PMA is uniformly better than our active learning algorithm, SMD-AMA. The difference in error rates, between SMD-PMA and SMD-AMA, at the end of the stream is about 1.32%. In the case of Magic, Redwine datasets, the difference in performance of SMD-AMA and SMD-PMA is negligible. On WDBC, SMD-PMA initially does better than SMD-AMA, but after having queried the labels of sufficient number of points, SMD-AMA does just as well as SMD-PMA.

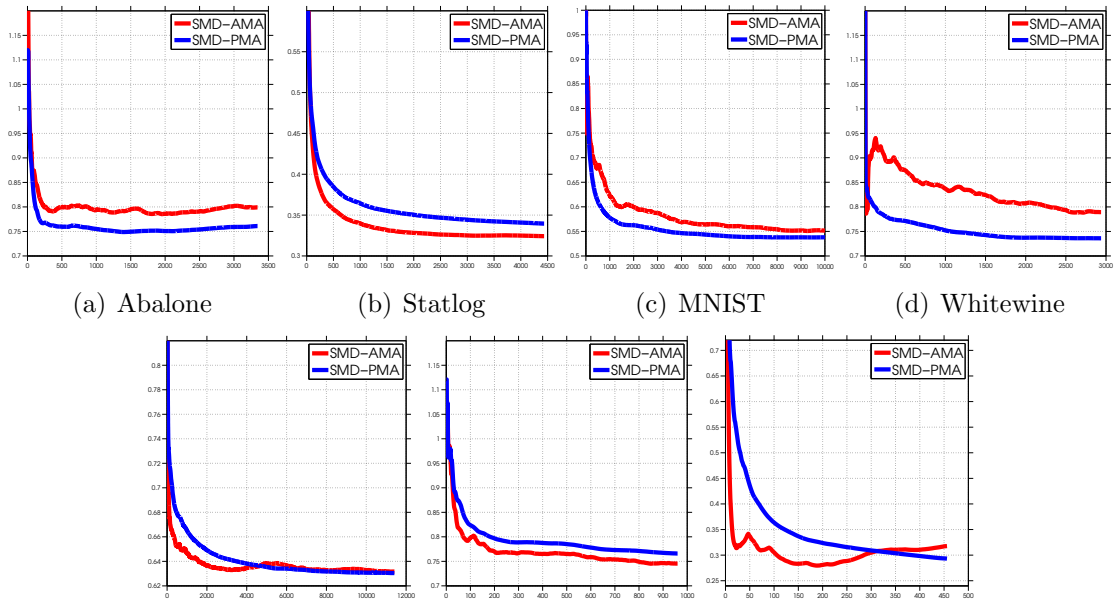
The number of queries made in Abalone, MNIST, and Whitewine is less than 14%

of the length of the stream, which implies that we do as well as passive learning, for Abalone, and MNIST, at the expense of far fewer labels. In the case of Statlog, and Redwine datasets the number of queries made is comparatively larger, about 67.28%, and 56.25% of the size of the dataset respectively. On Magic and WDBC the fraction of queries made is less than 25% of the number of training points in the dataset.

In Table 7, we report the loss on the test data, of SMD-AMA, SMD-PMA at the end of the data stream. Figure 4 reports the test loss of both the algorithms with the number of samples seen in the stream. Note that while test error is always between 0 and 1, the test loss can be larger than 1. For instance, in Figure 4 (d), the loss can be as large as 1.16. In fact for the convex aggregation model that we consider in this chapter, and with the squared loss, the maximum loss can be as large as $\max_{z\in[-1,1]}(1-z)^2 = 4$. The purpose of these experiments was to examine how the difference between the test loss suffered by SMD-AMA, and SMD-PMA, changes with the number of points seen in the stream. In the case of Statlog, and MNIST the difference in losses is generally smaller than in the case of Abalone and Whitewine, and on Magic, Redwine, and WDBC datasets SMD-AMA generally has suffers slightly smaller loss than SMD-PMA. However, since the scale for test loss is larger than 1, these results seem to imply that both SMD-AMA and SMD-PMA have almost similar rates of decay for the test loss.

5.5.3 Number of Queries Vs Number of Points Seen

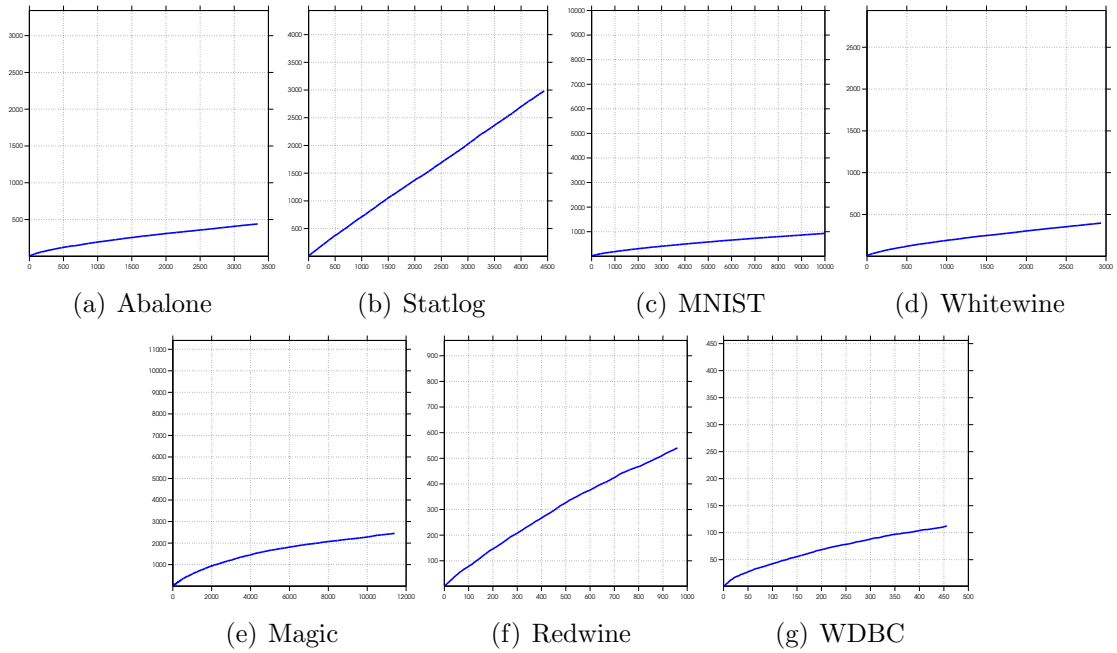
Figure 5 shows how the number of queries made by SMD-AMA scale with the number of points seen in the steam on all the four datasets. This scaling is almost linear in the case of Statlog. This was expected, given the fact that on Statlog, we query the labels of almost 65% of the points in the stream. However, in the case of Abalone, MNIST, and Whitewine, the scaling seems to be sublinear. Based on these experimental results, we expect that on Magic, and WDBC datasets, the scaling of the number of



(a) Abalone (b) Statlog (c) MNIST (d) Whitewine

(e) Magic (f) Redwine (g) WDBC

Figure 4: Comparison of the test loss of SMD-AMA and SMD-PMA with the number of points seen. All results are reported by averaging over 10 repetitions of our experiments. The loss function used is the squared loss.



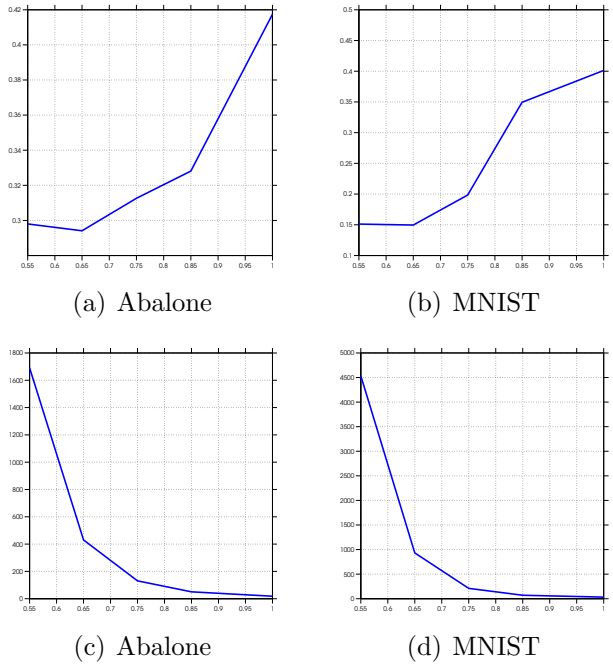
(a) Abalone (b) Statlog (c) MNIST (d) Whitewine

(e) Magic (f) Redwine (g) WDBC

Figure 5: The number of queries made by SMD-AMA as a function of the number of data points seen. On the x-axis is the number of points seen, and on the y-axis is the number of labels requested.

queries w.r.t. the number of points to be sublinear; while on the Redwine dataset, like Statlog dataset, we expect this scaling to be linear.

5.5.4 Effect of Parameter κ.



(a) Abalone (b) MNIST

(c) Abalone (d) MNIST

Figure 6: The plots in the top row show the test error of SMD-AMA (on y-axis), at the end of the stream, as a function of the parameter κ (on the x-axis). The bottom two plots show the number of queries made (on y-axis) as a function of the parameter κ (on the x-axis).

The effect of parameter κ, can be studied as follows. Step 3 of SMD-AMA, sets the value of ϵ_t to $t^{1-2\kappa}$. From Step 5 of SMD-AMA, it is clear that $p_t \geq \epsilon_t$. Hence the average number of queries scales faster than $\sum_{t=1}^{T} t^{1-2\kappa}$. This implies that,

$$\mathbb{E}[\text{Number of Queries}] = \begin{cases} \Omega(T^{2-2\kappa}) \text{ if } \frac{1}{2} \leq \kappa < 1 \\ \Omega(\log(T)) \text{ if } \kappa = 1. \end{cases}$$

Theorem 12 says that the excess risk of SMD-AMA has an upper bound that scales as $O(T^{\kappa-1})$. Hence from this discussion we know that large κ leads to a smaller lower bound on the expected number of queries made, but a larger upper bound on the excess risk. Figure 6 demonstrates the trade-off between test error and the number

Table 8: Comparison of the error rate of QBB and SMD-AMA for a given budget.

Dataset	Budget	Error rate of SMD-AMA	Error rate of QBB
Abalone	441	0.2889	0.3293
Statlog	2984	0.0491	0.0407
MNIST	932	0.1496	0.1756
Whitewine	398	0.3075	0.3543
Redwine	540	0.2933	0.3146
WDBC	115	0.0938	0.0796
Magic	2450	0.2166	0.2499

of label queries made by SMD-AMA by changing parameter κ. As we can see from these plots, larger κ lead to small number of queries, but a larger test error. On the other hand, when we choose a small value of κ, the number of queries made is large, but the test error is small. We expect similar results to hold true for other datasets too.

5.5.5 Comparison with QBB

In contrast to SMD-PMA and SMD-AMA, QBB is a pool based active learning algorithm, and not a stream based active learning algorithm. Hence, QBB has access to the entire set of unlabeled data points, and in each round can choose one data point to query. In order to provide a fair comparison of QBB and SMD-AMA, we used the number of queries made by SMD-AMA from our first set of experiments, as a budget parameter for the QBB algorithm, and report the error rate of the hypothesis returned by SMD-AMA, and QBB at the end of the budget. It is clear from Table 8 that, for all the datasets except Statlog, and WDBC, QBB is significantly inferior to SMD-AMA under the given budget constraints. For the Statlog dataset, the test error of SMD-AMA and QBB are comparable, and we guess this is because the number of training points queried by SMD-AMA, which is given as a budget parameter to our QBB experiments, is a very large fraction of the size of the dataset. On the WDBC dataset, we believe that it might be possible to obtain better results

for both SMD-AMA, and SMD-PMA by carefully fine-tuning the step size used in the mirror-descent step.

5.6 Conclusions and Discussions

In this chapter we considered active learning of convex aggregation of classification models. We presented a stochastic mirror descent algorithm, which uses importance weighting to obtain unbiased importance weighted stochastic gradients of a convex risk function. We established excess risk guarantees of the resultant convex aggregation outputted by SMD-AMA w.r.t. the best possible convex aggregate. Experimental results show that SMD-AMA produces as good error rates as a passive learning algorithm while querying substantially fewer number of points. In particular for some datasets we were able to achieve error rates on par with passive learning by using only 13% of the data that a passive learner would use. This work can be extended in many directions, some of which are stated below

Improved rates for excess risk. Our bound on the excess risk is seemingly pessimistic. SMD-PMA is known to attain excess risk rates of $O(\sqrt{\frac{\log(M)}{T}})$, while in our case the excess risk is of the order $O(\frac{\sqrt{\log(M)}}{T^{1-\kappa}})$. Our experimental results from Section 5.5 seem to suggest that the test loss of the hypothesis returned by SMD-AMA, and SMD-PMA are very close. Basing on this experimental evidence, we conjecture that it might be possible to provide a sharper bound on the excess risk of SMD-AMA.

5.7 Bibliographic Notes

This chapter is currently under submission at AISTATS 2014.

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

In this thesis, we designed active learning algorithms by using ideas from sequential analysis. By using the stochastic mirror-descent algorithm, we designed pool based, and stream based active learning algorithms with provable excess risk guarantees. We showed how one could connect the problem of active learning to the problem of multi-armed bandits, which facilitates transfer of ideas such as confidence bounds, and barrier type regularizers commonly used in the bandit and online optimization to active learning. We shall now mention a few problems of interest.

6.0.1 Stronger connections between Active Learning and Problems in Sequential Analysis.

We believe that there is tremendous potential for ideas from multi-armed bandits, and various other extensions of multi-armed bandits such as contextual bandits, bandit optimization to be used for active learning problems. Most of these algorithms are very simple, efficient and hence should be useful in designing simple, efficient active learning algorithms.

6.0.2 Active Learning for Regression problems.

All the contributions in this thesis are geared towards active learning for the binary classification problem. There has been work for active learning in regression problems both on the algorithmic side [28, 34] and on the complexity side [24]. However, to our knowledge, the sequential lens to active learning that we have taken in this thesis, has not been used for active learning in regression problems. A seemingly related problem to active regression is that of efficient global optimization, where we minimize an unknown function f by using as few observations $f(x)$ as possible. For efficient

global optimization, where exploration-exploitation trade-off enters the picture, a well known heuristic is expected-improvement. Bull [22] analyzed asymptotic properties of this algorithm under certain conditions. While it is tempting to equate active regression problem to global optimization problem by saying that in both the cases we are trying to learn a model in a certain model class that minimizes expected squared loss, it is important to understand that the queries in global optimization are far more powerful, as we can ask for the evaluation of f at any point x. In contrast in regression problems under the distributional setting, we can ask the label for only those data points which have been sampled from the underlying distribution.

Use of Robust Estimators in Active Learning. A recurrent theme in our work has been the use of importance sampling, which was used to build importance weighted estimate of the risk of a hypothesis. A major problem with such importance weighted estimators is that they have high variance. Derivation of importance weighted estimators of risk, which have small variance, seems to be an interesting avenue to pursue. An interesting approach would be to use ideas similar to the ones used in the field of robust estimation such as winsorized mean [15] and Catoni's M estimator [25].

Active Learning under specific statistical assumptions. It is of tremendous practical and theoretical interest to design active learning algorithms, which are both computationally efficient, and also label efficient, under certain special assumptions on the data generating process. For example, we now know that the class of single index models can be efficiently learned [56], both from the statistical and computational viewpoint in the passive learning setting. How can we learn such models in a label-efficient, and computational, and statistically efficient way. Similarly, how can we design active learning algorithms for learning additive models.

REFERENCES

[1] Abernethy, J., Hazan, E., and Rakhlin, A., "Competing in the dark: An efficient algorithm for bandit linear optimization," in *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, vol. 3, p. 3, 2008.

[2] Agrawal, R., "Sample mean based index policies with o (log n) regret for the multi-armed bandit problem," *Advances in Applied Probability*, pp. 1054–1078, 1995.

[3] Audibert, J.-Y., Munos, R., and Szepesvári, C., "Exploration-exploitation tradeoff using variance estimates in multi-armed bandits," *Theor. Comput. Sci.*, 2009.

[4] Audibert, J., Munos, R., and Szepesvári, C., "Exploration–exploitation tradeoff using variance estimates in multi-armed bandits," *Theoretical Computer Science*, vol. 410, no. 19, pp. 1876–1902, 2009.

[5] Auer, P., Cesa-Bianchi, N., and Fischer, P., "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2, pp. 235–256, 2002.

[6] Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R., "The non-stochastic multiarmed bandit problem," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2002.

[7] Balcan, M., Beygelzimer, A., and Langford, J., "Agnostic active learning," *JCSS*, vol. 75, no. 1, 2009.

[8] Baram, Y., El-Yaniv, R., and Luz, K., "Online choice of active learning algorithms," *The Journal of Machine Learning Research*, vol. 5, pp. 255–291, 2004.

[9] Barnard, G., "Sequential tests in industrial statistics," *Supplement to the Journal of the Royal Statistical Society*, vol. 8, no. 1, pp. 1–26, 1946.

[10] Bartlett, P., Dani, V., Hayes, T., Kakade, S., Rakhlin, A., and Tewari, A., "High-probability regret bounds for bandit online linear optimization," *COLT*, 2008.

[11] Baum, E. and Lang, K., "Query learning can work poorly when a human oracle is used," in *IJCNN*, 1992.

[12] Beck, A. and Teboulle, M., "Mirror descent and nonlinear projected subgradient methods for convex optimization," *Operations Research Letters*, vol. 31, no. 3, pp. 167–175, 2003.

[13] Berry, D. A. and Fristedt, B., *Bandit Problems: Sequential Allocation of Experiments*. Monographs on Statistics and Applied Probability, London: Chapman & Hall, 1985.

[14] Beygelzimer, A., Dasgupta, S., and Langford, J., "Importance weighted active learning," in *ICML*, 2009.

[15] Bickel, P. J., "On some robust estimates of location," *The Annals of Mathematical Statistics*, vol. 36, no. 3, pp. 847–858, 1965.

[16] Boucheron, S., Bousquet, O., and Lugosi, G., "Theory of classification: A survey of some recent advances," *ESAIM: P&S*, vol. 9, pp. 323–375, 2005.

[17] Boyd, S. and Vandenberghe, L., *Convex Optimization*. Cambridge University Press, 2004.

[18] Bubeck, S., Cesa-Bianchi, N., Kakade, S., and others, "Towards minimax policies for online linear optimization with bandit feedback," in *Annual Conference on Learning Theory*, pp. 41–1, Microtome, 2012.

[19] Bubeck, S., "Introduction to online optimization," *Lecture Notes*, 2011.

[20] Bubeck, S. and Cesa-Bianchi, N., "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *arXiv preprint arXiv:1204.5721*, 2012.

[21] Buja, A., Stuetzle, W., and Shen, Y., "Loss functions for binary class probability estimation and classification: Structure and applications." 2005.

[22] Bull, A., "Convergence rates of efficient global optimization algorithms," *The Journal of Machine Learning Research*, vol. 12, pp. 2879–2904, 2011.

[23] Carpentier, A., Lazaric, A., Ghavamzadeh, M., Munos, R., and Auer, P., "Upper-confidence-bound algorithms for active learning in multi-armed bandits," in *Algorithmic Learning Theory*, pp. 189–203, Springer, 2011.

[24] Castro, R., Willett, R., and Nowak, R., "Faster rates in regression via active learning," *Advances in Neural Information Processing Systems*, vol. 18, p. 179, 2006.

[25] Catoni, O., "Challenging the empirical mean and empirical variance: a deviation study," in *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, vol. 48, pp. 1148–1185, Institut Henri Poincaré, 2012.

[26] Cesa-Bianchi, N. and Lugosi, G., *Prediction, learning, and games*. Cambridge Univ Press, 2006.

[27] Cesa-Bianchi, N., Gentile, C., and Orabona, F., "Robust bounds for classification via selective sampling," in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 121–128, ACM, 2009.

[28] Chaloner, K. and Verdinelli, I., "Bayesian experimental design: A review," *Statistical Science*, pp. 273–304, 1995.

[29] Chu, W., Zinkevich, M., Li, L., Thomas, A., and Tseng, B., "Unbiased online active learning in data streams," in *SIGKDD*, 2011.

[30] Cohn, D., Atlas, L., and Ladner, R., "Improving generalization with active learning," *Machine Learning*, vol. 15, no. 2, 1994.

[31] Dasgupta, S., Hsu, D., and Monteleoni, C., "A general agnostic active learning algorithm," *NIPS*, 2007.

[32] Dekel, O., Gentile, C., and Sridharan, K., "Robust selective sampling from single and multiple teachers.," in *COLT*, pp. 346–358, 2010.

[33] Dietterich, T. G., "Ensemble methods in machine learning," in *Multiple classifier systems*, pp. 1–15, Springer, 2000.

[34] Fedorov, V. V., *Theory of optimal experiments*. Access Online via Elsevier, 1972.

[35] Freund, Y. and Schapire, R., "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Computational learning theory*, pp. 23–37, Springer, 1995.

[36] Freund, Y., Schapire, R. E., and others, "Experiments with a new boosting algorithm," in *ICML*, vol. 96, pp. 148–156, 1996.

[37] Freund, Y., Seung, H. S., Shamir, E., and Tishby, N., "Selective sampling using the query by committee algorithm," in *Machine Learning*, pp. 133–168, 1997.

[38] Friedman, J., Hastie, T., and Tibshirani, R., "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)," *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.

[39] Ganti, R. and Gray, A., "Upal: Unbiased pool based active learning," in *AISTATS*, 2012.

[40] Ganti, R. and Gray, A., "Building bridges: Viewing active learning from the multi-armed bandit lens," in *Association for Uncertainty in Artificial Intelligence*, 2013.

[41] Ghosh, M., Mukhopadhyay, N., and Sen, P., *Sequential estimation*, vol. 904. Wiley-Interscience, 2011.

[42] Golovin, D. and Krause, A., "Adaptive submodularity: A new approach to active learning and stochastic optimization.," in *COLT*, pp. 333–345, 2010.

[43] Gonen, A., Sabato, S., and Shalev-Shwartz, S., "Efficient active learning of halfspaces: an aggressive approach," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pp. 480–488, 2013.

[44] Hanneke, S., "A bound on the label complexity of agnostic active learning," in *Proceedings of the 24th international conference on Machine learning*, pp. 353–360, ACM, 2007.

[45] Hanneke, S. and Yang, L., "Surrogate losses in passive and active learning," *arXiv preprint arXiv:1207.3772*, 2012.

[46] Hanneke, S., "Adaptive rates of convergence in active learning.," in *COLT*, Citeseer, 2009.

[47] Hanneke, S., "Rates of convergence in active learning," *The Annals of Statistics*, vol. 39, no. 1, pp. 333–361, 2011.

[48] Hoi, S., Jin, R., Zhu, J., and Lyu, M., "Batch mode active learning and its application to medical image classification," in *ICML*, 2006.

[49] Horn, R. and Johnson, C., *Matrix analysis*. Cambridge Univ Press, 1990.

[50] Hsu, D., Kakade, S., and Zhang, T., "An analysis of random design linear regression," *Arxiv preprint arXiv:1106.2363*, 2011.

[51] Hsu, D., Kakade, S., and Zhang, T., "Dimension-free tail inequalities for sums of random matrices," *Arxiv preprint arXiv:1104.1672*, 2011.

[52] Hsu, D. J., *Algorithms for active learning*. PhD. Thesis, University of California, San Diego, Computer Science Department, 2010.

[53] Iyengar, V. S., Apte, C., and Zhang, T., "Active learning using adaptive resampling," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 91–98, ACM, 2000.

[54] Juditsky, A., Nazin, A., Tsybakov, A., and Vayatis, N., "Recursive aggregation of estimators by the mirror descent algorithm with averaging," *Problems of Information Transmission*, vol. 41, no. 4, pp. 368–384, 2005.

[55] Kakade, S., Sridharan, K., and Tewari, A., "On the complexity of linear prediction: Risk bounds, margin bounds, and regularization," *Advances in Neural Information Processing Systems*, vol. 22, pp. 793–800, 2008.

[56] Kalai, A. T. and Sastry, R., "The isotron algorithm: High-dimensional isotonic regression.," in *COLT*, 2009.

[57] Kapoor, A. and Greiner, R., "Reinforcement learning for active model selection," in *Proceedings of the 1st international workshop on Utility-based data mining*, pp. 17–23, ACM, 2005.

[58] Kearns, M. J., Schapire, R. E., and Sellie, L. M., "Toward efficient agnostic learning," *Machine Learning*, vol. 17, no. 2-3, pp. 115–141, 1994.

[59] Koltchinskii, V., "Rademacher complexities and bounding the excess risk in active learning," *The Journal of Machine Learning Research*, vol. 11, pp. 2457–2485, 2010.

[60] Kumar, P., "A survey of some results in stochastic adaptive control," *SIAM Journal on Control and Optimization*, vol. 23, no. 3, pp. 329–380, 1985.

[61] Kushner, H. and Yin, G., *Stochastic approximation algorithms and applications*. Springer New York, 1997.

[62] Lanckriet, G. R. and Sriperumbudur, B. K., "On the convergence of the concave-convex procedure," in *Advances in neural information processing systems*, pp. 1759–1767, 2009.

[63] Lewis, D. and Gale, W., "A sequential algorithm for training text classifiers," in *SIGIR*, 1994.

[64] Litvak, A., Pajor, A., Rudelson, M., and Tomczak-Jaegermann, N., "Smallest singular value of random matrices and geometry of random polytopes," *Advances in Mathematics*, vol. 195, no. 2, pp. 491–523, 2005.

[65] Madani, O., Lizotte, D. J., and Greiner, R., "Active model selection," in *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pp. 357–365, AUAI Press, 2004.

[66] Mamitsuka, N., "Query learning strategies using boosting and bagging," in *Machine learning: proceedings of the fifteenth international conference (ICML'98)*, p. 1, Morgan Kaufmann Pub, 1998.

[67] Mason, L., Baxter, J., Bartlett, P., and Frean, M., "Functional gradient techniques for combining hypotheses," *NIPS*, pp. 221–246, 1999.

[68] McCallum, A. and Nigam, K., "Employing EM and pool-based active learning for text classification," in *ICML*, 1998.

[69] McCallum, A., Nigam, K., and others, "Employing em and pool-based active learning for text classification.," in *ICML*, vol. 98, pp. 350–358, 1998.

[70] Melville, P. and Mooney, R. J., "Diverse ensembles for active learning," in *Proceedings of the twenty-first international conference on Machine learning*, p. 74, ACM, 2004.

[71] Nemirovski, A., "Topics in non-parametric statistics," *Lectures on probability theory and statistics*, 2000.

[72] Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A., "Robust stochastic approximation approach to stochastic programming," *SIAM Journal on Optimization*, vol. 19, no. 4, pp. 1574–1609, 2009.

[73] Nemirovski, A. and Yudin, D., *Problem complexity and method efficiency in optimization*. Wiley New York, 1983.

[74] Orabona, F. and Cesa-Bianchi, N., "Better algorithms for selective sampling," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 433–440, 2011.

[75] Quinonero, J., Sugiama, M., Schwaighofer, A., and Lawrence, N., "Dataset shift in machine learning," 2008.

[76] Raginsky, M. and Rakhlin, A., "Information complexity of black-box convex optimization: A new look via feedback information theory," in *Communication,*

Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on, pp. 803–510, IEEE, 2009.

[77] Ramdas, A. and Singh, A., "Optimal rates for stochastic convex optimization under tsybakov noise condition," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pp. 365–373, 2013.

[78] Reid, M. D. and Williamson, R. C., "Surrogate regret bounds for proper losses," in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 897–904, ACM, 2009.

[79] Robbins, H., "Some aspects of the sequential design of experiments," *Bulletin of the American Mathematical Society*, vol. 58, no. 5, pp. 527–535, 1952.

[80] Salomon, A. and Audibert, J.-Y., "Deviations of stochastic bandit regret," in *Algorithmic Learning Theory*, pp. 159–173, Springer, 2011.

[81] Schapire, R. E., "The boosting approach to machine learning: An overview," *Lecutre Notes In Statistics-New York-Springer Verlag*, pp. 149–172, 2003.

[82] Scheffer, T., Decomain, C., and Wrobel, S., "Active hidden markov models for information extraction," in *Advances in Intelligent Data Analysis*, pp. 309–318, Springer, 2001.

[83] Settles, B., "Active learning literature survey," Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[84] Settles, B. and Craven, M., "An analysis of active learning strategies for sequence labeling tasks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1070–1079, Association for Computational Linguistics, 2008.

[85] Settles, B., Craven, M., and Friedland, L., "Active learning with real annotation costs," in *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*, pp. 1–10, 2008.

[86] Siegmund, D., *Sequential analysis: tests and confidence intervals*. Springer, 2010.

[87] Sugiyama, M., Krauledat, M., and Müller, K.-R., "Covariate shift adaptation by importance weighted cross validation," *The Journal of Machine Learning Research*, vol. 8, pp. 985–1005, 2007.

[88] Sutton, R. S. and Barto, A. G., *Reinforcement learning: An introduction*, vol. 1. Cambridge Univ Press, 1998.

[89] Tong, S. and Koller, D., "Support vector machine active learning with applications to text classification," *JMLR*, 2002.

[90] Trapeznikov, K., Saligrama, V., and Castanón, D., "Active boosted learning (actboost)," *AISTATS*, 2010.

[91] Tropp, J., "User-friendly tail bounds for sums of random matrices," *Arxiv preprint arXiv:1004.4389*, 2010.

[92] Tsybakov, A., "Optimal aggregation of classifiers in statistical learning," *Annals of Statistics*, vol. 32, no. 1, pp. 135–166, 2004.

[93] Vapnik, V. N. and Chervonenkis, A. Y., "On the uniform convergence of relative frequencies of events to their probabilities," *Theory of Probability & Its Applications*, vol. 16, no. 2, pp. 264–280, 1971.

[94] Wald, A., "Sequential tests of statistical hypotheses," *The Annals of Mathematical Statistics*, pp. 117–186, 1945.

[95] Xiaojin, Z., *Semi-Supervised Learning with Graphs*. PhD thesis, Carnegie Mellon University, School of Computer Science, 2005.

[96] Zhang, T., "Statistical behavior and consistency of classification methods based on convex risk minimization," *Annals of Statistics*, vol. 32, no. 1, 2004.

[97] Zhao, P., Hoi, S. C., and Zhuang, J., "Active learning with expert advice," in *Uncertainty in Artificial Intelligence*, 2013.