UNIVERSITY OF CALIFORNIA, SAN DIEGO

Active Learning and Confidence-rated Prediction

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Computer Science

by

Chicheng Zhang

Committee in charge:

Professor Kamalika Chaudhuri, Chair
Professor Sanjoy Dasgupta
Professor Fan Chung Graham
Professor Tara Javidi
Professor Lawrence Saul

2017

The dissertation of Chicheng Zhang is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2017

DEDICATION

To my family.

EPIGRAPH

We must know - we will know.

—David Hilbert

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Professor Kamalika Chaudhuri. Kamalika's guidance and support over the past five years makes me passionate about research, and her vision shaped my view to the field of machine learning research immensely. Her clarity of explanation, and advice on writing and presentation will also continue to influence me in my career. I would also like to thank Professors Sanjoy Dasgupta, Fan Chung Graham, Tara Javidi, and Lawrence Saul for serving on my thesis committee and offering many insightful comments.

I was fortunate to have a wonderful internship at Yahoo Labs in Summer 2015 - thanks to my mentors Alina Beygelzimer, Daniel Hsu and John Langford, who provided me with careful guidance, and taught me how to formulate a practically motivated problem and seek its solutions. Thanks to my mentors Alina Beygelzimer and Francesco Orabona for careful guidance in my internship at Yahoo Research in Summer 2016, in which I enjoyed exploring the realm of online learning and bandit problems. Thanks to Kevin Chen and Jimin Song at Rutgers for close collaborations on spectral learning for epigenomics, from which I gained lots of hands-on experience working with real data. Thanks to my officemate Songbai Yan for close collaborations on active learning of halfspaces with noise, and clearing out several theoretical obstacles.

I would like to thank my undergraduate advisor, Professor Liwei Wang, who introduced me to machine learning theory and encouraged me to conduct research in this field. I am also grateful to my groupmates and friends Yanqi Dai, Kai Fan, Mingyang Jiang, Chi Jin, Wenyi Liu, Dongheng Sun, Ziteng Wang, Hongyi Zhang, Yang Zhang and many others from whom I learned tremendously in my undergrad at PKU.

I learned a lot from interacting with fellow graduate students and postdocs at UCSD - many thanks to Julaiti Alafate, Akshay Balsubramani, Matt Der, Joseph Geumlek, Long Jin, Dokyum Kim, Zack Lipton, Shuang Liu, Suqi Liu, Stefanos Poulis, Shuang Song, Yan Shu, Matus Telgarsky, Chris Tosh, Sharad Vikram, Eric Yizhen Wang, Xinan Wang, Saining Xie, Songbai Yan, Zhen Zhai, Jiapeng Zhang and many others for sharing their research insights in machine learning, and thanks to Yacong Ding, Sidi Fu, Marito Hayashi, Yihan Jiang, Yanqin Jin, Jing Li, Hao Zhuang, Nan Zou and many others for sharing with me a broader view of scientific research.

Lastly, I would like to thank my family, friends, and my girlfriend Yue for their love and support in my life.

Chapter 4, is based on the material as it appears in Advances in Neural Information

Processing Systems 2015 (Chicheng Zhang and Kamalika Chaudhuri, "Active Learning from Weak and Strong Labelers"). The dissertation author is the primary investigator and author of this material.

Chapter 5, is based on the material as it appears in Advances in Neural Information Processing Systems 2016 (Alina Beygelzimer, Daniel Hsu, John Langford and Chicheng Zhang, "Search Improves Label for Active Learning"). The dissertation author is the primary investigator and author of this material.

Chapter 7, is based on the material as it appears in Conference on Learning Theory 2016 (Chicheng Zhang and Kamalika Chaudhuri, "The Extended Littlestone's Dimension for Learning with Mistakes and Abstentions"). The dissertation author is the primary investigator and author of this material.

Chapter 8, is based on the material as it appears in Advances in Neural Information Processing Systems 2014 (Chicheng Zhang and Kamalika Chaudhuri, "Beyond Disagreement-based Agnostic Active Learning"). The dissertation author is the primary investigator and author of this material.

Thanks to many anonymous reviewers for helpful comments. Much of this work is supported by NSF IIS-1162581 and IIS-1617157.

VITA

2012	B. S. in Machine Intelligence, Peking University, China
2015	M. S. in Computer Science, University of California, San Diego, USA
2016	C. Phil in Computer Science, University of California, San Diego, USA
2017	Ph. D. in Computer Science, University of California, San Diego, USA

PUBLICATIONS

Songbai Yan and Chicheng Zhang, "Revisiting Perceptron: Efficient and Label-Optimal Active Learning of Halfspaces". Manuscript.

Alina Beygelzimer, Francesco Orabona and Chicheng Zhang. "Efficient Online Bandit Multiclass Learning with $\tilde{O}(\sqrt{T})$ Regret". ICML 2017.

Alina Beygelzimer, Daniel Hsu, John Langford and Chicheng Zhang, "Search Improves Label for Active Learning". NIPS 2016.

Chicheng Zhang and Kamalika Chaudhuri, "The Extended Littlestone's Dimension for Learning with Mistakes and Abstentions". COLT 2016.

Chicheng Zhang and Kamalika Chaudhuri, "Active Learning from Weak and Strong Labelers". NIPS 2015.

Chicheng Zhang, Jimin Song, Kevin C. Chen and Kamalika Chaudhuri, "Spectral Learning of Large Structured HMMs for Comparative Epigenomics". NIPS 2015.

Chicheng Zhang and Kamalika Chaudhuri, "Beyond Disagreement-based Agnostic Active Learning". NIPS 2014.

ABSTRACT OF THE DISSERTATION

Active Learning and Confidence-rated Prediction

by

Chicheng Zhang

Doctor of Philosophy in Computer Science

University of California, San Diego, 2017

Professor Kamalika Chaudhuri, Chair

This thesis studies active learning and confidence-rated prediction, and the interplay between these two notions.

Active learning is a machine learning paradigm that allows a learner to perform label queries over the examples interactively. The goal of active learning is to get an accurate classifier using only a few label queries. In this thesis, we take a step further in the study of active learning, with a focus on active learning with complex queries. Specifically:

- We study the problem of active learning with weak and strong labelers. We present a statistically consistent algorithm that has a lower cost complexity compared to learning with the strong labeler alone, under certain conditions.
- We consider active learning with a novel type of queries, namely search queries. We show that in the setting of model selection, using the search queries can substantially reduce the

labeling effort of active learning.

Confidence-rated prediction considers the learning setting where the learned classifier is allowed to abstain, i.e. to predict "I Don't know". In this setting, abstaining rather than making a thoughtless classification decision may sometimes be preferable. In this thesis, we study confidence-rated prediction in batch and online settings, and advance the state of the art results. Specifically:

- In the batch setting, we propose a linear program based algorithm that has some optimality properties, and has superior performance over previous approaches.
- In the online setting, we propose an algorithm that achieves minimax optimal tradeoffs between its performance measures, and establish a novel combinatorial measure called Extended Littlestone's Dimension that characterizes this tradeoff.

Furthermore, we propose confidence-based active learning, establishing a connection between active learning and confidence-rated prediction. We show that our confidence-based active learning algorithm achieves statistical consistency, works for general hypothesis classes and data distributions, and has a lower label complexity compared to the state of the art active learning algorithms.

Chapter 1

Introduction

1.1 Active Learning

Active learning is a machine learning paradigm that allows a learner to perform label queries over the examples interactively. Its goal is to get an accurate classifier using only a few label queries. This is in sharp contrast with the *passive learning* paradigm, in which the learner directly draws labeled examples at random from the data distribution.

Active learning is motivated by the large volume of unlabeled data available and expensive labeling effort. Take the task of part-of-speech tagging as an example. In this setting, the unlabeled examples are sentences from corpora, and the labels are part-of-speech tags for each word. For instance, the sentence "I see a car" has labels (Pronoun, Verb, Determiner, Noun). Obtaining labels of sentences requires substantial human effort, as a labeling expert needs to read and understand the sentences before tagging. By using active learning, our hope is to reduce the label requirement by utilizing the power of interactive label queries. A typical active learning algorithm alternates between two steps: first, it makes label queries on some unlabeled examples based on the current model; second, it updates the current model based on the newly acquired labeled examples.

The past few decades have witnessed many exciting progress in both theory and practice of active learning. For example, Tong and Koller [TK01] considers actively learning a linear classifier for text classification. The support vector machine based learning algorithm selects the next example to perform label query based on its distance to the current linear separator. It is

empirically shown in [TK01] that, in various tasks, given the same label budget, the error rate of the classifier learned by the proposed active learning algorithm is substantially lower than that trained by passive learning.

In this thesis, we consider active learning in the PAC model [VC71, Val84, KSS94] (where PAC stands for Probably Approximately Correct). Specifically, the learning algorithm is given abilities to draw random unlabeled examples from the underlying distribution, and has the freedom to perform interactive label queries to the unlabeled examples drawn. Its goal is to get a classifier with a target error rate, with a target success probability, using as few label queries as possible.

To see why active learning helps reduce the label requirement, let us consider the following simple but illustrative example. Suppose the unlabeled distribution is uniform over the $[0,1]$ interval, and there is an unknown threshold $t^* \in (0,1)$ such that the examples on its right (resp. left) is labeled $+1$ (resp. -1). How many examples are needed for passive learning to get a classifier with target error rate ϵ? Intuitively, if no examples are in the interval $[t^* - 2\epsilon, t^* + 2\epsilon]$[1], we cannot hope to make an accurate guess about t^* (by accurate here we mean within precision ϵ). It can be shown by a similar but rigorous argument that the sample size needed for passive learning is at least $\Omega\left(\frac{1}{\epsilon}\right)$.

On the other hand, consider the following binary search style active learning algorithm. First, we randomly draw $n = O\left(\frac{1}{\epsilon}\right)$ unlabeled examples, making sure that there is an example in $(t^* - \frac{\epsilon}{2}, t^*)$ and another example in $(t^*, t^* + \frac{\epsilon}{2})$ with high probability. Next, we query the label of the example furthest to the left and the one furthest to the right. If they are the same, stop. If their labels are different, it must be the case where the leftmost label is -1 and the rightmost label is $+1$. Then we query the label of the median; the key observation is that once the label of the median is revealed, we can substantially reduce our search space for the threshold t^*. If the label is positive then we perform our search within the left half, since we know that the threshold cannot line on the right of the median point. Symmetrically, if the label is negative we perform our search within the right half. By performing the search recursively, eventually we find two neighboring points in the dataset with opposite labels. We now return the midpoint between these two points as the learned threshold. It can be shown that the error rate of the output threshold classifier is at most ϵ. Since we are performing binary search, only $O(\log n) = O\left(\ln \frac{1}{\epsilon}\right)$

[1] For simplicity, let us assume $t^* \in [2\epsilon, 1 - 2\epsilon]$.

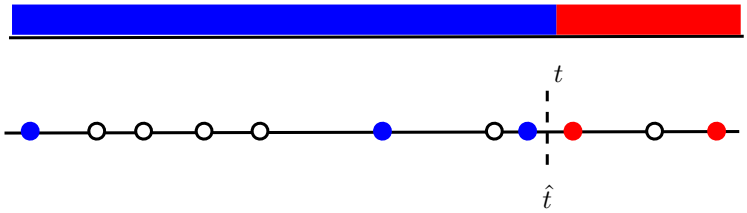

Figure 1.1: Upper: there is an unknown threshold t that separates positive class (red) and negative class (blue). Lower: active learning (binary search) makes $O(\ln \frac{1}{\epsilon})$ label queries, while passive learning needs $\Omega(\frac{1}{\epsilon})$ labeled examples.

Figure 1.2: If the data does not agree with some threshold classifier, using algorithms in realizable case (such as binary search) may converge to locally optimal thresholds such as $\hat{t}$, which has higher error than globally optimal threshold t^*.

labels are queried. Therefore, active learning provides an *exponentially* label saving compared to passive learning. See Figure 1.1 for a pictorial illustration.

Although the above binary search example demonstrates the power of active learning, challenges still remain in the design of active learning algorithms. First, it is unclear how to generalize the above example to learn more complex classifiers, such as linear classifiers, decision trees, etc. Second, in the noisy setting (also known as *nonrealizable* or *agnostic setting*), binary search style algorithms may converge to suboptimal classifiers. Consider an example inspired by [DH08] (See Figure 1.2 for an illustration). Same as the example in the last paragraph, we would like to learn a threshold classifier under the uniform distribution. However, this time the learning problem is not realizable - no threshold classifier perfectly separates the data. Different from the previous example, there is a large cluster of positive examples on the left. This makes the globally optimal threshold t^* lie on the left of the cluster. In this setting, a binary search style algorithm may behave exactly the same as in the previous paragraph, failing to realize that the existence of the large positive cluster. Thus, even with the label budget going to infinity, the error of learned threshold will converge to a locally optimal error value[2]. This issue, namely *statistical inconsistency*, is a key issue we would like to avoid in this thesis. We would like our active learning algorithm to be *statistically consistent*, that is, the learned classifier should converge to the globally optimal one with high probability.

[2]Furthermore, the learned threshold converges to a locally optimal threshold $\hat{t}$.

In addition to feedback from a single labeler, in practice, there are a variety of complex feedback from human that an active learning algorithm can utilize: for instance, giving hints on relevant features [PD17, RMJ06], providing examples from given classes [CTGC05, AP10], returning "I Don't know" for examples that are ambiguous [YCJ16, HLV$^+$16], etc. Moreover, there can be multiple labelers available at the same time with different expertise [DC08]. It is therefore of interest utilizing these types of feedback in a principled way, and understand under what circumstances active learning algorithms can benefit from them. In the first part of this thesis (Chapters 4 and 5), we answer these questions by considering two types of complex feedback: active learning from weak and strong labelers and active learning with a search oracle.

1.1.1 Our Contributions

Active Learning from Weak and Strong Labelers. In Chapter 4, we study the problem of active learning with weak and strong labelers. In addition to having a labeling oracle (strong labeler) which provides expensive but correct responses, we are also given a weak labeler that provides cheap but sometimes wrong label responses. For instance, in medical diagnosis, the oracle can be thought of as a specialist, while the weak labeler can be thought of as a medical resident. In this setting, the goal is to learn a classifier with low error, with as small total cost as possible. We provide an algorithm that works in this setting. Our algorithm has a lower cost complexity than that of active learning using the oracle alone, under certain conditions on the similarity between the oracle and the weak labeler.

Active Learning with a Search Oracle. In Chapter 5, we consider active learning with the help of a new type of oracle - Search oracle. It is motivated by a key challenge in active learning, namely the *rare class* problem. Consider the example in Figure 1.3. Suppose we would like to learn an interval classifier under uniform distribution over $[0,1]$, where the negative examples (blue) lie in an unknown interval $[a^*,b^*]$. We assume that ρ, the length of this interval is $\Omega(\epsilon)$. Therefore, to learn an accurate classifier, it is necessary to locate the negative class, in particular identifying at least one negative example. However, finding a negative example is difficult - no algorithm can find a negative example faster than random search. Therefore, the number of label requests is at least $\Omega\left(\frac{1}{\rho}\right)$. In the setting where $\rho = \Theta(\epsilon)$, the label complexity is $\Omega\left(\frac{1}{\epsilon}\right)$, which is no better than the sample complexity of passive learning. The rare class problem gets more severe when there are *small disjuncts* structures over the examples, that is,

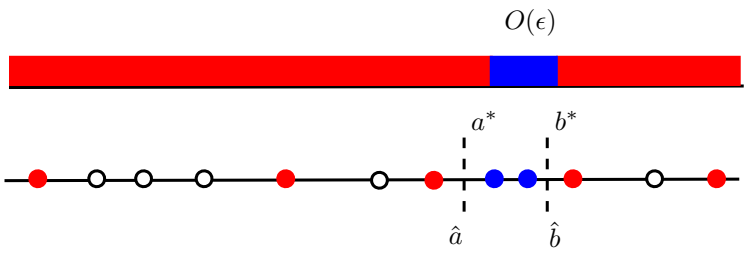

Figure 1.3: Upper: suppose the negative class lies in an interval of mass $\rho = \Theta(\epsilon)$, active learning is unable to achieve label savings compared to passive learning. Lower: if we are given a negative example to begin with, then binary search can be applied to achieve exponential savings on the label complexity.

some classes are spread out into small disjoint "islands". As a concrete example, the negative examples lie in a union of k intervals, each of which is a subset of $[0,1]$. Thus, in order to learn an classifier with target error, the learning algorithm needs to identify all the islands, the task in which active learning does not help.

Nevertheless, these problems can be remedied by "seed" examples. For instance, in the interval learning problem, if we are given a negative example in $[a^*, b^*]$ to begin with, then we can again perform binary search on both sides of the interval to find an approximate boundary, reducing the label complexity to $O\left(\log \frac{1}{\epsilon}\right)$. How can we get such seed examples? This motivates our SEARCH oracle.

Formally, the SEARCH oracle receives a set of classifiers V as input, and returns a counterexample of V. That is, an example on which all classifiers in V disagree with the Bayes classifier h^*. For example, to ask for a example on which h^* predicts -1 (a negative example in the realizable case), we let V be a set that contains only one classifier h that always output $+1$. We show that the SEARCH oracle is useful in the setting of model selection, where we would like to pick a classifier with appropriate complexity to fit the data. By using the additional SEARCH oracle, the number of label requests can be substantially reduced compared to active learning algorithms using label queries alone [Han09].

1.2 Confidence-rated Prediction

In many applications of machine learning, misclassification may be costly, but the learning algorithm has the option to occasionally say "I don't know", i.e. to abstain from prediction. It is therefore essential to develop good algorithms that can trade off misclassification for abstention.

In the second part of this thesis (Chapters 6 and 7), we study the problem of confidence-rated prediction, that is, classification with an abstention option.

For instance, in an online credit card fraud detection system, classifying a new transaction as fraudulent can result in asset loss of customers; however the system has the option to predict "I Don't Know" and pass the transaction onto a human expert. Another example is a medical diagnosis system. When the system is in doubt about a patient's symptom, it has the option to say "I Don't Know" to ask for more examination on the patient, or ask a physician for assistance. Moreover, confidence-rated prediction can be used in stagewise prediction problems [TS13, MP17]. In this setting, the prediction process is broken into multiple stages. At each stage, the algorithm has two options: making a prediction, or go to the next stage to gather more information about the example at the expense of incurring some cost. The goal of the algorithm is predict accurately with a low cost. Hence it needs to tradeoff between total costs and prediction accuracy. Here, one can train an confidence-rated predictor for each stage, and use them to decide between the two options available.

The performance of a confidence-rated predictor is measured by two metrics – the *mistake*, or the *error*, the fraction of examples on which it outputs a wrong label, and the *abstention*, or the fraction of examples on which it abstains. Low error and low abstention are desirable. However, the error of a confidence-rated predictor typically grows with decreasing abstention, and thus there is a tradeoff between the two. The goal of confidence-rated prediction is to develop predictors that have good mistake-abstention tradeoffs.

1.2.1 Our Contributions

In this thesis, we advance the state of the art results by establishing improved algorithms for confidence rated prediction in both batch and online settings.

Improved Confidence-rated Predction in the Batch Setting. In Chapter 6, we study confidence-rated prediction in the batch setting. We focus on the *transductive setting*, in which a set of unlabeled examples is given to the algorithm in the prediction stage, and the algorithm is asked to output labels or abstentions on all the examples at the same time. We provide a novel confidence-rated prediction algorithm with error-abstention tradeoffs. Our algorithm is fully general, in the sense that it applies to any hypothesis class and any data distribution. We show that our algorithm is *optimal* in the realizable case, in the sense that

any other algorithm that has the same error guarantee will necessarily have the same or higher abstention rate. We provide extensions to the non-realizable case and show empirical evidence that it has a better error-abstention tradeoff over previous approaches.

Optimal Confidence-rated Predction in the Online Setting. In Chapter 7, we consider the problem of online confidence-rated prediction. At time t, the learner is given an example x_t, and is asked to output either the correct label for x_t or abstain from prediction. Then the true label of the instance y_t is revealed, and the process goes on iteratively. We provide a minimax analysis on the mistake-abstention tradeoff in the realizable case, where there is some classifier in a hypothesis class that agrees with all the examples shown. Our key contribution is to define a combinatorial measure of hypothesis classes named Extended Littlestone's Dimension, which characterizes the optimal tradeoff between mistakes and abstentions in this setting. It naturally exploits the underlying structures of hypothesis classes. Using this notion, we develop a minimax algorithm, namely the extended standard optimal algorithm, that achieves the optimal tradeoff in the realizable case. Our algorithm significantly improve over previous works [LLWS11, SZB10]. Moreover, we provide upper bounds in non-realizable settings, which does not appear in prior work.

1.3 Interplays between Active Learning and Confidence-rated Prediction

Intuitively, active learning and confidence-rated prediction are connected in the following sense. If an active learning algorithm decides not to query for the label of an example x, it may be certain about its label. Thus, if we (hypothetically) ask it to make confidence-rated prediction on x, then it is unlikely that it will abstain on x. This heuristic motivates a wide range of active learning algorithms, for instance [LG94, LS06]. However, the heuristic lacks a rigorous theoretical analysis in the literature, with the only exception being [EYW12]. [EYW12] proposes a reduction from realizable active learning to perfect confidence-rated prediction (with zero error guarantees). This naturally motivates the following question: is it possible to build a stronger reduction from active learning to confidence-rated prediction? Possible directions include establishing reductions that works beyond realizable settings, and relaxing the zero-error requirement on confidence-rated prediction in the reduction, etc.

1.3.1 Our Contributions

In Chapter 8, we propose confidence-based active learning, answering the above question. Taking a step beyond [EYW12], we show that active learning can be further reduced to *imperfect confidence-rated prediction*, even in the agnostic setting. Specifically, we show that given any confidence-rated predictor with guaranteed error, we can plug it into our confidence-based active learning framework, getting an algorithm that achieves statistical consistency. Furthermore, if we plug in our novel confidence-rated predictor in Chapter 6, we get a new active learning algorithm that works for general hypothesis classes and data distributions, achieves statistical consistency, and has the state of the art label complexities in various settings. Specifically, its label complexity bound is sharper than known disagreement-based approaches [Han14].

Chapter 2

Related Work

In this chapter, we give a literature review over works related to this thesis. We focus on three fields: active learning, confidence-rated prediction and the connection between these two notions.

2.1 Active Learning

Active Learning with Label Queries. According to [Das11], the two main threads of research on active learning with label queries are exploitation of cluster structure [DH08, UWBD13, KUBD15], and efficient search in hypothesis spaces. In this thesis, we focus on the latter: we are given a hypothesis class $\mathcal{H}$, and the goal is to find an $h \in \mathcal{H}$ that achieves a target excess generalization error, while minimizing the number of label queries. Three main approaches have been studied in this setting: generalized binary search, disagreement-based active learning and margin-based active learning. We review each line of approach respectively.

1. The first and most natural approach is generalized binary search [FSST97, Das04, Das05, Now11, GK11, YCJ16, CHK17, TD17], which was analyzed in the realizable case by [Das05] and in various limited noise settings by [Kää06, Now11, NJC13, YCJ16]. It generalizes the one-dimensional binary search example in Chapter 1 and roughly follows the following process. We keep a set of candidate classifiers V that contains the optimal classifier h^*. Now, for the next example to query, we find the one that best "bisects" the candidate set, where the measure of the candidate set can be described using Bayesian posterior

probability, volume, cardinality, etc. Finding such queries has the benefit that, whichever the revealed label is, the measure of the updated candidate set is guaranteed to be reduced by a large factor. While this approach has the advantage of low label complexity, it is in general statistically inconsistent in the case where there is unknown noise [DH08].

2. The second approach, disagreement-based active learning (DBAL), achieves statistical consistency even with unknown noise. Its main idea is as follows. Same as generalized binary search, based on the examples seen so far, the algorithm maintains a candidate set V_t of classifiers in hypothesis class $\mathcal{H}$ that is guaranteed with high probability to contain h^*. Given a randomly drawn unlabeled example x_t, if all classifiers in V_t agree on its label, then its label is inferred; observe that with high probability, this inferred label is $h^*(x_t)$. Otherwise, x_t is said to be in the *disagreement region* of V_t, and the algorithm queries for its label. V_t is updated based on x_t and its label based on generalization error bounds in statistical learning theory [VC71], and the algorithm continues. Note that by using these sample-based statistics, disagreement-based active learning manages to tolerate unknown noise in general data distributions. [CAL94] provides the first disagreement-based algorithm for the realizable case. [BBL09] provides an agnostic disagreement-based algorithm, which is analyzed in [Han07] using a generic notion named disagreement coefficient. [Fri09, Wan11] provide examples showing non-trivial disagreement coefficient bounds for various learning problems. We remark that the notion of disagreement coefficient is also known as Alexander's capacity function [Ale87] in the empirical process literature, and is used by [RR11] to establish lower bounds for active learning. [Han09, Kol10, HY12] give algorithms that are adaptive to the Tsybakov noise conditions [MT04, Tsy04]. [DHM07, BDL09, BHLZ10, BHK$^+$11, ABDL13, HAH$^+$15] have observed that it is possible to determine if an unlabeled example x_t is in the disagreement region of V_t without explicitly maintaining V_t. This observation has led to practical implementations. Instead, a labeled dataset S_t is maintained; the labels of the examples in S_t are obtained by either querying the oracle or direct inference. To determine whether an x_t lies in the disagreement region of V_t, two constrained empirical risk minimization (ERM) procedures are performed; empirical risk is minimized over S_t while constraining the classifier to output the label of x_t as 1 and -1 respectively. If these two classifiers have similar training errors, then x_t lies in the disagreement region of V_t; otherwise the

algorithm infers a label for x_t that agrees with the label assigned by h^*. Chapter 4 of this thesis will use this idea to maintain implicit candidate sets.

3. The third line of work, margin-based active learning [BBZ07, BL13, ABL14, ABHU15, ABHZ16], achieves a better label complexity than disagreement-based approaches for active learning linear classifiers under the uniform distribution and log-concave distributions. Its main idea is as follows. The algorithm keeps a linear classifier over time, and proceeds in epochs. At each epoch, it samples a few examples from a "low margin" region, that is, the region in which the examples' projection along the current linear classifier are small. Then, based on the newly acquired examples, it retrains its learned linear classifier. When the algorithm terminates, it outputs its learned linear classifier. In [ABL14, ABHU15, ABHZ16], computational efficiency issues in noisy settings are also addressed by combining the above algorithmic framework with several novel procedures, such as hinge loss minimization, polynomial regression, soft outlier removal, etc. As can be seen, a major limitation of margin-based active learning is that they apply only to a restricted setting of hypothesis classes and data distributions, and it is not apparent how to apply it generally.

Our confidence-based active learning algorithm in Chapter 8 combines the advantages of both disagreement-based active learning and margin-based active learning - on one hand, it works for general hypothesis classes and unknown data distributions; on the other hand, it has a lower label complexiy compared to disagreement-based approaches. For more references on the theory and practice of active learning, we refer the reader to the excellent surveys by [Set10, Das11, Han14].

Active Learning with Complex Label Queries. There has been a considerable amount of empirical work on active learning where multiple annotators can provide labels for the unlabeled examples. One line of work assumes a generative model for each annotator's labels. The learning algorithm learns the parameters of the individual labelers, and uses them to decide which labeler to query for each example. [YRFD11, YRF$^+$12, FZL$^+$12] consider separate logistic regression models for each annotator, while [LMW14, LMW15] assume that each annotator's labels are corrupted with a different amount of random classification noise. A second line of work [DC08, IPSW14] that includes Pro-Active Learning, assumes that each labeler is an expert over an unknown subset of categories, and uses data to measure the class-wise expertise in order to optimally place label queries. In general, it is not known under what conditions these algorithms

are statistically consistent, particularly when the modeling assumptions do not strictly hold, and under what conditions they provide label savings over regular active learning. [UBS12], the first theoretical work to consider this problem, consider a model where the weak labeler is more likely to provide incorrect labels in heterogeneous regions of space where similar examples have different labels. Their formalization is orthogonal to ours in Chapter 4 – while theirs is more natural in a non-parametric setting, ours is more natural for fitting classifiers in a hypothesis class. [MCR14] have also considered learning from strong and weak labelers; unlike ours, their work is in the online selective sampling setting, and applies only to linear classifiers and robust regression. [DGS12] study learning from multiple teachers in the online selective sampling setting in a model where different labelers have different regions of expertise. [YCJ16, HLV$^+$16] study settings where the labeling oracle is allowed to refrain from returning a binary label, i.e. to answer "I don't know". Adaptive algorithms that utilize both label and abstention feedback are designed to identify the target classifier.

Active Learning with General Queries. [Ang87, Ang04] study upper and lower bounds of exact learning (where the goal is to identify the underlying classifier h^* from a finite hypothesis class $\mathcal{H}$) with a wide range of queries, such as membership query, equivalence query, subset query, superset queries, etc. Specifically, the membership query model is similar to PAC active learning, but instead allowing an algorithm to synthesize an unlabeled example to query for its label. It has also been analyzed in the context of generalized binary search, see [CN07, Now11, YCJ16, CHK17]. The equivalence query model studies the setting where the learning algorithm is allowed to present to the oracle a classifier and ask for its counterexamples. In this setting, [Lit87] establishes an equivalence between the mistake bound model in online classification and equivalence query model. This immediately implies that the Littlestone's Dimension characterizes the minimax optimal query complexity in this model. The partial equivalence query model [MT89] generalizes the equivalence query model by allowing the learning algorithm to present a binary classifier defined only on a subset of the instance domain $\mathcal{X}$ and ask for its counterexamples on that subset. [Han06] considers a general setting of exact learning, where the algorithm is allowed to choose from a collection of queries of varying costs. The work of [Han06] reduces the problem to a minimax game and provides a greedy algorithm that achieves a cost complexity that is at most a factor of $\log|\mathcal{H}|$ worse than optimal cost complexity, where $\mathcal{H}$ is the hypothesis class. [BH12] considers a new type of query, namely class-conditional query (CCQ), motivated

by computer vision applications [CTGC05]. When making a CCQ, the algorithm presents to the oracle a set of unlabeled examples U and a class label c, and ask the oracle to return a example from U in class c. In Chapter 5 of this thesis, motivated by practical applications [AP10, SCL$^+$14], we study a new type of queries named Search. Search can be viewed as a special case of partial equivalence queries, and can be simulated by CCQ.

Most of these works study active learning by interacting with a single oracle only, with the exception of [Han06, XZM$^+$17, KLMZ17]. [Han06] considers an example where the learning algorithm is allowed to combine label queries with CCQ, and shows that its algorithm can get a low cost complexity in this setting. [XZM$^+$17, KLMZ17] study combining label queries with pairwise ranking queries for PAC learning in noiseless and noisy settings. In Chapter 5, we consider the setting where we have both Search and Label oracles available, where Search may have a higher cost compared to Label. We propose algorithms that use Search to assist Label, getting improved Label complexity over Label-only active learning algorithms [Han09].

2.2 Confidence-rated Prediction

While confidence-rated prediction has been empirically studied since the pioneering work of Chow [Cho70], there has been relatively little work that provides guarantees while making minimal generative assumptions. Theoretical work in this area falls roughly into two settings – batch and online.

Confidence-rated Prediction in the Batch Setting. In the batch setting, early theoretical work due to [FMS04] proposes an algorithm that outputs the label predicted by a weighted majority of classifiers, and abstains when agreement among this weighted majority is below a threshold. When the training set size n is large enough, their algorithm achieves error $2\epsilon^* + O(1/n^{1/2-c})$ and abstention $5\epsilon^* + O(\sqrt{\ln|\mathcal{H}|}/n^{1/2-c})$ where ϵ is the minimum error of any hypothesis in the class $\mathcal{H}$, and c is a constant. In contrast, our goal in Chapter 6 is to provide error guarantees of the form $\epsilon^* + \eta$ for any η, and minimize abstention under these guarantees. A second line of work [HW06, BW08, YW10] considers confidence-rated prediction by thresholding a real valued function that belongs to a fixed function class. They show consistency when the prediction function and the associated abstention threshold is learned from data by an ERM procedure. [KKM12, KT14] study a related problem called *reliable learning*, and gives a predictor that achieves low error at the expense of abstentions. The abstention region in [KKM12] is defined as

the disagreement region of two classifiers, one being positive reliable and the other being negative reliable, using the notations of [KKM12]. In contrast, our abstention regions in Chapter 6 can be more general. [Bal16] considers the problem in transductive setting, where the goal is to make aggregated predictions with abstention based on an ensemble of classifiers, where error upper bounds on individual classifiers are known. [CDM16b, CDM16a] considers the setting where the abstention costs are example-dependent, and proposes a convex loss minimization framework to tackle this problem.

A recent learning-theoretic treatment of confidence-rated prediction in the batch setting has been proposed by El-Yaniv and Wiener [EYW10, EYW11]. [EYW10] addresses the realizable case and provides a predictor which abstains on examples which lie in the disagreement region of the version space; this predictor achieves zero error, and its abstention rate is equal to the probability mass of the disagreement region. [EYW11] shows how to extend this algorithm to the non-realizable case so as to get zero error with respect to the best hypothesis in the class. Motivated by [DHM07, BHLZ10], they provide a confidence-rated predictor for the non-realizable case that works for any abstention rate. Their predictor does not offer error guarantees (other than for zero error), but has superior empirical performance compared with using the distance from the decision boundary in SVM [GWBV02, Muk03]. In contrast, we give an algorithm in Chapter 6 that gives error-abstention tradeoffs for a range of error guarantees $\eta \geq 0$, and show its optimality in the realizable case.

Confidence-rated Prediction in the Online Setting. The first formal framework for online learning with abstentions is the Knows What It Knows (KWIK) model [LLWS11]. [LLWS11] formalizes versions of this model for both classification and regression settings, and provides a classification algorithm that achieves no mistakes and $|\mathcal{H}| - 1$ abstentions when the sequences provided are realizable for a finite hypothesis class $\mathcal{H}$. [SS11] provides algorithms for online regression that apply even when the realizability assumption is relaxed. In [SZB10], the KWIK model is relaxed in the classification setting by allowing the learner to make $\leq k$ mistakes. Their work presents an algorithm that, given a finite hypothesis class $\mathcal{H}$, can make at most k mistakes with at most $(k+1)|\mathcal{H}|^{\frac{1}{k+1}}$ abstentions. [DZ13] extends [SZB10] by providing efficient algorithms for the class of disjunctions. Our work in Chapter 7 provides minimax mistake-abstention tradeoffs that improves over these works in the realizable setting, and moreover addresses the more challenging nonrealizable setting. Finally, another important line of work

for online classification with abstentions is conformal prediction [SV08], which, given a conformity measure R and an error probability measure δ, shows a strategy for constructing confidence sets in an online manner that contain the correct label with probability $1-\delta$. Our framework differs from this line of work in that the conformity measure for us is not specified.

There is a large volume of literature on online classification when no abstentions are allowed. The mistake bound model, initially proposed by [Lit87], considers online binary classification in the realizable case. [Lit87] also introduces the standard optimal algorithm and optimal mistake bound (aka Littlestone's dimension $\text{Ldim}(\mathcal{H})$). There has been much literature on developing algorithms for specific hypothesis classes in the mistake bound model; see [SSBD14, CBL06] for examples. [BPS09] considers online classification (with no abstentions) in the agnostic case; they show that if the hypothesis class $\mathcal{H}$ has finite Littlestone's dimension, then it is possible to design an online prediction algorithm that makes $l + \tilde{O}(\sqrt{\text{Ldim}(\mathcal{H})T} + \text{Ldim}(\mathcal{H}))$ mistakes over T rounds, where l is the minimum error of any hypotheses in $\mathcal{H}$. In follow-up work, [RST10, RSS12, RST15b, RST15a] have developed a rich theory of online learning, and defined complexity measures such as sequential Rademacher complexity, and sequential covering number that characterize the complexity of online learning. However, this theory does not apply to online learning with abstentions.

2.3 Interplay between Active Learning and Confidence-rated Prediction

The confidence-rated prediction algorithm of [EYW10] can be seen as a reminiscent of the classical active learning algorithm of [CAL94], which queries labels of examples that only lie in the disagreement region of the version space, and the notion of zero error with respect to best hypothesis in $\mathcal{H}$ in [EYW11] is similar to those used by [DHM07, Han07].

A formal connection between disagreement-based active learning in realizable case and perfect confidence-rated prediction (with a zero error guarantee) was established by [EYW12]. Specifically, they show that if the perfect confidence-rated predictor has a abstention rate of $O(\frac{\text{polylog}(m)}{m})$, then disagreement-based active learning algorithms [Han14] will have a label complexity of $O(\text{polylog}(\frac{1}{\epsilon}))$. Subsequent work [WHEY15] show that the converse is also true via a sample compression argument, thus proving these two conditions are equivalent. [GEY17] further

show a similar equivalence in the non-realizable case.

An imperfect confidence-rated predictor is implicitly proposed in the literature on margin-based active learning [BBZ07, BL13]. Specifically, the confidence-rated predictor abstains on an example if its projection on the current linear classifier is small, i.e. it has a small "margin". The algorithm achieves a lower label complexity compared to disagreement-based algorithms in the case when the underlying distribution over x is uniform or log-concave. However, it does not apply to general hypothesis classes and data distributions. Nevertheless, it provides evidence that a deeper connection between active learning and confidence-rated prediction might be possible via imperfect confidence-rated predictions. Indeed we establish such a connection in Chapter 8.

Chapter 3

Preliminaries

In this chapter, we introduce some definitions that will be used throughout the thesis.

3.1 Basic Notations

Define the indicator function $\mathbb{1}(\cdot)$ as follows: $\mathbb{1}(A) = 1$ if predicate A is true, $\mathbb{1}(A) = 0$ if A is false. We call function $f(\cdot) = \tilde{O}(g(\cdot))$ if $f(\cdot) = O(g(\cdot) \ln g(\cdot))$. For a set A, define $A^* := \cup_{k=0}^{\infty} A^k$ as the set of sequences whose elements are from A.

3.2 PAC Learning

The PAC learning model [VC71, Val84, KSS94] is a standard model in statistical learning theory. In this model, examples belong to an instance space $\mathcal{X}$, and their labels lie in a label space $\mathcal{Y} = \{-1, 1\}$; labeled examples are drawn from an underlying data distribution D on $\mathcal{X} \times \mathcal{Y}$. We use $D_{\mathcal{X}}$ or U to denote the unlabeled distribution, that is, the marginal distribution of D on $\mathcal{X}$. We use $D_{Y|X}$ to denote the conditional distribution on $Y|X = x$ induced by D. A classifier, or hypothesis h is a mapping from $\mathcal{X}$ to $\mathcal{Y}$. A hypothesis class (or hypothesis space) $\mathcal{H}$ is a set of classifiers.

Given a hypothesis class $\mathcal{H}$ of VC dimension d, the error of any $h \in \mathcal{H}$ with respect to a distribution Π over $\mathcal{X} \times \mathcal{Y}$ is defined as $\mathrm{err}_{\Pi}(h) = \mathbb{P}_{(x,y)\sim\Pi}[h(x) \neq y]$. We will also use $\mathrm{err}(h, \Pi)$ to denote $\mathrm{err}_{\Pi}(h)$, and if $\Pi = D$, we will simply write $\mathrm{err}(h)$. We define the optimal classifier in

$\mathcal{H}$ with respect to Π as $h^*(\Pi) = \arg\min_{h\in\mathcal{H}} \mathrm{err}_\Pi(h)$, and optimal error as $\nu^*(\Pi) = \mathrm{err}_\Pi(h^*(\Pi))$. We will also use h^* to denote $h^*(D)$, and use ν^* to denote $\nu^*(D)$. When $\nu^* = 0$, we are said to be in the *realizable case*; in the more general *agnostic* case, we make no assumptions on the data distribution D, and thus ν^* can be positive. For a set of examples S, we abuse notation and use S to also denote the uniform distribution over its elements. We say that classifier h is consistent with dataset S if for every example (x,y) in S, $h(x)$ is equal to y. We define probability over Π as $\mathbb{P}_\Pi[\cdot] := \mathbb{P}_{(x,y)\sim\Pi}[\cdot]$, and expectation over Π as $\mathbb{E}_\Pi[\cdot] := \mathbb{E}_{(x,y)\sim\Pi}[\cdot]$. The excess error of classifier h in $\mathcal{H}$ is defined as the difference between its error and the optimal error in $\mathcal{H}$, that is $\mathrm{err}(h) - \nu^*$.

The goal of PAC learning is to develop a learning algorithm $\mathcal{A}$, such that it receives learning parameters $\epsilon, \delta \in (0,1)$ as input, draws examples from D, and returns a classifier h in $\mathcal{H}$ with excess error ϵ with probability $1-\delta$. As discussed in Chapter 1, this basic model without label queries is called PAC passive learning. A key quantity of interest in PAC passive learning is the *sample complexity*, that is, how many examples are needed to achieve the PAC learning goal. This is made precise in the following definition.

Definition 3.1 (Sample Complexity in PAC Passive Learning)**.** *A PAC passive learning algorithm $\mathcal{A}$ is said to (ϵ,δ)-learn a hypothesis class $\mathcal{H}$ with sample complexity $n(\epsilon,\delta)$, if it draws $n(\epsilon,\delta)$ random examples iid from D, and outputs a classifier $\hat{h} \in \mathcal{H}$ such that with probability at least $1-\delta$, $\mathrm{err}_D(\hat{h}) \le \nu^*(D) + \epsilon$.*

3.3 Active Learning

In Chapters 4, 5, and 8, we study (variants of) active learning in the PAC setting. In this setting, the learner does not draw labeled examples directly from D. Instead, it has access to examples through two oracles – an example oracle EX which returns an unlabeled example $x \in \mathcal{X}$ drawn from $D_\mathcal{X}$ and a labeling oracle Label which returns the label y of an input $x \in \mathcal{X}$ drawn from $D_{Y|X}$. Based on the interaction between the two oracles, the algorithm returns its learned classifier as output.

Similar to the goal of PAC passive learning, the goal of PAC active learning is to develop a learning algorithm $\mathcal{A}$, such that it receives learning parameters $\epsilon, \delta \in (0,1)$ as input, query the oracles EX and Label, and returns a classifier h in $\mathcal{H}$ with excess error ϵ with probability $1-\delta$.

A key quantity of interest in active learning is *label complexity*, that is, how many label queries are needed to learn a classifier with excess error ϵ with probability $1-\delta$. This is made precise in the following definition. Note that in active learning, we usually do not optimize for the number of unlabeled examples drawn. However, as we will see in subsequent chapters, the number of unlabeled examples is usually no worse than the sample complexity of passive learning.

Definition 3.2 (Label Complexity in PAC Active Learning)**.** *A PAC active learning algorithm $\mathcal{A}$ is said to (ϵ,δ)-learn a hypothesis class $\mathcal{H}$ with label complexity $m(\epsilon,\delta)$, if it draws a set of unlabeled examples iid from $D_{\mathcal{X}}$ through oracle EX, performs $m(\epsilon,\delta)$ queries to oracle* Label*, and outputs a classifier $\hat{h}\in\mathcal{H}$ such that with probability at least $1-\delta$, $\mathrm{err}_D(\hat{h})\leq\nu^*(D)+\epsilon$.*

In the sample complexity and label complexity bounds of PAC passive and active learning, we use the following convention: the $\tilde{O}$ notation sometimes hides $\ln\frac{1}{\delta}$ factors.

Previous approaches to agnostic active learning have frequently used the notion of *disagreements* [Han14]. The disagreement between two hypotheses h_1 and h_2 with respect to a data distribution Π is the fraction of examples according to Π to which h_1 and h_2 assign different labels; formally, $\rho_\Pi(h_1,h_2)=\mathbb{P}_\Pi[h_1(x)\neq h_2(x)]$. Observe that a data distribution Π induces a pseudo-metric ρ_Π on the elements of $\mathcal{H}$; this is called the disagreement metric. For any r and any $h\in\mathcal{H}$, define $\mathrm{B}_\Pi(h,r)$ to be the disagreement ball of radius r around h with respect to the data distribution Π. Formally, $\mathrm{B}_\Pi(h,r)=\{h'\in\mathcal{H}:\rho_\Pi(h,h')\leq r\}$. We will also use ρ to denote ρ_D, and use B to denote B_D.

Define the *region of disagreement* of a set of hypotheses V as $\mathrm{Dis}(V):=\{x\in\mathcal{X}:\exists h,h'\in V \text{ s.t. } h(x)\neq h'(x)\}$. We denote the disagreement region of a disagreement ball of radius r centered around h^* by $\Delta(r):=\mathrm{Dis}(\mathrm{B}(h^*,r))$. A key quantity, namely disagreement coefficient has been defined by [Han09] to analyze disagreement-based active learning algorithms. We define it formally here.

Definition 3.3 (Disagreement Coefficient)**.** *The* disagreement coefficient of V at scale r *is*

$$\theta_V(r):=\sup_{h\in V,r'\geq r}\frac{\mathbb{P}_{D_{\mathcal{X}}}[\mathrm{Dis}(\mathrm{B}(h,r'))]}{r'}.$$

Intuitively, this measures the rate of shrinkage of the disagreement region with the radius of the ball $\mathrm{B}(h,r)$ for any h in V. If $V=\mathcal{H}$ is the whole hypothesis class, we use $\theta(r)$ to denote

$\theta_V(r)$.

For minor technical reasons, we assume the hypothesis space is "dense", that is, for all $r > 0$, $\sup_{h \in B(h^*,r)} \rho(h,h^*) = r$. We will call this the "denseness assumption".

In the active learning algorithms in Chapters 4, 5, and 8, we will maintain a candidate set of classifiers, either explicitly or implicitly. In the realizable case, we use version spaces [Mit82] as candidate sets.

Definition 3.4 (Version Space)**.** *The version space with respect to a set S of labeled examples is the set of all $h \in \mathcal{H}$ such that $h(x) = y$ for all $(x,y) \in S$.*

In the nonrealizable case, the version space can be empty due to the non-separability of the data. Instead, we use a probabilistic notion, namely $(1-\alpha)$-confidence set as the candidate set of classifiers.

Definition 3.5 (Confidence Set)**.** *Given a set S of n labeled examples, let $C(S) \subseteq \mathcal{H}$ be a function of S; $C(S)$ is said to be a $(1-\alpha)$-confidence set for the true risk minimizer if for all data distributions Δ over $\mathcal{X} \times \mathcal{Y}$,*

$$\mathbb{P}_{S \sim \Delta^n}[h^*(\Delta) \in C(S)] \geq 1 - \alpha,$$

Recall that $h^*(\Delta) = \arg\min_{h \in \mathcal{H}} \mathrm{err}_\Delta(h)$.

3.4 Confidence-rated Prediction in the Batch Setting

In Chapter 6, we study confidence-rated prediction in the batch setting. Formally, a confidence-rated predictor is a function $P : \mathcal{X} \to \{-1, +1, \perp\}$ which takes an input from a domain $\mathcal{X}$, and either outputs a label in $\{-1,1\}$ or $\perp$ to indicate "I don't know". Our goal is to develop algorithms, which, given a labeled training dataset drawn from an underlying distribution D, builds a confidence-rated predictor based on the training data.

The performance of a confidence-rated predictor is measured by two metrics – its error and its abstention. Usually there is tradeoff between these two metrics - if we are allowed more errors, then the abstention rate can be reduced.

Definition 3.6 (Error of a Confidence-rated Predictor)**.** *The* error of a confidence-rated predictor P with respect to the true labels*, denoted by* $\mathrm{err}_D(P)$ *is the fraction of examples from D on which*

P predicts 1 *when the true label is* -1 *and vice versa. Formally,*

$$\mathrm{err}_D(P) = \mathbb{P}_D[P(x) = -y].$$

Definition 3.7 (Abstention of a Confidence-rated Predictor)**.** *The* abstention $\mathrm{abs}_D(P)$ *of a confidence-rated predictor P is the fraction of examples from the underlying data distribution D on which P abstains (that is, it predicts $\perp$). Formally,*

$$\mathrm{abs}_D(P) = \mathbb{P}_D[P(x) = \perp].$$

3.5 Confidence-rated Prediction in the Online Setting

In Chapter 7, we consider the setup of online learning, where examples arrive sequentially and may be adversarial. We focus on a specialized setting, i.e. online confidence-rated prediction. In this setting, the interaction between the learner and the adversary can be described as follows. For rounds $t = 1, 2, \ldots, n$:

1. The adversary presents an instance x_t from instance space $\mathcal{X}$.
2. The learner makes a prediction $\hat{y}_t$ in $\{-1, +1, \perp\}$.
3. The adversary reveals the true label y_t in $\{-1, +1\}$.

Specifically, we focus on the deterministic prediction setting, where the learner is only allowed to output $\hat{y}_t$ deterministically. We remark that a natural randomized prediction model can also be considered, although it will not be covered in this thesis.

Note that if $\hat{y}_t$ is only allowed to be chosen from $\{-1, +1\}$, then the setting comes down to online classification, in which the mistake bound model is proposed [Lit87]. In the mistake bound model, the performance of the learner is measured by $\sum_{t=1}^{n} I(\hat{y}_t = -y_t)$, its total number of mistakes.

In contrast to mistake bound model, in the setting of online confidence-rated prediction, the performance of the learner is measured by two quantities: first, its total number of mistakes $\sum_{t=1}^{n} \mathbb{1}(\hat{y}_t = -y_t)$ and second, its total number of abstentions $\sum_{t=1}^{n} \mathbb{1}(\hat{y}_t = \perp)$. We remark these two performance metrics are similar to those in the batch setting, except for that we are counting their absolute numbers instead of their fractions. The goal of the learner is to minimize its total

mistakes and its total abstentions simultaneously. Similar to the batch setting, usually there is a tradeoff between these two measures - if the learning algorithm is allowed more mistakes, the number of abstentions can be reduced and vice versa.

For deterministic learners, it is always suboptimal for the adversary to present an example on which the learner predicts correctly, as this will only impose additional constraints on examples shown in the future without changing the number of mistakes and abstentions made. A round t is called *nontrivial* if the learner makes a mistake or abstains on that round.

We consider the k-SZB model, named after its authors [SZB10]. As we will see, it is an interpolation between the mistake bound model [Lit87] and the KWIK (Know What It Knows) model [LLWS11], and is defined as follows.

Definition 3.8 (The k-SZB model)**.** *An online learning algorithm $\mathcal{A}$ achieves a (k,m)-SZB bound with respect to a set of sequences $\mathcal{S} \subseteq (\mathcal{X} \times \{-1,+1\})^*$, if and only if for any adversary that presents sequences $S = ((x_1,y_1),\ldots,(x_n,y_n))$ in $\mathcal{S}$, $\mathcal{A}$'s prediction $\hat{y}_1, \ldots, \hat{y}_n \in \{-1,+1,\perp\}$ satisfies*

$$\sum_{t=1}^{n} \mathbb{1}(\hat{y}_t = -y_t) \leq k,$$

$$\sum_{t=1}^{n} (\mathbb{1}(\hat{y}_t = -y_t) + \mathbb{1}(\hat{y}_t = \perp)) \leq m.$$

In other words, the algorithm guarantees that the number of mistakes is at most k, and number of nontrivial rounds (where the algorithm makes a mistake or abstains) is at most d. When $k = 0$, this is exactly the KWIK model [LLWS11]. When the learner not allowed to abstain, the number of mistakes is the same as the number of nontrivial rounds. In this case, the model comes down to the mistake bound model [Lit87].

Chapter 4

Active Learning from Weak and Strong Labelers

4.1 Introduction

In this chapter, we study the problem of active learning from weak and strong labelers. Specifically, in addition to unlabeled data and the usual labeling oracle O in standard active learning [1], we have an extra weak labeler W. The labeling oracle is a *gold standard* – an expert on the problem domain – and it provides high quality but expensive labels. The weak labeler is cheap, but may provide incorrect labels on some inputs. An example is learning to classify medical images where either expensive labels may be obtained from a physician (oracle), or cheaper but occasionally incorrect labels may be obtained from a medical resident (weak labeler). Our goal is to learn a classifier in a hypothesis class whose error with respect to the data labeled by the oracle is low, while exploiting the weak labeler to reduce the number of queries made to this oracle. Observe that in our model the weak labeler can be incorrect anywhere, and does not necessarily provide uniformly noisy labels everywhere, as was assumed by some previous works [CKW05, SCS15].

A plausible approach in this framework is to learn a *difference classifier* to predict where the weak labeler differs from the oracle, and then use a standard active learning algorithm

[1] We use O as LABEL in this chapter for notational brevity.

which queries the weak labeler when this difference classifier predicts agreement. Our first key observation is that this approach is *statistically inconsistent*; false negative errors (that predict no difference when O and W differ) lead to biased annotations for the target classification task. We address this problem by learning instead a *cost-sensitive difference classifier* that ensures that false negative errors rarely occur. Our second key observation is that as existing active learning algorithms usually query labels in localized regions of space, it is sufficient to train the difference classifier restricted to this region and still maintain consistency. This process leads to significant label savings. Combining these two ideas, we get an algorithm that is provably statistically consistent and that works under the assumption that there is a good difference classifier with low false negative error.

We analyze the label complexity of our algorithm as measured by the number of label requests to the labeling oracle. In general we cannot expect any statistically consistent algorithm to provide label savings under all circumstances, and indeed our worst case asymptotic label complexity is the same as that of active learning using the oracle alone. Our analysis characterizes when we can achieve label savings, and we show that this happens for example if the weak labeler agrees with the labeling oracle for some fraction of the examples close to the decision boundary. Moreover, when the target classification task is agnostic, the number of labels required to learn the difference classifier is of a lower order than the number of labels required for active learning; thus in realistic cases, learning the difference classifier adds only a small overhead to the total label requirement, and overall we get label savings over using the oracle alone.

4.2 Preliminaries

The Model. We begin with a general framework for actively learning from weak and strong labelers. Recall the notations in Chapter 3, in standard active learning, we are given unlabeled data drawn from a distribution U over an input space $\mathcal{X}$, a label space $\mathcal{Y} = \{-1, 1\}$, a hypothesis class $\mathcal{H}$, and a labeling oracle O to which we can make interactive queries.

In our setting, we additionally have access to a weak labeling oracle W which we can query interactively. Querying W is significantly cheaper than querying O; however, querying W generates a label y_W drawn from a conditional distribution $\mathbb{P}_W[y_W|x]$ which is not the same as the conditional distribution $\mathbb{P}_O[y_O|x]$ of O. We use the notation $\mathcal{D}$ to denote the joint distribution

over examples and labels from O and W:

$$\mathbb{P}_{\mathcal{D}}[x, y_O, y_W] = \mathbb{P}_U[x]\mathbb{P}_O[y_O|x]\mathbb{P}_W[y_W|x]$$

Recall from Chapter 3 that D is the data distribution over labeled examples such that: $\mathbb{P}_D[x, y] = \mathbb{P}_U[x]\mathbb{P}_O[y|x]$. Our goal is to learn a classifier h in the hypothesis class $\mathcal{H}$ such that with probability $\geq 1 - \delta$ over the sample, we have: $\text{err}(h) \leq \text{err}(h^*) + \epsilon$, while making as few (interactive) queries to O as possible.

Observe that in this model W may disagree with the oracle O *anywhere* in the input space; this is unlike previous frameworks [CKW05, SCS15] where labels assigned by the weak labeler are corrupted by random classification noise with a higher variance than the labeling oracle. We believe this feature makes our model more realistic.

Second, unlike [UBS12], mistakes made by the weak labeler *do not have to be close to the decision boundary*. This keeps the model general and simple, and allows greater flexibility to weak labelers. Our analysis shows that if W is largely incorrect close to the decision boundary, then our algorithm will automatically make more queries to O in its later stages.

Finally note that O is allowed to be *non-realizable* with respect to the target hypothesis class $\mathcal{H}$.

4.3 Algorithm

Our main algorithm is a standard single-annotator DBAL algorithm with a major modification: when the DBAL algorithm makes a label query, we use an extra sub-routine to decide whether this query should be made to the oracle or the weak labeler, and make it accordingly. How do we make this decision? We try to predict if weak labeler differs from the oracle on this example; if so, query the oracle, otherwise, query the weak labeler.

Key Idea 1: Cost Sensitive Difference Classifier. How do we predict if the weak labeler differs from the oracle? A plausible approach is to learn a *difference classifier* h^{df} in a hypothesis class $\mathcal{H}^{df}$ to determine if there is a difference. Our first key observation is when the region where O and W differ cannot be perfectly modeled by $\mathcal{H}^{df}$, the resulting active learning algorithm is *statistically inconsistent*. Any false negative errors (that is, incorrectly predicting no difference) made by difference classifier leads to biased annotation for the target classification

task, which in turn leads to inconsistency. We address this problem by instead learning a *cost-sensitive difference classifier* and we assume that a classifier with low false negative error exists in $\mathcal{H}^{df}$. While training, we constrain the false negative error of the difference classifier to be low, and minimize the number of predicted positives (or disagreements between W and O) subject to this constraint. This ensures that the annotated data used by the active learning algorithm has diminishing bias, thus ensuring consistency.

Key Idea 2: Localized Difference Classifier Training. Unfortunately, even with cost-sensitive training, directly learning a difference classifier accurately is expensive. If d' is the VC-dimension of the difference hypothesis class $\mathcal{H}^{df}$, to learn a target classifier to excess error ϵ, we need a difference classifier with false negative error $O(\epsilon)$, which, from standard generalization theory, requires $\tilde{O}(d'/\epsilon)$ labels [BB05, Sim14]! Our second key observation is that we can save on labels by training the difference classifier in a localized manner – because the DBAL algorithm that builds the target classifier *only makes label queries* in the disagreement region of the current confidence set for h^*. Therefore we train the difference classifier only on this region and still maintain consistency. Additionally this provides label savings because while training the target classifier to excess error ϵ, we need to train a difference classifier with only $\tilde{O}(d'\phi_k/\epsilon)$ labels where ϕ_k is the probability mass of this disagreement region. The localized training process leads to an additional technical challenge: as the confidence set for h^* is updated, its disagreement region changes. We address this through an epoch-based DBAL algorithm, where the confidence set is updated and a fresh difference classifier is trained in each epoch.

Main Algorithm. Our main algorithm (Algorithm 4.1) combines these two key ideas, and like [DHM07, BDL09, BHLZ10, BHK$^+$11, ABDL13, HAH$^+$15], implicitly maintains the $(1-\delta)$-confidence set for h^* by through a labeled dataset $\hat{S}_k$. In epoch k, the target excess error is $\epsilon_k \approx \frac{1}{2^k}$, and the goal of Algorithm 4.1 is to generate a labeled dataset $\hat{S}_k$ that implicitly represents a $(1-\delta_k)$-confidence set on h^*. Additionally, $\hat{S}_k$ has the property that the empirical risk minimizer over it has excess error $\leq \epsilon_k$.

A naive way to generate such an $\hat{S}_k$ is by drawing $\tilde{O}(d/\epsilon_k^2)$ labeled examples, where d is the VC dimension of $\mathcal{H}$. Our goal, however, is to generate $\hat{S}_k$ using a much smaller number of label queries, which is accomplished by Algorithm 4.4. This is done in two ways. First, like standard DBAL, we infer the label of any x that lies *outside* the disagreement region of the current confidence set for h^*. Algorithm 4.3 identifies whether an x lies in this region. Second,

for any x in the disagreement region, we determine whether O and W agree on x using a difference classifier; if there is agreement, we query W, else we query O. The difference classifier used to determine agreement is retrained in the beginning of each epoch by Algorithm 4.2, which ensures that the annotation has low bias.

The algorithms use a constrained ERM procedure Cons-Learn. Given a hypothesis class H, a labeled dataset S and a set of constraining examples C, Cons-Learn$_H(C,S)$ returns a classifier in H that minimizes the empirical error on S subject to $h(x_i) = y_i$ for each $(x_i, y_i) \in C$.

Identifying the Disagreement Region. Algorithm 4.3 identifies if an unlabeled example x lies in the disagreement region of the current $(1-\delta)$-confidence set for h^*; recall that this confidence set is implicitly maintained through $\hat{S}_k$. The identification is based on two ERM queries. Let $\hat{h}$ be the empirical risk minimizer on the current labeled dataset $\hat{S}_{k-1}$, and $\hat{h}'$ be the empirical risk minimizer on $\hat{S}_{k-1}$ under the constraint that $\hat{h}'(x) = -\hat{h}(x)$. If the training errors of $\hat{h}$ and $\hat{h}'$ are very different, then, all classifiers with training error close to that of $\hat{h}$ assign the same label to x, and x lies outside the current disagreement region.

Training the Difference Classifier. Algorithm 4.2 trains a difference classifier on a random set of examples which lies in the disagreement region of the current confidence set for h^*. The training process is cost-sensitive, and is similar to [KKM12, KT14, BB05, Sim14]. A hard bound is imposed on the false-negative error, which translates to a bound on the annotation bias for the target task. The number of positives (i.e., the number of examples where W and O differ) is minimized subject to this constraint; this amounts to (approximately) minimizing the fraction of queries made to O.

The number of labeled examples used in training is large enough to ensure false negative error $O(\epsilon_k/\phi_k)$ over the disagreement region of the current confidence set; here ϕ_k is the probability mass of this disagreement region under U. This ensures that the overall annotation bias introduced by this procedure in the target task is at most $O(\epsilon_k)$. As ϕ_k is small and typically diminishes with k, this requires less labels than training the difference classifier globally which would have required $\tilde{O}(d'/\epsilon_k)$ queries to O.

Adaptive Active Learning using the Difference Classifier. Finally, Algorithm 4.4 is our main active learning procedure, which generates a labeled dataset $\hat{S}_k$ that is implicitly

[2]Note that if in Algorithm 4.5, the upper confidence bound of $\mathbb{P}_{x\sim U}(\text{In-Disagr-Region}(\hat{T}, \frac{3\epsilon}{2}, x) = 1)$ is lower than $\epsilon/64$, then we can halt Algorithm 4.2 and return an arbitrary h^{df} in $\mathcal{H}^{df}$. Using this h^{df} will still guarantee the correctness of Algorithm 4.1.

Algorithm 4.1 Active Learning Algorithm from Weak and Strong Labelers

1: Input: Unlabeled distribution U, target excess error ϵ, confidence δ, labeling oracle O, weak oracle W, hypothesis class $\mathcal{H}$, hypothesis class for difference classifier $\mathcal{H}^{df}$.
2: Output: Classifier $\hat{h}$ in $\mathcal{H}$.
3: Initialize: initial error $\epsilon_0 = 1$, confidence $\delta_0 = \delta/4$. Total number of epochs $k_0 = \lceil \log \frac{1}{\epsilon} \rceil$.
4: Initial number of examples $n_0 = O(\frac{1}{\epsilon_0^2}(d \ln \frac{1}{\epsilon_0^2} + \ln \frac{1}{\delta_0}))$.
5: Draw a fresh sample and query O for its labels $\hat{S}_0 = \{(x_1, y_1), \ldots, (x_{n_0}, y_{n_0})\}$. Let $\sigma_0 = \sigma(n_0, \delta_0)$.
6: **for** $k = 1, 2, \ldots, k_0$ **do**
7: Set target excess error $\epsilon_k = 2^{-k}$, confidence $\delta_k = \delta/4(k+1)^2$.
8: *# Train Difference Classifier*
9: $\hat{h}_k^{df} \leftarrow$ Call Algorithm 4.2 with inputs unlabeled distribution U, oracles W and O, target excess error ϵ_k, confidence $\delta_k/2$, previously labeled dataset $\hat{S}_{k-1}$.
10: *# Adaptive Active Learning using Difference Classifier*
11: $\sigma_k, \hat{S}_k \leftarrow$ Call Algorithm 4.4 with inputs unlabeled distribution U, oracles W and O, difference classifier $\hat{h}_k^{df}$, target excess error ϵ_k, confidence $\delta_k/2$, previously labeled dataset $\hat{S}_{k-1}$.
12: **end for**
13: **return** $\hat{h} \leftarrow$ Cons-Learn$_{\mathcal{H}}(\emptyset, \hat{S}_{k_0})$.

used to maintain a tighter $(1-\delta)$-confidence set for h^*. Specifically, Algorithm 4.4 generates a $\hat{S}_k$ such that the set V_k defined as:

$$V_k = \{h : \text{err}(h, \hat{S}_k) - \min_{\hat{h}_k \in \mathcal{H}} \text{err}(\hat{h}_k, \hat{S}_k) \leq 3\epsilon_k/4\}$$

has the property that:

$$\{h : \text{err}_D(h) - \text{err}_D(h^*) \leq \epsilon_k/2\} \subseteq V_k \subseteq \{h : \text{err}_D(h) - \text{err}_D(h^*) \leq \epsilon_k\}$$

This is achieved by labeling, through inference or query, a large enough sample of unlabeled data drawn from U. Labels are obtained from three sources - direct inference (if x lies outside the disagreement region as identified by Algorithm 4.3), querying O (if the difference classifier predicts a difference), and querying W. How large should the sample be to reach the target excess error? If $\text{err}_D(h^*) = \nu$, then achieving an excess error of ϵ requires $\tilde{O}(d\nu/\epsilon_k^2)$ samples, where d is the VC dimension of the hypothesis class. As ν is unknown in advance, we use a doubling procedure in lines 4-14 to iteratively determine the sample size.

Algorithm 4.2 Training Algorithm for Difference Classifier

Input: Unlabeled distribution U, oracles W and O, target error ε, hypothesis class $\mathcal{H}^{df}$, confidence δ, previous labeled dataset $\hat{T}$.
Output: Difference classifier $\hat{h}^{df}$.
Let $\hat{p}$ be an estimate of $\mathbb{P}_{x\sim U}(\text{IN-DISAGR-REGION}(\hat{T}, \frac{3\epsilon}{2}, x) = 1)$, obtained by calling Algorithm 4.5(deferred to the Section) with failure probability $\delta/3$. [2]
Let $U' = \emptyset$, $i = 1$, and

$$m = \frac{64 \cdot 1024\hat{p}}{\varepsilon}(d' \ln \frac{512 \cdot 1024\hat{p}}{\varepsilon} + \ln \frac{72}{\delta}) \tag{4.1}$$

repeat
  Draw an example x_i from U.
  if IN-DISAGR-REGION$(\hat{T}, \frac{3\epsilon}{2}, x_i) = 1$ **then** # x_i is in the disagreement region
    query both W and O for labels to get $y_{i,W}$ and $y_{i,O}$.
  end if
  $U' = U' \cup \{(x_i, y_{i,O}, y_{i,W})\}$
  $i = i + 1$
until $|U'| = m$
Learn a classifier $\hat{h}^{df} \in \mathcal{H}^{df}$ based on the following empirical risk minimizer:

$$\hat{h}^{df} = \arg\min_{h^{df} \in \mathcal{H}^{df}} \sum_{i=1}^{m} 1(h^{df}(x_i) = +1), \text{ s.t. } \sum_{i=1}^{m} 1(h^{df}(x_i) = -1 \wedge y_{i,O} \neq y_{i,W}) \leq m\varepsilon/256\hat{p} \tag{4.2}$$

return $\hat{h}^{df}$.

4.4 Performance Guarantees

We now examine the performance of our algorithm, which is measured by the number of label queries made to the oracle O. Additionally we require our algorithm to be statistically consistent, which means that the true error of the output classifier should converge to the true error of the best classifier in $\mathcal{H}$ on the data distribution D.

Since our framework is very general, we cannot expect any statistically consistent algorithm to achieve label savings over using O alone under all circumstances. For example, if labels provided by W are completely unrelated to O, no algorithm will achieve both consistency and label savings. We next provide an assumption under which Algorithm 4.1 works and yields label savings.

Assumption. The following assumption states that difference hypothesis class contains a good cost-sensitive predictor of when O and W differ in the disagreement region of $\mathrm{B}(h^*, r)$; a predictor is good if it has low false-negative error and predicts a positive label with low frequency. If there is no such predictor, then we cannot expect an algorithm similar to ours to achieve label savings.

Algorithm 4.3 IN-DISAGR-REGION($\hat{S}, \tau, x$): Test if x is in the disagreement region of current confidence set

Input: labeled dataset $\hat{S}$, rejection threshold τ, unlabeled example x.
Output: 1 if x in the disagreement region of current confidence set, 0 otherwise.
Train $\hat{h} \leftarrow$ CONS-LEARN$_{\mathcal{H}}(\{\emptyset, \hat{S}\})$.
Train $\hat{h}'_x \leftarrow$ CONS-LEARN$_{\mathcal{H}}(\{(x, -\hat{h}(x))\}, \hat{S}\})$.
if $\mathrm{err}(\hat{h}'_x, \hat{S}) - \mathrm{err}(\hat{h}, \hat{S}) > \tau$ **then** # x is in the agreement region
  return 0
else # x is in the disagreement region
  return 1
end if

Algorithm 4.4 Adaptive Active Learning using Difference Classifier

Input: Unlabeled data distribution U, oracles W and O, difference classifier h^{df}, target excess error ϵ, confidence δ, previous labeled dataset $\hat{T}$.
Output: Parameter σ, labeled dataset $\hat{S}$.
Let $\hat{h} =$ CONS-LEARN$_{\mathcal{H}}(\emptyset, \hat{T})$.
for $t = 1, 2, \ldots,$ **do**
  Let $\delta^t = \delta/t(t+1)$. Define: $\sigma(2^t, \delta^t) = \frac{8}{2^t}(2d \ln \frac{2e2^t}{d} + \ln \frac{24}{\delta^t})$.
  Draw 2^t examples from U to form $S^{t,U}$.
  for each $x \in S^{t,U}$ **do**
    if IN-DISAGR-REGION$(\hat{T}, \frac{3\epsilon}{2}, x) = 0$ **then** # x is in the agreement region
      Add $(x, \hat{h}(x))$ to $\hat{S}^t$.
    else # x is in the disagreement region
      If $h^{df}(x) = +1$, query O for the label y of x, otherwise query W. Add (x, y) to $\hat{S}^t$.
    end if
  end for
  Train $\hat{h}^t \leftarrow$ CONS-LEARN$_{\mathcal{H}}(\emptyset, \hat{S}^t)$.
  if $\sigma(2^t, \delta^t) + \sqrt{\sigma(2^t, \delta^t)\,\mathrm{err}(\hat{h}^t, \hat{S}^t)} \le \epsilon/512$ **then**
    $t_0 \leftarrow t$, **break**
  end if
end for
return $\sigma \leftarrow \sigma(2^{t_0}, \delta^{t_0})$, $\hat{S} \leftarrow \hat{S}^{t_0}$.

Assumption 4.1. *Let $\mathcal{D}$ be the joint distribution: $\mathbb{P}_{\mathcal{D}}[x, y_O, y_W] = \mathbb{P}_U[x]\mathbb{P}_W[y_W|x]\mathbb{P}_O[y_O|x]$. For any $r, \eta > 0$, there exists an $h^{df}_{\eta,r} \in \mathcal{H}^{df}$ with the following properties:*

$$\mathbb{P}_{\mathcal{D}}[h^{df}_{\eta,r}(x) = -1, x \in \mathrm{Dis}(\mathrm{B}(h^*, r)), y_O \neq y_W] \le \eta \tag{4.3}$$

$$\mathbb{P}_{\mathcal{D}}[h^{df}_{\eta,r}(x) = 1, x \in \mathrm{Dis}(\mathrm{B}(h^*, r))] \le \alpha(r, \eta) \tag{4.4}$$

Note that (4.3), which states there is a $h^{df} \in \mathcal{H}^{df}$ with low false-negative error, is minimally restrictive, and is trivially satisfied if $\mathcal{H}^{df}$ includes the constant classifier that always predicts 1. Theorem shows that (4.3) is sufficient to ensure statistical consistency.

(4.4) in addition states that the number of positives predicted by the classifier $h^{df}_{\eta,r}$ is upper bounded by $\alpha(r,\eta)$. Note $\alpha(r,\eta) \leq \mathbb{P}_U(\text{Dis}(\text{B}(h^*,r)))$ always; performance gain is obtained when $\alpha(r,\eta)$ is lower, which happens when the difference classifier predicts agreement on a significant portion of $\text{Dis}(\text{B}(h^*,r))$.

Statistical Consistency. Provided Assumption 4.1 holds, we next show that Algorithm 4.1 is statistically consistent. Establishing consistency is non-trivial for our algorithm as the output classifier is trained on labels from both O and W.

Theorem 4.1 (Statistical Consistency)**.** *Let h^* be the classifier that minimizes the error with respect to D. If Assumption 4.1 holds, then with probability $\geq 1-\delta$, the classifier $\hat{h}$ output by Algorithm 4.1 satisfies:* $\text{err}_D(\hat{h}) \leq \text{err}_D(h^*) + \epsilon$.

Label Complexity. The label complexity of standard DBAL is measured in terms of the disagreement coefficient (See Definition 3.3). It was shown by [DHM07] that the label complexity of DBAL for target excess generalization error ϵ is $\tilde{O}(d\theta(2\nu+\epsilon)(1+\frac{\nu^2}{\epsilon^2}))$. In contrast, the label complexity of our algorithm can be stated in Theorem 4.2. Here we use the $\tilde{O}$ notation for convenience; we have the same dependence on $\log 1/\epsilon$ and $\log 1/\delta$ as the bounds for DBAL.

Theorem 4.2 (Label Complexity)**.** *Let d be the VC dimension of $\mathcal{H}$ and let d' be the VC dimension of $\mathcal{H}^{df}$. If Assumption 4.1 holds, and if the error of the best classifier in $\mathcal{H}$ on D is ν, then with probability $\geq 1-\delta$, the following hold:*

1. *The number of label queries made by Algorithm 4.1 to the oracle O in epoch k at most:*

$$m_k = \tilde{O}\left(\frac{d(2\nu+\epsilon_{k-1})(\alpha(2\nu+\epsilon_{k-1},\frac{\epsilon_{k-1}}{1024})+\epsilon_{k-1})}{\epsilon_k^2} + \frac{d'\mathbb{P}[\text{Dis}(B_U(h^*,2\nu+\epsilon_{k-1}))]}{\epsilon_k}\right) \quad (4.5)$$

2. *The total number of label queries made by Algorithm 4.1 to the oracle O is at most:*

$$\tilde{O}\left(\sup_{r\geq\epsilon}\frac{\alpha(2\nu+r,\frac{r}{1024})+r}{2\nu+r}\cdot d\left(\frac{\nu^2}{\epsilon^2}+1\right)+\theta(2\nu+\epsilon)d'\left(\frac{\nu}{\epsilon}+1\right)\right) \quad (4.6)$$

4.4.1 Discussion

The first terms in (4.5) and (4.6) represent the labels needed to learn the target classifier, and second terms represent the overhead in learning the difference classifier.

In the realistic agnostic case (where $\nu > 0$), as $\epsilon \to 0$, the second terms are of *lower order* compared to the label complexity of DBAL. Thus *even if d' is somewhat larger than d, fitting the difference classifier does not incur an asymptotically high overhead in the more realistic agnostic case.* In the realizable case, when $d' \approx d$, the second terms are of the same order as the first; therefore we should use a simpler difference hypothesis class $\mathcal{H}^{df}$ in this case. We believe that the lower order overhead term comes from the fact that there exists a classifier in $\mathcal{H}^{df}$ whose false negative error is very low.

Comparing Theorem 4.2 with the corresponding results for DBAL, we observe that instead of $\theta(2\nu+\epsilon)$, we have the term $\sup_{r \geq \epsilon} \frac{\alpha(2\nu+r, r/1024)}{2\nu+r}$. Since $\sup_{r \geq \epsilon} \frac{\alpha(2\nu+r, r/1024)}{2\nu+r} \leq \theta(2\nu+\epsilon)$, the *worst case* asymptotic label complexity is the same as that of standard DBAL. This label complexity may be considerably better however if $\sup_{r \geq \epsilon} \frac{\alpha(2\nu+r, r/1024)}{2\nu+r}$ is less than the disagreement coefficient. As we expect, this will happen when the region of difference between W and O restricted to the disagreement regions is relatively small, and this region is well-modeled by the difference hypothesis class $\mathcal{H}^{df}$.

An interesting case is when the weak labeler differs from O close to the decision boundary and agrees with O away from this boundary. In this case, any consistent algorithm should switch to querying O close to the decision boundary. Indeed in earlier epochs, α is low, and our algorithm obtains a good difference classifier and achieves label savings. In later epochs, α is high, the difference classifiers always predict a difference and the label complexity of the later epochs of our algorithm is the same order as DBAL. In practice, if we suspect that we are in this case, we can switch to plain active learning once ϵ_k is small enough.

Case Study: Linear Classfication under Uniform Distribution. We provide a simple example where our algorithm provides a better asymptotic label complexity than DBAL. Let $\mathcal{H}$ be the class of homogeneous linear separators on the d-dimensional unit ball and let $\mathcal{H}^{df} = \{h \Delta h' : h, h' \in \mathcal{H}\}$. Furthermore, let U be the uniform distribution over the unit ball.

Suppose that O is a deterministic labeler such that $\text{err}_D(h^*) = \nu > 0$. Moreover, suppose that W is such that there exists a difference classifier $\bar{h}^{df}$ with false negative error 0 for which $\mathbb{P}_U[\bar{h}^{df}(x) = 1] \leq g$. Additionally, we assume that $g = o(\sqrt{d}\nu)$; observe that this is not a strict assumption on $\mathcal{H}^{df}$, as ν could be as much as a constant. Figure 4.1 shows an example in $d = 2$ that satisfies these assumptions. In this case, as $\epsilon \to 0$, Theorem 4.2 gives the following label complexity bound.

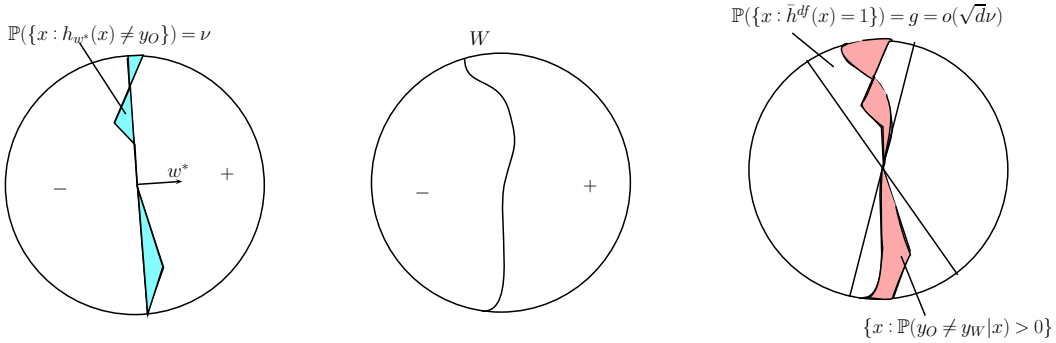

Figure 4.1: Linear classification over unit ball with $d = 2$. Left: Decision boundary of labeler O and $h^* = h_{w^*}$. The region where O differs from h^* is shaded, and has probability ν. Middle: Decision boundary of weak labeler W. Right: $\bar{h}^{df}$, W and O. Note that $\{x : \mathbb{P}(y_O \neq y_W|x) > 0\} \subseteq \{x : \bar{h}^{df}(x) = 1\}$.

Corollary 4.1. *With probability $\geq 1-\delta$, the number of label queries made to oracle O by Algorithm 4.1 is $\tilde{O}\left(d\max(\frac{g}{\nu},1)(\frac{\nu^2}{\epsilon^2}+1)+d^{3/2}\left(1+\frac{\nu}{\epsilon}\right)\right)$.*

As $g = o(\sqrt{d}\nu)$, this improves over the label complexity of DBAL, which is $\tilde{O}(d^{3/2}(1+\frac{\nu^2}{\epsilon^2}))$.

4.5 Additional Notations

In this section, we define several datasets and distributions that will be used in the analysis. Without loss of generality, assume the examples drawn throughout Algorithm 4.1 have distinct feature values x, since this happens with probability 1 under mild assumptions.

Algorithm 4.1 uses a mixture of three kinds of labeled data to learn a target classifier – labels obtained from querying O, labels inferred by the algorithm, and labels obtained from querying W. To analyze the effect of these three kinds of labeled data, we need to introduce some notation.

Recall that we define the joint distribution $\mathcal{D}$ over examples and labels both from O and W as follows:

$$\mathbb{P}_{\mathcal{D}}[x, y_O, y_W] = \mathbb{P}_U[x]\mathbb{P}_O[y_O|x]\mathbb{P}_W[y_W|x]$$

where given an example x, the labels generated by O and W are conditionally independent.

A dataset $\hat{S}$ with empirical error minimizer $\hat{h}$ and a rejection threshold τ define a implicit

candidate set for h^* as follows:

$$V(\hat{S}, \tau) = \{h : \text{err}(h, \hat{S}) - \text{err}(\hat{h}, \hat{S}) \leq \tau\}$$

At the beginning of epoch k, we have $\hat{S}_{k-1}$. $\hat{h}_{k-1}$ is defined as the empirical error minimizer of $\hat{S}_{k-1}$. The disagreement region of the implicit candidate set at epoch k, R_{k-1} is defined as $R_{k-1} := \text{Dis}(V(\hat{S}_{k-1}, 3\epsilon_k/2))$. Algorithm 4.3 In-Disagr-Region($\hat{S}_{k-1}, 3\epsilon_k/2, x$) provides a test deciding if an unlabeled example x is in R_{k-1} in epoch k. (See Lemma 4.6.)

Define $\mathcal{A}_k$ to be the distribution $\mathcal{D}$ conditioned on the set $\{(x, y_O, y_W) : x \in R_{k-1}\}$. At epoch k, Algorithm 4.2 has inputs distribution U, oracles W and O, target false negative error $\epsilon = \epsilon_k/128$, hypothesis class $\mathcal{H}^{df}$, confidence $\delta = \delta_k/2$, previous labeled dataset $\hat{S}_{k-1}$, and outputs a difference classifier $\hat{h}_k^{df}$. By the setting of m in Equation (4.1), Algorithm 4.2 first computes $\hat{p}_k$ using unlabeled examples drawn from U, which is an estimator of $\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})$. Then it draws a subsample of size

$$m_{k,1} = \frac{64 \cdot 1024 \hat{p}_k}{\epsilon_k}(d \ln \frac{512 \cdot 1024 \hat{p}_k}{\epsilon_k} + \ln \frac{144}{\delta_k}) \tag{4.7}$$

iid from $\mathcal{A}_k$. We call the resulting dataset $\mathcal{A}'_k$.

At epoch k, Algorithm 4.4 performs adaptive subsampling to refine the implicit $(1-\delta)$-confidence set. For each round t, it subsamples U to get an unlabeled dataset $S_k^{t,U}$ of size 2^t. Define the corresponding (hypothetical) dataset with labels queried from both W and O as $\mathcal{S}_k^t$. S_k^t, the (hypothetical) dataset with labels queried from O, is defined as:

$$S_k^t = \{(x, y_O) | (x, y_O, y_W) \in \mathcal{S}_k^t\}$$

In addition to obtaining labels from O, the algorithm obtains labels in two other ways. First, if an $x \in \mathcal{X} \setminus R_{k-1}$, then its label is safely inferred and with high probability, this inferred label $\hat{h}_{k-1}(x)$ is equal to $h^*(x)$. Second, if an x lies in R_{k-1} but if the difference classifier $\hat{h}_k^{df}$ predicts agreement between O and W, then its label is obtained by querying W. The actual dataset $\hat{S}_k^t$

generated by Algorithm 4.4 is defined as:

$$\begin{aligned}\hat{S}_k^t &= \{(x,\hat{h}_{k-1}(x))|(x,y_O,y_W)\in\mathcal{S}_k^t, x\notin R_{k-1}\}\\ &\cup\{(x,y_O)|(x,y_O,y_W)\in\mathcal{S}_k^t, x\in R_{k-1}, \hat{h}_k^{df}(x)=+1\}\\ &\cup\{(x,y_W)|(x,y_O,y_W)\in\mathcal{S}_k^t, x\in R_{k-1}, \hat{h}_k^{df}(x)=-1\}\end{aligned}$$

We use $\hat{D}_k$ to denote the labeled data distribution as follows:

$$\mathbb{P}_{\hat{D}_k}[x,y]=\mathbb{P}_U[x]\mathbb{P}_{\hat{Q}_k}[y|x]$$

$$\mathbb{P}_{\hat{Q}_k}[y|x]=\begin{cases}\mathbb{1}(\hat{h}_{k-1}(x)=y), & x\notin R_{k-1}\\ \mathbb{P}_O[y|x], & x\in R_{k-1}, \hat{h}_k^{df}(x)=+1\\ \mathbb{P}_W[y|x], & x\in R_{k-1}, \hat{h}_k^{df}(x)=-1\end{cases}$$

Therefore, $\hat{S}_k^t$ can be seen as a sample of size 2^t drawn iid from $\hat{D}_k$.

Observe that $\hat{h}_k^t$ is obtained by training an ERM classifier over $\hat{S}_k^t$, and $\delta_k^t=\delta_k/2t(t+1)$.

Suppose Algorithm 4.4 stops at iteration $t_0(k)$, then the final dataset returned is $\hat{S}_k=\hat{S}_k^{t_0(k)}$, with a total number of $m_{k,2}$ label requests to O. We define $S_k=S_k^{t_0(k)}$, $\mathcal{S}_k=\mathcal{S}_k^{t_0(k)}$ and $\sigma_k=\sigma(2^{t_0(k)},\delta_k^{t_0(k)})$.

For $k=0$, we define the notation $\hat{S}_k$ differently. $\hat{S}_0$ is the dataset drawn iid at random from D, with labels queried entirely to O. For notational convenience, define $S_0=\hat{S}_0$. σ_0 is defined as $\sigma_0=\sigma(n_0,\delta_0)$, where $\sigma(\cdot,\cdot)$ is defined by Equation (A.1) and n_0 is defined as:

$$n_0=(64\cdot 1024^2)\left(2d\ln(512\cdot 1024^2)+\ln\frac{96}{\delta}\right)$$

Recall that $\hat{h}_k=\arg\min_{h\in\mathcal{H}}\mathrm{err}(h,\hat{S}_k)$ is the empirical error minimizer with respect to the dataset $\hat{S}_k$.

Note that the empirical distance $\rho_Z(\cdot,\cdot)$ does not depend on the labels in dataset Z, therefore, $\rho_{\hat{S}_k}(h,h')=\rho_{S_k}(h,h')$. We will use them interchangeably throughout.

4.6 Adaptive Procedure for Estimating Probability Mass

For completeness, we describe in Algorithm 4.5 a standard doubling procedure for estimating the bias of a coin within a constant factor. This procedure is used by Algorithm 4.2 to estimate the probability mass of the disagreement region of the current confidence set based on unlabeled examples drawn from U.

Algorithm 4.5 Adaptive Procedure for Estimating the Mean of a Bernoulli Random Variable

1: Input: failure probability δ, an oracle $\mathcal{O}$ which returns iid Bernoulli random variables with unknown bias p.
2: Output: $\hat{p}$, an estimate of bias p such that $\hat{p} \leq p \leq 2\hat{p}$ with probability $\geq 1-\delta$.
3: **for** $i = 1, 2, \ldots$ **do**
4: Call the oracle $\mathcal{O}$ 2^i times to get empirical frequency $\hat{p}_i$.
5: **if** $\sqrt{\frac{4 \ln \frac{4 \cdot 2^i}{\delta}}{2^i}} \leq \hat{p}_i/3$ **then**
6: **return** $\hat{p} = \frac{2\hat{p}_i}{3}$
7: **end if**
8: **end for**

Lemma 4.1. *Suppose $p > 0$ and Algorithm 4.5 is run with failure probability δ. Then with probability $1-\delta$, (1) the output $\hat{p}$ is such that $\hat{p} \leq p \leq 2\hat{p}$. (2) The total number of calls to $\mathcal{O}$ is at most $O(\frac{1}{p^2} \ln \frac{1}{\delta p})$.*

Proof. Consider the event

$$E = \left\{ \text{ for all } i \in \mathbb{N}, |\hat{p}_i - p| \leq \sqrt{\frac{4 \ln \frac{2 \cdot 2^i}{\delta}}{2^i}} \right\}$$

By Lemma A.2 and union bound, $\mathbb{P}(E) \geq 1-\delta$. On event E, we claim that if i is large enough that

$$4\sqrt{\frac{4 \ln \frac{4 \cdot 2^i}{\delta}}{2^i}} \leq p \tag{4.8}$$

then the condition in line 5 will be met. Indeed, this implies

$$\sqrt{\frac{4 \ln \frac{4 \cdot 2^i}{\delta}}{2^i}} \leq \frac{p - \sqrt{\frac{4 \ln \frac{4 \cdot 2^i}{\delta}}{2^i}}}{3} \leq \frac{\hat{p}_i}{3}$$

Define i_0 as the smallest number i such that Equation (4.8) is true. Then by algebra, $2^{i_0} = O(\frac{1}{p^2} \ln \frac{1}{\delta p})$. Hence the number of calls to oracle $\mathcal{O}$ is at most $1 + 2 + \ldots + 2^{i_0} = O(\frac{1}{p^2} \ln \frac{1}{\delta p})$.

Consider the smallest i^* such that the condition in line 5 is met. We have that

$$\sqrt{\frac{4\ln\frac{4\cdot 2^{i^*}}{\delta}}{2^{i^*}}} \leq \hat{p}_{i^*}/3$$

By the definition of E,

$$|p - \hat{p}_{i^*}| \leq \hat{p}_{i^*}/3$$

that is, $2\hat{p}_{i^*}/3 \leq p \leq 4\hat{p}_{i^*}/3$, implying $\hat{p} \leq p \leq 2\hat{p}$. □

4.6.1 Events

Recall that $\delta_k = \delta/(4(k+1)^2), \epsilon_k = 2^{-k}$. Define $h_k^{df} := h^{df}_{2\nu+\epsilon_{k-1},\epsilon_k/512}$, where the notation $h^{df}_{r,\eta}$ is introduced in Assumption 4.1. In addition, define function $\gamma(n,\delta) := \frac{4}{n}\ln\frac{2}{\delta}$.

We begin by defining some events that we will condition on later in the proof, and showing that these events occur with high probability.

Define event

$$\begin{aligned}
E_k^1 := \Big\{ \quad & \mathbb{P}_{\mathcal{D}}(x \in R_{k-1})/2 \leq \hat{p}_k \leq \mathbb{P}_{\mathcal{D}}(x \in R_{k-1}), \\
& \text{and for all } h^{df} \in \mathcal{H}^{df}, \\
& |\mathbb{P}_{\mathcal{A}'_k}(h^{df}(x) = -1, y_O \neq y_W) - \mathbb{P}_{\mathcal{A}_k}(h^{df}(x) = -1, y_O \neq y_W)| \leq \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})} + \\
& \sqrt{\min(\mathbb{P}_{\mathcal{A}_k}(h^{df}(x) = -1, y_O \neq y_W), \mathbb{P}_{\mathcal{A}'_k}(h^{df}(x) = -1, y_O \neq y_W))\frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}} \\
& \text{and } |\mathbb{P}_{\mathcal{A}'_k}(h^{df}(x) = +1) - \mathbb{P}_{\mathcal{A}_k}(h^{df}(x) = +1)| \leq \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})} + \\
& \sqrt{\min(\mathbb{P}_{\mathcal{A}_k}(h^{df}(x) = +1), \mathbb{P}_{\mathcal{A}'_k}(h^{df}(x) = +1))\frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}}\Big\}
\end{aligned}$$

Fact 4.1. $\mathbb{P}(E_k^1) \geq 1 - \delta_k/2$.

Define event

$$
\begin{aligned}
E_k^2 = \Big\{ \quad & \forall t \in \mathbb{N}, \text{ for all } h, h' \in \mathcal{H}, \\
& (\mathrm{err}(h, S_k^t) - \mathrm{err}(h', S_k^t)) - (\mathrm{err}_D(h) - \mathrm{err}_D(h')) \leq \sigma(2^t, \delta_k^t) + \sqrt{\sigma(2^t, \delta_k^t)\rho_{S_k^t}(h, h')} \\
\text{and} \quad & \mathrm{err}(h, \hat{S}_k^t) - \mathrm{err}_{\hat{D}_k}(h) \leq \sigma(2^t, \delta_k^t) + \sqrt{\sigma(2^t, \delta_k^t)\,\mathrm{err}_{\hat{D}_k}(h)} \\
\text{and} \quad & \mathbb{P}_{\mathcal{S}_k^t}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) - \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) \\
& \leq \sqrt{\gamma(2^t, \delta_k^t)\mathbb{P}_{\mathcal{S}_k^t}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1})} + \gamma(2^t, \delta_k^t) \\
\text{and} \quad & \mathbb{P}_{\mathcal{S}_k^t}(\hat{h}_k^{df}(x) = -1 \cap x \in R_{k-1}) \leq 2(\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, x \in R_{k-1}) + \gamma(2^t, \delta_k^t)) \Big\}
\end{aligned}
$$

Fact 4.2. $\mathbb{P}(E_k^2) \geq 1 - \delta_k/2$.

We will also use the following definitions of events in our proof. Define event F_0 as

$$
\begin{aligned}
F_0 = \Big\{ \quad & \text{for all } h, h' \in \mathcal{H}, \\
& (\mathrm{err}(h, S_0) - \mathrm{err}(h', S_0)) - (\mathrm{err}_D(h) - \mathrm{err}_D(h')) \leq \sigma(n_0, \delta_0) + \sqrt{\sigma(n_0, \delta_0)\rho_{S_0}(h, h')} \Big\}
\end{aligned}
$$

For $k \in \{1, 2, \ldots, k_0\}$, event F_k is defined inductively as

$$F_k = F_{k-1} \cap (E_k^1 \cap E_k^2)$$

Fact 4.3. *For $k \in \{0, 1, \ldots, k_0\}$, $\mathbb{P}(F_k) \geq 1 - \delta_0 - \delta_1 - \ldots - \delta_k$. Specifically, $\mathbb{P}(F_{k_0}) \geq 1 - \delta$.*

The proofs of Facts 4.1, 4.2 and 4.3 are provided in Section 4.9.

4.7 Proof Outline and Main Lemmas

The main idea of the proof is to maintain the following three invariants on the outputs of Algorithm 4.1 in each epoch. We prove that these invariants hold simultaneously for each epoch with high probability by induction over the epochs. Throughout, for $k \geq 1$, the end of epoch k refers to the end of execution of line 13 of Algorithm 4.1 at iteration k. The end of epoch 0 refers to the end of execution of line 5 in Algorithm 4.1.

Invariant 4.1 states that if we replace the inferred labels and labels obtained from W in $\hat{S}_k$ by those obtained from O (thus getting the dataset S_k), then the excess errors of classifiers

in $\mathcal{H}$ will not decrease by much.

Invariant 4.1 (Approximate Favorable Bias)**.** *Let h be any classifier in $\mathcal{H}$, and h' be another classifier in $\mathcal{H}$ with excess error on D no greater than ϵ_k. Then, at the end of epoch k, we have:*

$$\operatorname{err}(h, S_k) - \operatorname{err}(h', S_k) \leq \operatorname{err}(h, \hat{S}_k) - \operatorname{err}(h', \hat{S}_k) + \epsilon_k/16$$

Invariant 4.2 establishes that in epoch k, Algorithm 4.4 selects enough examples so as to ensure that concentration of empirical errors of classifiers in $\mathcal{H}$ on S_k to their true errors.

Invariant 4.2 (Concentration)**.** *At the end of epoch k, $\hat{S}_k$, S_k and σ_k are such that:*
1. For any pair of classifiers $h, h' \in \mathcal{H}$, it holds that:

$$(\operatorname{err}(h, S_k) - \operatorname{err}(h', S_k)) - (\operatorname{err}_D(h) - \operatorname{err}_D(h')) \leq \sigma_k + \sqrt{\sigma_k \rho_{S_k}(h, h')} \tag{4.9}$$

2. The dataset $\hat{S}_k$ has the following property:

$$\sigma_k + \sqrt{\sigma_k \operatorname{err}(\hat{h}_k, \hat{S}_k)} \leq \epsilon_k/512 \tag{4.10}$$

Finally, Invariant 4.3 ensures that the difference classifier produced in epoch k has low false negative error on the disagreement region of the $(1-\delta)$ confidence set at epoch k.

Invariant 4.3 (Difference Classifier)**.** *At epoch k, the difference classifier output by Algorithm 4.2 is such that*

$$\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) \leq \epsilon_k/64 \tag{4.11}$$

$$\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = +1, x \in R_{k-1}) \leq 6(\alpha(2\nu + \epsilon_{k-1}, \epsilon_k/512) + \epsilon_k/1024) \tag{4.12}$$

We will show the following property about the three invariants. Its proof is deferred to Subsection 4.7.4.

Lemma 4.2. *There is a numerical constant $c_0 > 0$ such that the following holds. The collection of events $\{F_k\}_{k=0}^{k_0}$ is such that for $k \in \{0, 1, \ldots, k_0\}$:*

(1) If $k=0$, then on event F_k, at epoch k,

(1.1) Invariants 1,2 hold.

(1.2) m_0, the number of label requests to O is at most $c_0(d+\ln\frac{1}{\delta})$.

(2) If $k\geq 1$, then on event F_k, at epoch k,

(2.1) Invariants 1,2,3 hold.

(2.2) m_k, the number of label requests to O is at most

$$c_0\Big(\frac{(\alpha(2\nu+\epsilon_{k-1},\epsilon_k/1024)+\epsilon_k)(\nu+\epsilon_k)}{\epsilon_k^2}d(\ln^2\frac{1}{\epsilon_k}+\ln^2\frac{1}{\delta_k}) + \frac{\mathbb{P}_U(x\in\Delta(2\nu+\epsilon_{k-1}))}{\epsilon_k}(d'\ln\frac{1}{\epsilon_k}+\ln\frac{1}{\delta_k})\Big).$$

4.7.1 Active Label Inference and Identifying the Disagreement Region

We begin by proving some lemmas about Algorithm 4.3 which identifies if an example lies in the disagreement region of the current confidence set. This is done by using a constrained ERM oracle Cons-Learn$_H(\cdot,\cdot)$ using ideas similar to [DHM07, BDL09, BHLZ10, BHK$^+$11, ABDL13, HAH$^+$15].

Lemma 4.3. *When given as input a dataset $\hat{S}$, a threshold $\tau>0$, an unlabeled example x, Algorithm 4.3 In-Disagr-Region returns 1 if and only if x lies in* $\mathrm{Dis}(V(\hat{S},\tau))$.

Proof. ($\Rightarrow$) If Algorithm 4.3 returns 1, then we have found a classifier $\hat{h}'_x$ such that (1) $\hat{h}_x(x)=-\hat{h}(x)$, and (2) $\mathrm{err}(\hat{h}'_x,\hat{S})-\mathrm{err}(\hat{h},\hat{S})\leq\tau$, i.e. $\hat{h}'_x\in V(\hat{S},\tau)$. Therefore, x is in $\mathrm{Dis}(V(\hat{S},\tau))$.
($\Leftarrow$) If x is in $\mathrm{Dis}(V(\hat{S},\tau))$, then there exists a classifier $h\in\mathcal{H}$ such that (1) $h(x)=-\hat{h}(x)$ and (2) $\mathrm{err}(h,\hat{S})-\mathrm{err}(\hat{h},\hat{S})\leq\tau$. Hence by definition of $\hat{h}'_x$, $\mathrm{err}(\hat{h}'_x,\hat{S})-\mathrm{err}(\hat{h},\hat{S})\leq\tau$. Thus, Algorithm 4.3 returns 1. □

We now provide some lemmas about the behavior of Algorithm 4.3 called at epoch k.

Lemma 4.4. *Suppose Invariants 4.1 and 4.2 hold at the end of epoch $k-1$. If $h\in\mathcal{H}$ is such that $\mathrm{err}_D(h)\leq\mathrm{err}_D(h^*)+\epsilon_{k-1}/2$, then*

$$\mathrm{err}(h,\hat{S}_{k-1})-\mathrm{err}(\hat{h}_{k-1},\hat{S}_{k-1})\leq 3\epsilon_{k-1}/4$$

Proof. If $h \in \mathcal{H}$ has excess error at most $\epsilon_{k-1}/2$ with respect to D, then,

$$\begin{aligned}
& \operatorname{err}(h, \hat{S}_{k-1}) - \operatorname{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) \\
\leq\ & \operatorname{err}(h, S_{k-1}) - \operatorname{err}(\hat{h}_{k-1}, S_{k-1}) + \epsilon_{k-1}/16 \\
\leq\ & \operatorname{err}_D(h) - \operatorname{err}_D(\hat{h}_{k-1}) + \sigma_{k-1} + \sqrt{\sigma_{k-1}\rho_{S_{k-1}}(h, \hat{h}_{k-1})} + \epsilon_{k-1}/16 \\
\leq\ & \epsilon_{k-1}/2 + \sigma_{k-1} + \sqrt{\sigma_{k-1}\rho_{S_{k-1}}(h, \hat{h}_{k-1})} + \epsilon_{k-1}/16 \\
\leq\ & 9\epsilon_{k-1}/16 + \sigma_{k-1} + \sqrt{\sigma_{k-1}\operatorname{err}(h, \hat{S}_{k-1})} + \sqrt{\sigma_{k-1}\operatorname{err}(\hat{h}_{k-1}, \hat{S}_{k-1})} \\
\leq\ & 9\epsilon_{k-1}/16 + \sigma_{k-1} + \sqrt{\sigma_{k-1}\operatorname{err}(h, \hat{S}_{k-1})} + \sqrt{\sigma_{k-1}(\operatorname{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) + 9\epsilon_{k-1}/16)}
\end{aligned}$$

Where the first inequality follows from Invariant 4.1, the second inequality from Equation (4.9) of Invariant 4.2, the third inequality from the assumption that h has excess error at most $\epsilon_{k-1}/2$, and the fourth inequality from the triangle inequality, the fifth inequality is by adding a nonnegative number in the last term. Continuing,

$$\begin{aligned}
& \operatorname{err}(h, \hat{S}_{k-1}) - \operatorname{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) \\
\leq\ & 9\epsilon_{k-1}/16 + 4\sigma_{k-1} + 2\sqrt{\sigma_{k-1}(\operatorname{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) + 9\epsilon_{k-1}/16)} \\
\leq\ & 9\epsilon_{k-1}/16 + 4\sigma_{k-1} + 2\sqrt{\sigma_{k-1}\operatorname{err}(\hat{h}_{k-1}, \hat{S}_{k-1})} + 2\sqrt{\epsilon_{k-1}/512 \cdot 9\epsilon_{k-1}/16} \\
\leq\ & 9\epsilon_{k-1}/16 + \epsilon_{k-1}/32 + 2\sqrt{\epsilon_{k-1}/512 \cdot 9\epsilon_{k-1}/16} \\
\leq\ & 3\epsilon_{k-1}/4
\end{aligned}$$

Where the first inequality is by simple algebra (by letting $D = \operatorname{err}(h, \hat{S}_{k-1})$, $E = \operatorname{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) + 9\epsilon_{k-1}/16$, $F = \sigma_{k-1}$ in $D \leq E + F + \sqrt{DF} + \sqrt{EF} \Rightarrow D \leq E + 4F + 2\sqrt{EF}$), the second inequality is from $\sqrt{A+B} \leq \sqrt{A} + \sqrt{B}$ and $\sigma_{k-1} \leq \epsilon_{k-1}/512$ which utilizes Equation (4.10) of Invariant 4.2, the third inequality is again by Equation (4.10) of Invariant 4.2, the fourth inequality is by algebra. □

Lemma 4.5. *Suppose Invariants 4.1 and 4.2 hold at the end of epoch $k-1$. Then,*

$$\operatorname{err}_D(\hat{h}_{k-1}) - \operatorname{err}_D(h^*) \leq \epsilon_{k-1}/8$$

Proof. By Lemma 4.4, we know that since h^* has excess error 0 with respect to D,

$$\mathrm{err}(h^*, \hat{S}_{k-1}) - \mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) \leq 3\epsilon_{k-1}/4 \tag{4.13}$$

Therefore,

$$\begin{aligned}
& \mathrm{err}_D(\hat{h}_{k-1}) - \mathrm{err}_D(h^*) \\
\leq\ & \mathrm{err}(\hat{h}_{k-1}, S_{k-1}) - \mathrm{err}(h^*, S_{k-1}) + \sigma_{k-1} + \sqrt{\sigma_{k-1}\rho_{S_{k-1}}(\hat{h}_{k-1}, h^*)} \\
\leq\ & \mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) - \mathrm{err}(h^*, \hat{S}_{k-1}) + \sigma_{k-1} + \sqrt{\sigma_{k-1}\rho_{S_{k-1}}(\hat{h}_{k-1}, h^*)} + \epsilon_{k-1}/16 \\
\leq\ & \epsilon_{k-1}/16 + \sigma_{k-1} + \sqrt{\sigma_{k-1}(\mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) + \mathrm{err}(h^*, \hat{S}_{k-1}))} \\
\leq\ & \epsilon_{k-1}/16 + \sigma_{k-1} + \sqrt{\sigma_{k-1}(2\,\mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) + 3\epsilon_{k-1}/4)} \\
\leq\ & \epsilon_{k-1}/16 + \sigma_{k-1} + \sqrt{2\sigma_{k-1}\,\mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1})} + \sqrt{\epsilon_{k-1}/512 \cdot 3\epsilon_{k-1}/4} \\
\leq\ & \epsilon_{k-1}/8
\end{aligned}$$

where the first inequality is from Equation (4.9) of Invariant 4.2, the second inequality uses Invariant 4.1, the third inequality follows from the optimality of $\hat{h}_{k-1}$ and triangle inequality, the fourth inequality uses Equation (4.13), the fifth inequality uses the fact that $\sqrt{A+B} \leq \sqrt{A} + \sqrt{B}$ and $\sigma_{k-1} \leq \epsilon_{k-1}/512$, which is from Equation (4.10) of Invariant 4.2, the last inequality again utilizes the Equation (4.10) of Invariant 4.2. □

Lemma 4.6. *Suppose Invariants 4.1, 4.2, and 4.3 hold in epoch $k-1$ conditioned on event F_{k-1}. Then conditioned on event F_{k-1}, the implicit confidence set $V_{k-1} = V(\hat{S}_{k-1}, 3\epsilon_k/2)$ is such that:*
(1) If $h \in \mathcal{H}$ satisfies $\mathrm{err}_D(h) - \mathrm{err}_D(h^*) \leq \epsilon_k$, then h is in V_{k-1}.
(2) If $h \in \mathcal{H}$ is in V_{k-1}, then $\mathrm{err}_D(h) - \mathrm{err}_D(h^*) \leq \epsilon_{k-1}$. Hence $V_{k-1} \subseteq \mathrm{B}(h^*, 2\nu + \epsilon_{k-1})$.
(3) Algorithm 4.3, IN-DISAGR-REGION, when run on inputs dataset $\hat{S}_{k-1}$, threshold $3\epsilon_k/2$, unlabeled example x, returns 1 if and only if x is in R_{k-1}.

Proof. (1) Let h be a classifier with $\mathrm{err}_D(h) - \mathrm{err}_D(h^*) \leq \epsilon_k = \epsilon_{k-1}/2$. Then, by Lemma 4.4, one has $\mathrm{err}(h, \hat{S}_{k-1}) - \mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) \leq 3\epsilon_{k-1}/4 = 3\epsilon_k/2$. Hence, h is in V_{k-1}.

(2) Fix any h in V_{k-1}, by definition of V_{k-1},

$$\mathrm{err}(h, \hat{S}_{k-1}) - \mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) \leq 3\epsilon_k/2 = 3\epsilon_{k-1}/4 \tag{4.14}$$

Recall that from Lemma 4.5,

$$\mathrm{err}_D(\hat{h}_{k-1}) - \mathrm{err}_D(h^*) \leq \epsilon_{k-1}/8$$

Thus for classifier h, applying Invariant 4.1 by taking $h' := \hat{h}_{k-1}$, we get

$$\mathrm{err}(h, S_{k-1}) - \mathrm{err}(\hat{h}_{k-1}, S_{k-1}) \leq \mathrm{err}(h, \hat{S}_{k-1}) - \mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) + \epsilon_{k-1}/32 \tag{4.15}$$

Therefore,

$$\begin{aligned}
& \mathrm{err}_D(h) - \mathrm{err}_D(\hat{h}_{k-1}) \\
\leq\ & \mathrm{err}(h, S_{k-1}) - \mathrm{err}(\hat{h}_{k-1}, S_{k-1}) + \sigma_{k-1} + \sqrt{\sigma_{k-1}\rho_{S_{k-1}}(h, \hat{h}_{k-1})} \\
\leq\ & \mathrm{err}(h, S_{k-1}) - \mathrm{err}(\hat{h}_{k-1}, S_{k-1}) + \sigma_{k-1} + \sqrt{\sigma_{k-1}(\mathrm{err}(h, \hat{S}_{k-1}) + \mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}))} \\
\leq\ & \mathrm{err}(h, \hat{S}_{k-1}) - \mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) + \sigma_{k-1} + \sqrt{\sigma_{k-1}(\mathrm{err}(h, \hat{S}_{k-1}) + \mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}))} + \epsilon_{k-1}/16 \\
\leq\ & 13\epsilon_{k-1}/16 + \sigma_{k-1} + \sqrt{\sigma_{k-1}(2\,\mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1}) + 3\epsilon_{k-1}/4)} \\
\leq\ & 13\epsilon_{k-1}/16 + \sigma_{k-1} + \sqrt{2\sigma_{k-1}\,\mathrm{err}(\hat{h}_{k-1}, \hat{S}_{k-1})} + \sqrt{\epsilon_{k-1}/512 \cdot 3\epsilon_{k-1}/4} \\
\leq\ & 7\epsilon_{k-1}/8
\end{aligned}$$

where the first inequality is from Equation (4.9) of Invariant 4.2, the second inequality uses the fact that $\rho_{\hat{S}_{k-1}}(h, h') = \rho_{S_{k-1}}(h, h') \leq \mathrm{err}(h, \hat{S}_{k-1}) + \mathrm{err}(h', \hat{S}_{k-1})$ for $h, h' \in \mathcal{H}$, the third inequality uses Equation (4.15); the fourth inequality is from Equation (4.14); the fifth inequality is from the fact that $\sqrt{A+B} \leq \sqrt{A} + \sqrt{B}$ and $\sigma_{k-1} \leq \epsilon_{k-1}/512$, which is from Equation (4.10) of Invariant 4.2, the last inequality again follows from Equation (4.10) of Invariant 4.2 and algebra. In conjunction with the fact that $\mathrm{err}_D(\hat{h}_{k-1}) - \mathrm{err}_D(h^*) \leq \epsilon_{k-1}/8$, this implies

$$\mathrm{err}_D(h) - \mathrm{err}_D(h^*) \leq \epsilon_{k-1}$$

By triangle inequality, $\rho(h, h^*) \leq 2\nu + \epsilon_{k-1}$, hence $h \in \mathrm{B}(h^*, 2\nu + \epsilon_{k-1})$. In summary $V_{k-1} \subseteq$

$\mathrm{B}(h^*, 2\nu + \epsilon_{k-1})$.

(3) Follows directly from Lemma 4.3 and the fact that $R_{k-1} = \mathrm{Dis}(V_{k-1})$. □

4.7.2 Training the Difference Classifier

Recall that $\Delta(r) = \mathrm{Dis}(\mathrm{B}(h^*, r))$ is the disagreement region of the disagreement ball centered around h^* with radius r.

Lemma 4.7 (Difference Classifier Invariant)**.** *There is a numerical constant $c_1 > 0$ such that the following holds. Suppose that Invariants 4.1 and 4.2 hold at the end of epoch $k-1$ conditioned on event F_{k-1} and that Algorithm 4.2 has inputs unlabeled data distribution U, oracle O, $\epsilon = \epsilon_k/128$, hypothesis class $\mathcal{H}^{df}$, $\delta = \delta_k/2$, previous labeled dataset $\hat{S}_{k-1}$. Then conditioned on event F_k,*
(1) $\hat{h}_k^{df}$, the output of Algorithm 4.2, maintains Invariant 4.3.
(2)(Label Complexity: Part 1.) The number of label queries made to O is at most

$$m_{k,1} \leq c_1 \Big(\frac{\mathbb{P}_U(x \in \Delta(2\nu + \epsilon_{k-1}))}{\epsilon_k} (d' \ln \frac{1}{\epsilon_k} + \ln \frac{1}{\delta_k}) \Big)$$

Proof. (1) Recall that $F_k = F_{k-1} \cap E_k^1 \cap E_k^2$, where E_k^1, E_k^2 are defined in Subsection 4.6.1. Suppose event F_k happens.

Proof of Equation (4.11)**.** Recall that $\hat{h}_k^{df}$ is the optimal solution of optimization problem (4.2). We have by feasibility and the fact that on event E_k^3, $2\hat{p}_k \geq \mathbb{P}_{\mathcal{D}}(x \in R_{k-1})$,

$$\mathbb{P}_{\mathcal{A}'_k}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W) \leq \frac{\epsilon_k}{256\hat{p}_k} \leq \frac{\epsilon_k}{128\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}$$

By definition of event E_k^2, this implies

$$\begin{aligned} & \mathbb{P}_{\mathcal{A}_k}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W) \\ \leq\ & \mathbb{P}_{\mathcal{A}'_k}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W) + \sqrt{\mathbb{P}_{\mathcal{A}'_k}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W) \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}} \\ & + \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})} \\ \leq\ & \frac{\epsilon_k}{64\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})} \end{aligned}$$

Indicating

$$\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) \leq \frac{\epsilon_k}{64}$$

Proof of Equation (4.12)**.** By definition of h_k^{df} in Subsection 4.6.1, h_k^{df} is such that:

$$\mathbb{P}_{\mathcal{D}}(h_k^{df}(x) = +1, x \in \Delta(2\nu + \epsilon_{k-1})) \leq \alpha(2\nu + \epsilon_{k-1}, \epsilon_k/512)$$

$$\mathbb{P}_{\mathcal{D}}(h_k^{df}(x) = -1, y_O \neq y_W, x \in \Delta(2\nu + \epsilon_{k-1})) \leq \epsilon_k/512$$

By item (2) of Lemma 4.6, we have $R_{k-1} \subseteq \text{Dis}(\text{B}(h^*, 2\nu + \epsilon_{k-1}))$, thus

$$\mathbb{P}_{\mathcal{D}}(h_k^{df}(x) = +1, x \in R_{k-1}) \leq \alpha(2\nu + \epsilon_{k-1}, \epsilon_k/512) \tag{4.16}$$

$$\mathbb{P}_{\mathcal{D}}(h_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) \leq \epsilon_k/512 \tag{4.17}$$

Equation (4.17) implies that

$$\mathbb{P}_{\mathcal{A}_k}(h_k^{df}(x) = -1, y_O \neq y_W) \leq \frac{\epsilon_k}{512\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})} \tag{4.18}$$

Recall that $\mathcal{A}_k'$ is the dataset subsampled from $\mathcal{A}_k$ in line 3 of Algorithm 4.2. By definition of event E_k^1, we have that for h_k^{df},

$$\begin{aligned}
& \mathbb{P}_{\mathcal{A}_k'}(h_k^{df}(x) = -1, y_O \neq y_W) \\
\leq\ & \mathbb{P}_{\mathcal{A}_k}(h_k^{df}(x) = -1, y_O \neq y_W) + \sqrt{\mathbb{P}_{\mathcal{A}_k}(h_k^{df}(x) = -1, y_O \neq y_W)\frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}} \\
& + \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})} \\
\leq\ & \frac{\epsilon_k}{256\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})} \leq \frac{\epsilon_k}{256\hat{p}_k}
\end{aligned}$$

where the second inequality is from Equation (4.18), and the last inequality is from the fact that

$\hat{p}_k \le \mathbb{P}_{\mathcal{D}}(x \in R_{k-1})$. Hence, h_k^{df} is a feasible solution to the optimization problem (4.2). Thus,

$$\begin{aligned}
& \mathbb{P}_{\mathcal{A}_k}(\hat{h}_k^{df}(x) = +1) \\
\le\ & \mathbb{P}_{\mathcal{A}'_k}(\hat{h}_k^{df}(x) = +1) + \sqrt{\mathbb{P}_{\mathcal{A}'_k}(\hat{h}_k^{df}(x) = +1)\frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}} + \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})} \\
\le\ & 2(\mathbb{P}_{\mathcal{A}'_k}(\hat{h}_k^{df}(x) = +1) + \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}) \\
\le\ & 2(\mathbb{P}_{\mathcal{A}'_k}(h_k^{df}(x) = +1) + \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}) \\
\le\ & 2\Big((\mathbb{P}_{\mathcal{A}_k}(h_k^{df}(x) = +1) + \sqrt{\mathbb{P}_{\mathcal{A}_k}(h_k^{df}(x) = +1)\frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}} + \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}) \\
& + \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}\Big) \\
\le\ & 6(\mathbb{P}_{\mathcal{A}_k}(h_k^{df}(x) = +1) + \frac{\epsilon_k}{1024\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})})
\end{aligned}$$

where the first inequality is by definition of event E_k^1, the second inequality is by algebra, the third inequality is by optimality of $\hat{h}_k^{df}$ in (4.2), $\mathbb{P}_{\mathcal{A}'_k}(\hat{h}_k^{df}(x) = +1) \le \mathbb{P}_{\mathcal{A}'_k}(h_k^{df}(x) = +1)$, the fourth inequality is by definition of event E_k^1, the fifth inequality is by algebra.

Therefore,

$$\begin{aligned}
\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = +1, x \in R_{k-1}) &\le 6\left(\mathbb{P}_{\mathcal{D}}(h_k^{df}(x) = +1, x \in R_{k-1}) + \frac{\epsilon_k}{1024}\right) \\
&\le 6\left(\alpha(2\nu + \epsilon_{k-1}, \frac{\epsilon_k}{512}) + \frac{\epsilon_k}{1024}\right)
\end{aligned}$$

where the second inequality follows from Equation (4.16). This establishes the correctness of Invariant 4.3.

(2) The number of label requests to O follows from line 3 of Algorithm 4.2 (see Equation (4.7)). That is, we can choose c_1 large enough (independently of k), such that

$$m_{k,1} \le c_1\Big(\frac{\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})}{\epsilon_k}(d' \ln\frac{1}{\epsilon_k} + \ln\frac{1}{\delta_k})\Big) \le c_1\Big(\frac{\mathbb{P}_U(x \in \Delta(2\nu + \epsilon_{k-1}))}{\epsilon_k}(d' \ln\frac{1}{\epsilon_k} + \ln\frac{1}{\delta_k})\Big)$$

where in the second step we use the fact that on event F_k, by item (2) of Lemma 4.6, $R_{k-1} \subseteq \mathrm{Dis}(\mathrm{B}(h^*, 2\nu + \epsilon_{k-1}))$, thus $\mathbb{P}_{\mathcal{D}}(x \in R_{k-1}) \le \mathbb{P}_{\mathcal{D}}(x \in \Delta(2\nu + \epsilon_{k-1})) = \mathbb{P}_U(x \in \Delta(2\nu + \epsilon_{k-1}))$. $\square$

4.7.3 Adaptive Subsampling

Lemma 4.8. *There is a numerical constant $c_2 > 0$ such that the following holds. Suppose Invariants 4.1, 4.2, and 4.3 hold in epoch $k-1$ on event F_{k-1}; Algorithm 4.4 receives inputs unlabeled distribution U, classifier $\hat{h}_{k-1}$, difference classifier $\hat{h}^{df} = \hat{h}_k^{df}$, target excess error $\epsilon = \epsilon_k$, confidence $\delta = \delta_k/2$, previous labeled dataset $\hat{S}_{k-1}$. Then on event F_k,*
(1) $\hat{S}_k$, the output of Algorithm 4.4, maintains Invariants 4.1 and 4.2.
(2) (Label Complexity: Part 2.) The number of label queries to O in Algorithm 4.4 is at most:

$$m_{k,2} \leq c_2\Big(\frac{(\nu+\epsilon_k)(\alpha(2\nu+\epsilon_{k-1},\epsilon_k/512)+\epsilon_k)}{\epsilon_k^2}\cdot d(\ln^2\frac{1}{\epsilon_k}+\ln^2\frac{1}{\delta_k})\Big)$$

Proof. (1) Recall that $F_k = F_{k-1}\cap E_k^1 \cap E_k^2$, where E_k^1, E_k^2 are defined in Subsection 4.6.1. Suppose event F_k happens.

Proof of Invariant 4.1. We consider a pair of classifiers $h, h' \in \mathcal{H}$, where h is an arbitrary classifier in $\mathcal{H}$ and h' has excess error at most ϵ_k.

At iteration $t = t_0(k)$ of Algorithm 4.4, the breaking criterion in line 14 is met, i.e.

$$\sigma(2^{t_0(k)},\delta_k^{t_0(k)}) + \sqrt{\sigma(2^{t_0(k)},\delta_k^{t_0(k)})\,\mathrm{err}(\hat{h}^{t_0(k)},\hat{S}_k^{t_0(k)})} \leq \epsilon_k/512 \tag{4.19}$$

First we expand the definition of $\mathrm{err}(h,S_k)$ and $\mathrm{err}(h,\hat{S}_k)$ respectively:

$$\begin{aligned}\mathrm{err}(h,S_k) &= \mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x)=+1, h(x)\neq y_O, x\in R_{k-1}) + \\ &\quad \mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x)=-1, h(x)\neq y_O, x\in R_{k-1}) + \mathbb{P}_{\mathcal{S}_k}(h(x)\neq y_O, x\notin R_{k-1})\end{aligned}$$

$$\begin{aligned}\mathrm{err}(h,\hat{S}_k) &= \mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x)=+1, h(x)\neq y_O, x\in R_{k-1}) + \\ &\quad \mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x)=-1, h(x)\neq y_W, x\in R_{k-1}) + \mathbb{P}_{\mathcal{S}_k}(h(x)\neq h^*(x), x\notin R_{k-1})\end{aligned}$$

where we use the fact that by Lemma 4.6, for all examples $x \notin R_{k-1}$, $\hat{h}_{k-1}(x) = h^*(x)$.

We next show that $\mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x)=-1, h(x)\neq y_O, x\in R_{k-1})$ is close to $\mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x)=-1, h(x)\neq y_W, x\in R_{k-1})$.

From Lemma 4.7, we know that conditioned on event F_k,

$$\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) \leq \epsilon_k/64$$

In the meantime, from Equation (4.19), $\gamma(2^{t_0(k)}, \delta_k^{t_0(k)}) \leq \sigma(2^{t_0(k)}, \delta_k^{t_0(k)}) \leq \epsilon_k/512$. Recall that $\mathcal{S}_k = \mathcal{S}_k^{t_0(k)}$. Therefore, by definition of E_k^2,

$$\begin{aligned}
& \mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) \\
\leq\ & \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) + \sqrt{\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1})\gamma(2^{t_0(k)}, \delta_k^{t_0(k)})} \\
& + \gamma(2^{t_0(k)}, \delta_k^{t_0(k)}) \\
\leq\ & \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) + \sqrt{\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1})\epsilon_k/512} + \frac{\epsilon_k}{512} \\
\leq\ & \epsilon_k/32
\end{aligned}$$

By triangle inequality, for all classifier $h_0 \in \mathcal{H}$,

$$|\mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, h_0(x) \neq y_O, x \in R_{k-1}) - \mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, h_0(x) \neq y_W, x \in R_{k-1})| \leq \epsilon_k/32 \tag{4.20}$$

Specifically for h and h', Equation (4.20) hold:

$$|\mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, h(x) \neq y_O, x \in R_{k-1}) - \mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, h(x) \neq y_W, x \in R_{k-1})| \leq \epsilon_k/32$$

$$|\mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, h'(x) \neq y_O, x \in R_{k-1}) - \mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, h'(x) \neq y_W, x \in R_{k-1})| \leq \epsilon_k/32$$

Combining, we get:

$$\begin{aligned}
& (\mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, h(x) \neq y_W, x \in R_{k-1}) - \mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, h'(x) \neq y_W, x \in R_{k-1})) \quad (4.21) \\
- \quad & (\mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, h(x) \neq y_O, x \in R_{k-1}) - \mathbb{P}_{\mathcal{S}_k}(\hat{h}_k^{df}(x) = -1, h'(x) \neq y_O, x \in R_{k-1})) \leq \epsilon_k/16
\end{aligned}$$

We now show the labels inferred in the region $\mathcal{X} \setminus R_{k-1}$ is "favorable" to the classifiers whose excess error is at most $\epsilon_k/2$.

By triangle inequality,

$$\mathbb{P}_{\mathcal{S}_k}(h(x) \neq y_O, x \notin R_{k-1}) - \mathbb{P}_{\mathcal{S}_k}(h^*(x) \neq y_O, x \notin R_{k-1}) \leq \mathbb{P}_{\mathcal{S}_k}(h(x) \neq h^*(x), x \notin R_{k-1})$$

By Lemma 4.6, since h' has excess error at most ϵ_k, h' agrees with h^* on all x in $\mathcal{X} \setminus R_{k-1}$ on event F_{k-1}, hence $\mathbb{P}_{\mathcal{S}_k}(h'(x) \neq h^*(x), x \notin R_{k-1}) = 0$. This gives

$$\begin{aligned} & \mathbb{P}_{\mathcal{S}_k}(h(x) \neq y_O, x \notin R_{k-1}) - \mathbb{P}_{\mathcal{S}_k}(h'(x) \neq y_O, x \notin R_{k-1}) \\ \leq \; & \mathbb{P}_{\mathcal{S}_k}(h(x) \neq h^*(x), x \notin R_{k-1}) - \mathbb{P}_{\mathcal{S}_k}(h'(x) \neq h^*(x), x \notin R_{k-1}) \end{aligned} \tag{4.22}$$

Combining Equations (4.21) and (4.22), we conclude that

$$\operatorname{err}(h, S_k) - \operatorname{err}(h', S_k) \leq \operatorname{err}(h, \hat{S}_k) - \operatorname{err}(h', \hat{S}_k) + \epsilon_k/16$$

This establishes the correctness of Invariant 4.1.

Proof of Invariant 4.2. Recall by definition of E_k^2 the following concentration results hold for all $t \in \mathbb{N}$:

$$(\operatorname{err}(h, S_k^t) - \operatorname{err}(h', S_k^t)) - (\operatorname{err}_D(h) - \operatorname{err}_D(h')) \leq \sigma(2^t, \delta_k^t) + \sqrt{\sigma(2^t, \delta_k^t)\rho_{S_k^t}(h, h')}$$

In particular, for iteration $t_0(k)$ we have

$$\begin{aligned} & (\operatorname{err}(h, S_k^{t_0(k)}) - \operatorname{err}(h', S_k^{t_0(k)})) - (\operatorname{err}_D(h) - \operatorname{err}_D(h')) \\ \leq \; & \sigma(2^{t_0(k)}, \delta_k^{t_0(k)}) + \sqrt{\sigma(2^{t_0(k)}, \delta_k^{t_0(k)})\rho_{S_k^{t_0(k)}}(h, h')} \end{aligned}$$

Recall that $\hat{S}_k = \hat{S}_k^{t_0(k)}$, $\hat{h}_k = \hat{h}_k^{t_0(k)}$, and $\sigma_k = \sigma(2^{t_0(k)}, \delta_k^{t_0(k)})$, hence the above is equivalent to

$$|(\operatorname{err}(h, S_k) - \operatorname{err}(h', S_k)) - (\operatorname{err}_D(h) - \operatorname{err}_D(h'))| \leq \sigma_k + \sqrt{\sigma_k \rho_{S_k}(h, h')} \tag{4.23}$$

Equation (4.23) establishes the correctness of Equation (4.9) of Invariant 4.2. Equation (4.10) of Invariant 4.2 follows from Equation (4.19).

(2) We define $\tilde{h}_k = \arg\min_{h \in \mathcal{H}} \operatorname{err}_{\hat{D}_k}(h)$, and define $\tilde{\nu}_k$ to be $\operatorname{err}_{\hat{D}_k}(\tilde{h}_k)$. To prove the bound on the number of label requests, we first claim that if t is sufficiently large that

$$\sigma(2^t, \delta_k^t) + \sqrt{\sigma(2^t, \delta_k^t)\tilde{\nu}_k} \leq \epsilon_k/1536 \tag{4.24}$$

then the algorithm will satisfy the breaking criterion at line 14 of Algorithm 4.4, that is, for this value of t,

$$\sigma(2^t, \delta_k^t) + \sqrt{\sigma(2^t, \delta_k^t)\,\mathrm{err}(\hat{h}^t, \hat{S}_k^t)} \le \epsilon_k/512 \tag{4.25}$$

Indeed, by definition of E_k^2, if event F_k happens,

$$\begin{aligned} & \mathrm{err}(\tilde{h}_k, \hat{S}_k^t) \\ \le\ & \mathrm{err}_{\hat{D}_k}(\tilde{h}_k) + \sigma(2^t, \delta_k^t) + \sqrt{\sigma(2^t, \delta_k^t)\,\mathrm{err}_{\hat{D}_k}(\tilde{h}_k)} \\ =\ & \tilde{\nu}_k + \sigma(2^t, \delta_k^t) + \sqrt{\sigma(2^t, \delta_k^t)\tilde{\nu}_k} \end{aligned} \tag{4.26}$$

Therefore,

$$\begin{aligned} & \sigma(2^t, \delta_k^t) + \sqrt{\sigma(2^t, \delta_k^t)\,\mathrm{err}(\hat{h}_k^t, \hat{S}_k^t)} \\ \le\ & \sigma(2^t, \delta_k^t) + \sqrt{\sigma(2^t, \delta_k^t)\,\mathrm{err}(\tilde{h}_k, \hat{S}_k^t)} \\ \le\ & \sigma(2^t, \delta_k^t) + \sqrt{\sigma(2^t, \delta_k^t)(2\tilde{\nu}_k + 2\sigma(2^t, \delta_k^t))} \\ \le\ & 3\sigma(2^t, \delta_k^t) + 2\sqrt{\sigma(2^t, \delta_k^t)\tilde{\nu}_k} \\ \le\ & \epsilon_k/512 \end{aligned}$$

where the first inequality is from the optimality of $\hat{h}_k^t$, the second inequality is from Equation (4.26), the third inequality is by algebra, the last inequality follows from Equation (4.24). The claim follows.

Next, we solve for the minimum t that satisfies (4.24), which is an upper bound of $t_0(k)$. Fact A.1 implies that there is a numerical constant $c_3 > 0$ such that

$$2^{t_0(k)} \le c_3 \frac{\tilde{\nu}_k + \epsilon_k}{\epsilon_k^2}(d \ln \frac{1}{\epsilon_k} + \ln \frac{1}{\delta_k}))$$

Thus, there is a numerical constant $c_4 > 0$ such that

$$t_0(k) \le c_4(\ln d + \ln \frac{1}{\epsilon_k} + \ln \ln \frac{1}{\delta_k})$$

Hence, there is a numerical constant $c_5 > 0$ (that does not depend on k) such that the following holds. If event F_k happens, then the number of label queries made by Algorithm 4.4 to O can

be bounded as follows:

$$\begin{aligned}
m_{k,2} &= \sum_{t=1}^{t_0(k)} |S_k^{t,U} \cap \{x : \hat{h}_k^{df}(x) = +1\} \cap R_{k-1}| \\
&= \sum_{t=1}^{t_0(k)} 2^t \mathbb{P}_{\mathcal{S}_k^t}(\hat{h}_k^{df}(x) = +1, x \in R_{k-1}) \\
&\leq \sum_{t=1}^{t_0(k)} 2^t (2\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = +1, x \in R_{k-1}) + 2 \cdot 4 \frac{\ln \frac{2}{\delta_k^t}}{2^t}) \\
&\leq 4 \cdot 2^{t_0(k)} \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = +1, x \in R_{k-1}) + 8 \cdot t_0(k) \ln \frac{2}{\delta_k^{t_0(k)}} \\
&\leq c_5 \Big((\frac{(\tilde{\nu}_k + \epsilon_k)\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = +1, x \in R_{k-1})}{\epsilon_k^2} + 1) \cdot d(\ln^2 \frac{1}{\epsilon_k} + \ln^2 \frac{1}{\delta_k}) \Big) \\
&\leq c_5 \Big((\frac{(\tilde{\nu}_k + \epsilon_k) \cdot 6(\alpha(2\nu + \epsilon_{k-1}, \epsilon_k/512) + \epsilon_k/1024)}{\epsilon_k^2} + 1) \cdot d(\ln^2 \frac{1}{\epsilon_k} + \ln^2 \frac{1}{\delta_k}) \Big)
\end{aligned}$$

where the second equality is from the fact that $|S_k^{t,U} \cap \{x : \hat{h}_k^{df}(x) = -1\} \cap R_{k-1}| = |S_k^{t,U}| \cdot \mathbb{P}_{\mathcal{S}_k^t}(\hat{h}_k^{df}(x) = -1, x \in R_{k-1})$, in conjunction with $|S_k^{t,U}| = 2^t$; the first inequality is by definition of E_k^2, the second and third inequality is from algebra that $t_0(k) \ln \frac{1}{\delta_k^{t_0(k)}} \leq c_5 d(\ln^2 \frac{1}{\epsilon_k} + \ln^2 \frac{1}{\delta_k})$ for some constant $c_5 > 0$, along with the choice of c_2, the fourth step is from Lemma 4.7 which states that Invariant 4.3 holds at epoch k.

What remains to be argued is an upper bound on $\tilde{\nu}_k$. Note that

$$\begin{aligned}
& \tilde{\nu}_k \\
= & \min_{h \in \mathcal{H}} [\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, h(x) \neq y_W, x \in R_{k-1}) \\
& + \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = +1, h(x) \neq y_O, x \in R_{k-1}) + \mathbb{P}_{\mathcal{D}}(h(x) \neq h^*(x), x \notin R_{k-1})] \\
\leq & \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, h^*(x) \neq y_W, x \in R_{k-1}) + \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = +1, h^*(x) \neq y_O, x \in R_{k-1}) \\
\leq & \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, h^*(x) \neq y_O, x \in R_{k-1}) + \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = +1, h^*(x) \neq y_O, x \in R_{k-1}) + \epsilon_k/64 \\
\leq & \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, h^*(x) \neq y_O, x \in R_{k-1}) + \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = +1, h^*(x) \neq y_O, x \in R_{k-1}) \\
& + \mathbb{P}_{\mathcal{D}}(h(x) \neq y_O, x \notin R_{k-1}) + \epsilon_k/64 \\
= & \nu + \epsilon_k/64
\end{aligned}$$

where the first step is by definition of $\mathrm{err}_{\hat{D}_k}(h)$, the second step is by the suboptimality of h^*, the third step is by Equation (4.20), the fourth step is by adding a positive term $\mathbb{P}_{\mathcal{D}}(h(x) \neq y_O, x \notin R_{k-1})$, the fifth step is by definition of $\mathrm{err}_D(h)$. Therefore, we conclude that there is a numerical constant $c_2 > 0$, such that $m_{k,2}$, the number of label requests to O in Algorithm 4.4 is at most

$$c_2\Big(\frac{(\nu+\epsilon_k)(\alpha(2\nu+\epsilon_{k-1},\epsilon_k/512)+\epsilon_k)}{\epsilon_k^2}\cdot d(\ln^2\frac{1}{\epsilon_k}+\ln^2\frac{1}{\delta_k})\Big)$$

□

4.7.4 Putting It Together – Statistical Consistency and Label Complexity

Proof of Lemma 4.2. With foresight, pick $c_0 > 0$ to be a large enough constant. We prove the result by induction.

Base case. Consider $k = 0$. Recall that F_0 is defined as

$$F_0 = \Big\{ \quad \text{for all } h, h' \in \mathcal{H}, \\ (\mathrm{err}(h, S_0) - \mathrm{err}(h', S_0)) - (\mathrm{err}_D(h) - \mathrm{err}_D(h')) \leq \sigma(n_0, \delta_0) + \sqrt{\sigma(n_0,\delta_0)\rho_{S_0}(h,h')}\Big\}$$

Note that by definition in Subsection 4.5, $\hat{S}_0 = S_0$. Therefore Invariant 4.1 trivially holds. When F_0 happens, Equation (4.9) of Invariant 4.2 holds, and n_0 is such that $\sqrt{\sigma_0} \leq \epsilon_0/1024$, thus,

$$\sigma_0 + \sqrt{\sigma_0 \,\mathrm{err}(\hat{h}_0, \hat{S}_0)} \leq \epsilon_0/512$$

which establishes the validity of Equation (4.10) of Invariant 4.2.

Meanwhile, the number of label requests to O is

$$n_0 = 64 \cdot 1024^2 (d \ln(512 \cdot 1024^2) + \ln\frac{96}{\delta})) \leq c_0(d + \ln\frac{1}{\delta})$$

Inductive case. Suppose the claim holds for $k' < k$. The inductive hypothesis states that Invariants 1,2,3 hold in epoch $k-1$ on event F_{k-1}. By Lemma 4.7 and Lemma 4.8, Invariants 1,2,3 holds in epoch k on event F_k. Suppose F_k happens. By Lemma 4.7, there is a numerical

constant $c_1 > 0$ such that the number of label queries in Algorithm 4.2 in line 12 is at most

$$m_{k,1} \le c_1\Big(\frac{\mathbb{P}_U(x \in \Delta(2\nu+\epsilon_{k-1}))}{\epsilon_k}(d' \ln\frac{1}{\epsilon_k} + \ln\frac{1}{\delta_k})\Big)$$

Meanwhile, by Lemma 4.8, there is a numerical constant $c_2 > 0$ such that the number of label queries in Algorithm 4.4 in line 14 is at most

$$m_{k,2} \le c_2\Big(\frac{(\alpha(2\nu+\epsilon_{k-1},\epsilon_k/512)+\epsilon_k)(\nu+\epsilon_k)}{\epsilon_k^2} \cdot d(\ln^2\frac{1}{\epsilon_k} + \ln^2\frac{1}{\delta_k})\Big)$$

Thus, the number of label requests in total at epoch k is at most

$$\begin{aligned} m_k &= m_{k,1} + m_{k,2} \\ &\le c_0\Big(\frac{(\alpha(2\nu+\epsilon_{k-1},\epsilon_k/512)+\epsilon_k)(\nu+\epsilon_k)}{\epsilon_k^2} d(\ln^2\frac{1}{\epsilon_k} + \ln^2\frac{1}{\delta_k}) \\ &\qquad + \frac{\mathbb{P}_U(x \in \Delta(2\nu+\epsilon_{k-1}))}{\epsilon_k}(d' \ln\frac{1}{\epsilon_k} + \ln\frac{1}{\delta_k})\Big) \end{aligned}$$

This completes the induction. □

Theorem 4.3 (Statistical Consistency)**.** *If F_{k_0} happens, then the classifier $\hat{h}$ returned by Algorithm 4.1 is such that*

$$\mathrm{err}_D(\hat{h}) - \mathrm{err}_D(h^*) \le \epsilon$$

Proof. By Lemma 4.2, Invariants 4.1, 4.2, 4.3 hold at epoch k_0. Thus by Lemma 4.5,

$$\mathrm{err}_D(\hat{h}) - \mathrm{err}_D(h^*) = \mathrm{err}_D(\hat{h}_{k_0}) - \mathrm{err}_D(h^*) \le \epsilon_{k_0}/8 \le \epsilon$$

□

Proof of Theorem 4.1. This is an immediate consequence of Theorem 4.3. □

Theorem 4.4 (Label Complexity)**.** *If F_{k_0} happens, then the number of label queries made by Algorithm 4.1 to O is at most*

$$\tilde{O}\left((\sup_{r\ge\epsilon}\frac{\alpha(2\nu+r,r/1024)}{2\nu+r})d(\frac{\nu^2}{\epsilon^2}+1) + (\sup_{r\ge\epsilon}\frac{\mathbb{P}_U(x\in\Delta(2\nu+r))}{2\nu+r})d'(\frac{\nu}{\epsilon}+1)\right)$$

Proof. Conditioned on event F_{k_0}, we bound the sum $\sum_{k=0}^{k_0} m_k$.

$$
\begin{aligned}
& \sum_{k=0}^{k_0} m_k \\
\leq\ & c_0(d+\ln\frac{1}{\delta}) + c_0\Big(\sum_{k=1}^{k_0} \frac{(\alpha(2\nu+\epsilon_{k-1},\epsilon_k/512)+\epsilon_k)(\nu+\epsilon_k)}{\epsilon_k^2} d(\ln^2\frac{1}{\epsilon_k}+\ln^2\frac{1}{\delta_k}) \\
& + \frac{\mathbb{P}_U(x\in\Delta(2\nu+\epsilon_{k-1}))}{\epsilon_k}(d'\ln\frac{1}{\epsilon_k}+\ln\frac{1}{\delta_k})\Big) \\
\leq\ & c_0(d+\ln\frac{1}{\delta}) + c_0\Big(\sum_{k=1}^{k_0} \frac{(\alpha(2\nu+\epsilon_{k-1},\epsilon_k/512)+\epsilon_k)(\nu+\epsilon_k)}{\epsilon_k^2} d(3\ln^2\frac{1}{\epsilon}+2\ln^2\frac{1}{\delta}) \\
& + \frac{\mathbb{P}_U(x\in\Delta(2\nu+\epsilon_{k-1}))}{\epsilon_k}(2d'\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta})\Big) \\
\leq\ & (\sup_{r\geq\epsilon}\frac{\alpha(2\nu+r,r/1024)+r}{2\nu+r}) d(3\ln^2\frac{1}{\epsilon}+2\ln^2\frac{1}{\delta}) \sum_{k=0}^{k_0}\frac{(\nu+\epsilon_k)^2}{\epsilon_k^2} \\
& + \sup_{r\geq\epsilon}\frac{\mathbb{P}_U(x\in\Delta(2\nu+r))}{2\nu+r}(2d'\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta})\sum_{k=0}^{k_0}\frac{(\nu+\epsilon_k)}{\epsilon_k} \\
\leq\ & \tilde{O}\left((\sup_{r\geq\epsilon}\frac{\alpha(2\nu+r,r/1024)+r}{2\nu+r}) d(\frac{\nu^2}{\epsilon^2}+1) + (\sup_{r\geq\epsilon}\frac{\mathbb{P}_U(x\in\Delta(2\nu+r))}{2\nu+r}) d'(\frac{\nu}{\epsilon}+1)\right)
\end{aligned}
$$

where the first inequality is by Lemma 4.2, the second inequality is by noticing for all $k \geq 1$, $\ln^2\frac{1}{\epsilon_k}+\ln^2\frac{1}{\delta_k} \leq 3\ln^2\frac{1}{\epsilon}+2\ln^2\frac{1}{\delta}$ and $d'\ln\frac{1}{\epsilon_k}+\ln\frac{1}{\delta_k} \leq 2d'\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta}$, the rest of the derivations follows from standard algebra.

□

Proof of Theorem 4.2. Item 1 is an immediate consequence of Lemma 4.2, whereas item 2 is a consequence of Theorem 4.4. □

4.8 Case Study: Linear Classification under Uniform Distribution over Unit Ball

We remind the reader the setting of our example in Section 4.4. $\mathcal{H}$ is the class of homogeneous linear separators on the d-dimensional unit ball and $\mathcal{H}^{df}$ is defined to be $\{h\Delta h' : h, h' \in \mathcal{H}\}$. Note that d' is at most $5d$. Furthermore, U is the uniform distribution over the unit ball. O is a deterministic labeler such that $\mathrm{err}_D(h^*) = \nu > 0$, W is such that there exists a

difference classifier $\bar{h}^{df}$ with false negative error 0 for which $\Pr_U(\bar{h}^{df}(x)=1) \le g = o(\sqrt{d}\nu)$. We prove the label complexity bound provided by Corollary 4.1.

Proof of Corollary 4.1. We claim that under the assumptions of Corollary 4.1, $\alpha(2\nu+r, r/1024)$ is at most g. Indeed, by taking $h^{df} = \bar{h}^{df}$, observe that

$$P(\bar{h}^{df}(x) = -1, y_W \neq y_O, x \in \Delta(2\nu+r)) \le P(\bar{h}^{df}(x) = -1, y_W \neq y_O) = 0$$

$$P(\bar{h}^{df}(x) = +1, x \in \Delta(2\nu+r)) \le g$$

This shows that $\alpha(2\nu+r, 0) \le g$. Hence, $\alpha(2\nu+r, r/1024) \le \alpha(2\nu+r, 0) \le g$. Therefore,

$$\sup_{r: r \ge \epsilon} \frac{\alpha(2\nu+r, r/1024)+r}{2\nu+r} \le \sup_{r \ge \epsilon} \frac{g+r}{\nu+r} \le \max(\frac{g}{\nu}, 1)$$

Recall that the disagreement coefficient $\theta(2\nu+r) \le \sqrt{d}$ for all r, and $d' \le 5d$. Thus, by Theorem 4.2, the number of label queries to O is at most

$$\tilde{O}\left(d\max(\frac{g}{\nu}, 1)(\frac{\nu^2}{\epsilon^2}+1) + d^{3/2}\left(1+\frac{\nu}{\epsilon}\right)\right)$$

□

4.9 Remaining Proofs

Proof of Fact 4.1. (1) First by Lemma 4.1, $\mathbb{P}_{\mathcal{D}}(x \in R_{k-1})/2 \le \hat{p}_k \le \mathbb{P}_{\mathcal{D}}(x \in R_{k-1})$ holds with probability $1 - \delta_k/6$.

Second, for each classifier $h^{df} \in \mathcal{H}^{df}$, define functions $f^1_{h^{df}}$, and $f^2_{h^{df}}$ associated with it. Formally,

$$f^1_{h^{df}}(x, y_O, y_W) = \mathbb{1}(h^{df}(x) = -1, y_O \neq y_W)$$

$$f^2_{h^{df}}(x, y_O, y_W) = \mathbb{1}(h^{df}(x) = +1)$$

Consider the function class $\mathcal{F}^1 = \{f^1_{h^{df}} : h^{df} \in \mathcal{H}^{df}\}$, $\mathcal{F}^2 = \{f^2_{h^{df}} : h^{df} \in \mathcal{H}^{df}\}$. Note that both $\mathcal{F}^1$ and $\mathcal{F}^2$ have VC dimension d', which is the same as $\mathcal{H}^{df}$. We note that $\mathcal{A}'_k$ is a random sample of size m_k drawn iid from $\mathcal{A}_k$. The fact follows from normalized VC inequality on $\mathcal{F}^1$ and $\mathcal{F}^2$

and the choice of m_k in Algorithm 4.2 called in epoch k, along with union bound. □

Proof of Fact 4.2. For fixed t, we note that S_k^t is a random sample of size 2^t drawn iid from D. By Equation (A.4) of Lemma A.5, for any fixed $t \in \mathbb{N}$,

$$\mathbb{P}\Big(\text{ for all } h,h' \in \mathcal{H}, |(\text{err}(h,S_k^t) - \text{err}(h',S_k^t)) - (\text{err}_D(h) - \text{err}_D(h'))| \leq \sigma(2^t,\delta_k^t) + \sqrt{\sigma(2^t,\delta_k^t)\rho_{S_k^t}(h,h')}\Big) \geq 1 - \delta_k^t/8$$

Meanwhile, for fixed $t \in \mathbb{N}$, note that $\hat{S}_k^t$ is a random sample of size 2^t drawn iid from $\hat{D}_k$. By Equation (A.2) of Lemma A.5,

$$\mathbb{P}\Big(\text{ for all } h,h' \in \mathcal{H}, \text{err}(h,\hat{S}_k^t) - \text{err}_{\hat{D}_k}(h) \leq \sigma(2^t,\delta_k^t) + \sqrt{\sigma(2^t,\delta_k^t)\,\text{err}_{\hat{D}_k}(h)}\Big) \geq 1 - \delta_k^t/8 \quad (4.27)$$

Moreover, for fixed $t \in \mathbb{N}$, note that $\mathcal{S}_k^t$ is a random sample of size 2^t drawn iid from $\mathcal{D}$. By Lemma A.2,

$$\begin{aligned}\mathbb{P}\Big(& \mathbb{P}_{\mathcal{S}_k^t}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) \leq \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1}) \\ + & \sqrt{\gamma(2^t,\delta_k^t)\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, y_O \neq y_W, x \in R_{k-1})} + \gamma(2^t,\delta_k^t)\Big) \geq 1 - \delta_k^t/8 \end{aligned} \quad (4.28)$$

Finally, for fixed $t \in N$, note that $\mathcal{S}_k^t$ is a random sample of size 2^t drawn iid from $\mathcal{D}$. By Lemma A.2,

$$\begin{aligned}\mathbb{P}\Big(& \mathbb{P}_{\mathcal{S}_k^t}(\hat{h}_k^{df}(x) = -1, x \in R_{k-1}) \leq \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, x \in R_{k-1}) \\ + & \sqrt{\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, x \in R_{k-1})\gamma(2^t,\delta_k^t)} + \gamma(2^t,\delta_k^t)\Big) \geq 1 - \delta_k^t/8 \end{aligned} \quad (4.29)$$

Note that by algebra,

$$\begin{aligned} & \mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, x \in R_{k-1}) + \sqrt{\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, x \in R_{k-1})\gamma(2^t,\delta_k^t)} + \gamma(2^t,\delta_k^t) \\ \leq & \; 2(\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, x \in R_{k-1}) + \gamma(2^t,\delta_k^t)) \end{aligned}$$

Therefore,

$$\mathbb{P}\Big(\mathbb{P}_{\mathcal{S}_k^t}(\hat{h}_k^{df}(x) = -1, x \in R_{k-1}) \leq 2(\mathbb{P}_{\mathcal{D}}(\hat{h}_k^{df}(x) = -1, x \in R_{k-1}) + \gamma(2^t,\delta_k^t))\Big) \geq 1 - \delta_k^t/12 \quad (4.30)$$

The proof follows by applying union bound over Equations (4.27), (4.27), (4.28) and (4.30) and $t \in \mathbb{N}$. □

We emphasize that $\mathcal{S}_k^t$ is chosen iid at random after $\hat{h}_k^{df}$ is determined, thus uniform convergence argument over $\mathcal{H}^{df}$ is not necessary for Equations (4.28) and (4.30).

Proof of Fact 4.3. By induction on k.

Base Case. For $k = 0$, it follows directly from normalized VC inequality that $\mathbb{P}(F_0) \geq 1 - \delta_0$.

Inductive Case. Assume $\mathbb{P}(F_{k-1}) \geq 1 - \delta_0 - \ldots - \delta_{k-1}$ holds. By union bound,

$$\mathbb{P}(F_k) \geq \mathbb{P}(F_{k-1} \cap E_k^1 \cap E_k^2) \geq \mathbb{P}(F_{k-1}) - \delta_k/2 - \delta_k/2 \geq \mathbb{P}(F_{k-1}) - \delta_k$$

Hence, $\mathbb{P}(F_k) \geq 1 - \delta_0 - \ldots - \delta_k$. This finishes the induction.
In particular, $\mathbb{P}(F_{k_0}) \geq 1 - \delta_0 - \ldots \delta_{k_0} \geq 1 - \delta$. □

4.10 Acknowledgements

This chapter is based on the material as it appears in Advances in Neural Information Processing Systems 2015 (Chicheng Zhang and Kamalika Chaudhuri, "Active Learning from Weak and Strong Labelers"). The dissertation author is the primary investigator and author of this material.

Chapter 5

Active Learning using a Search Oracle

5.1 Introduction

Most active learning theory is based on interacting with Label oracles. As discussed in Section 1.1.1, a well-known deficiency of Label arises in the presence of rare classes in classification problems, frequently the case in practice [AP10, SCL$^+$14]. Class imbalance may be so extreme that simply finding an example from the rare class can exhaust the labeling budget. Consider the problem of learning interval functions in $[0, 1]$. Any Label-only active learner needs at least $\Omega(1/\epsilon)$ Label queries to learn an arbitrary target interval with error at most ϵ [Das05]. Given any positive example from the interval, however, the query complexity of learning intervals collapses to $O(\log(1/\epsilon))$, as we can just do a binary search for each of the end points.

A natural approach used to overcome this hurdle in practice is to search for known examples of the rare class [AP10, SCL$^+$14]. Domain experts are often adept at finding examples of a class by various, often clever means. For instance, when building a hate speech filter, a simple web search can readily produce a set of positive examples. Sending a random batch of unlabeled text to Label is unlikely to produce any positive examples at all. Another form of interaction common in practice is providing counterexamples to a learned predictor. When monitoring the stream filtered by the current hate speech filter, a human editor may spot a clear-cut example of

hate speech that seeped through the filter. The editor, using all the search tools available to her, may even be tasked with searching for such counterexamples. The goal of the learning system is then to interactively restrict the searchable space, guiding the search process to where it is most effective.

In this chapter, we define a new oracle, SEARCH, that provides counterexamples to version spaces. Given a set of possible classifiers H mapping unlabeled examples to labels, a *candidate set* $V \subseteq H$ is the subset of classifiers still under consideration by the algorithm. A *counterexample* to a candidate set is a labeled example which every classifier in the candidate set classifies incorrectly. When there is no counterexample to the candidate set, SEARCH returns nothing. In addition, we consider a nested sequence of hypothesis classes of increasing complexity, akin to model selection in passive learning [Vap82, DGL96]. When SEARCH produces a counterexample to the candidate set, it gives a proof that the current hypothesis class is too simplistic to solve the problem effectively. We show that this guided increase in hypothesis complexity results in a radically lower LABEL complexity than directly learning on the complex space. Sample complexity bounds for model selection in LABEL-only active learning were studied by [BHV10, Han09].

SEARCH can easily model the practice of seeding discussed earlier. If the first hypothesis class has just the constant always-negative classifier $h(x) = -1$, a seed example with label $+1$ is a counterexample to the candidate set. Our most basic algorithm uses SEARCH just once before using LABEL, but it is clear from inspection that multiple seeds are not harmful, and they may be helpful if they provide the proof required to operate with an appropriately complex hypothesis class. Defining SEARCH with respect to a candidate set rather than a single classifier allows us to formalize "counterexample far from the boundary" in a general fashion which is compatible with the way LABEL-based active learning algorithms work.

5.2 Definitions and Setting

Recall from Chapter 3, in active learning, there is an underlying distribution D over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the instance space and $\mathcal{Y} := \{-1, +1\}$ is the label space. The learner can obtain independent draws from D, but the label is hidden unless explicitly requested through a query to the LABEL oracle. Let $D_{\mathcal{X}}$ denote the marginal of D over $\mathcal{X}$.

We consider active learning with a nested sequence of hypotheses classes $H_0 \subset H_1 \subset \cdots \subset H_k \cdots$, where $H_k \subseteq \mathcal{Y}^{\mathcal{X}}$ has VC dimension d_k. For a set of labeled examples $S \subseteq \mathcal{X} \times \mathcal{Y}$, let

$H_k(S) := \{h \in H_k : \forall (x,y) \in S \centerdot h(x) = y\}$ be the set of hypotheses in H_k consistent with S. Let $h_k^* = \arg\min_{h \in H_k} \mathrm{err}(h)$ breaking ties arbitrarily and let $k^* := \arg\min_{k \geq 0} \mathrm{err}(h_k^*)$ breaking ties in favor of the smallest such k. For simplicity, we assume the minimum is attained at some finite k^*. Finally, define $h^* := h_{k^*}^*$, the optimal hypothesis in the sequence of classes. The goal of the learner is to learn a hypothesis with error rate not much more than that of h^*.

In addition to LABEL, the learner can also query SEARCH with a candidate set.

Oracle $\textsc{Search}_H(V)$ (where $H \in \{H_k\}_{k=0}^{\infty}$)

Input: Set of hypotheses $V \subset H$

Output: Labeled example $(x, h^*(x))$ s.t. $h(x) \neq h^*(x)$ for all $h \in V$, or $\perp$ if there is no such example.

Thus if $\textsc{Search}_H(V)$ returns an example, this example is a *systematic* mistake made by all hypotheses in V. (If $V = \emptyset$, we expect SEARCH to return some example, i.e., not $\perp$.)

5.3 The Relative Power of the Two Oracles

Although SEARCH cannot always implement LABEL efficiently, it is as effective at reducing the region of disagreement. The clearest example is learning threshold classifiers $H := \{h_w : w \in [0,1]\}$ in the realizable case, where $h_w(x) = +1$ if $w \leq x \leq 1$, and -1 if $0 \leq x < w$. A simple binary search with LABEL achieves an exponential improvement in query complexity over passive learning. The agreement region of any set of threshold classifiers with thresholds in $[w_{\min}, w_{\max}]$ is $[0, w_{\min}) \cup [w_{\max}, 1]$. Since SEARCH is allowed to return any counterexample in the agreement region, there is no mechanism for forcing SEARCH to return the label of a particular point we want. However, this is not needed to achieve logarithmic query complexity with SEARCH: If binary search starts with querying the label of $x \in [0,1]$, we can query $\textsc{Search}_H(V_x)$, where $V_x := \{h_w \in H : w < x\}$ instead. If SEARCH returns $\perp$, we know that the target $w^* \leq x$ and can safely reduce the region of disagreement to $[0,x)$. If SEARCH returns a counterexample $(x_0, -1)$ with $x_0 \geq x$, we know that $w^* > x_0$ and can reduce the region of disagreement to $(x_0, 1]$.

This observation holds more generally. In the proposition below, we assume that $\textsc{Label}(x)$ equals $h^*(x)$ for simplicity. If $\textsc{Label}(x)$ is noisy, the proposition holds for any active learning algorithm that doesn't eliminate any $h \in H : h(x) = \textsc{Label}(x)$ from the candidate set.

Proposition 5.1. *For any call $x \in \mathcal{X}$ to* LABEL *such that* LABEL$(x) = h^*(x)$*, we can construct a call to* SEARCH *that achieves a no lesser reduction in the region of disagreement.*

Proof. For any $V \subseteq H$, let $H_{\text{SEARCH}}(V)$ be the hypotheses in H consistent with the output of SEARCH$_H(V)$: if SEARCH$_H(V)$ returns a counterexample (x, y) to V, then $H_{\text{SEARCH}}(V) := \{h \in H : h(x) = y\}$; otherwise, $H_{\text{SEARCH}}(V) := V$. Let $H_{\text{LABEL}}(x) := \{h \in H : h(x) = \text{LABEL}(x)\}$. Also, let $V_x := H_{+1}(x) := \{h \in H : h(x) = +1\}$. We will show that V_x is such that $H_{\text{SEARCH}}(V_x) \subseteq H_{\text{LABEL}}(x)$, and hence $\text{Dis}(H_{\text{SEARCH}}(V_x)) \subseteq \text{Dis}(H_{\text{LABEL}}(x))$.

There are two cases to consider: If $h^*(x) = +1$, then SEARCH$_H(V_x)$ returns $\perp$. In this case, $H_{\text{LABEL}}(x) = H_{\text{SEARCH}}(V_x) = H_{+1}(x)$, and we are done. If $h^*(x) = -1$, SEARCH(V_x) returns a valid counterexample (possibly $(x, -1)$) in the region of agreement of $H_{+1}(x)$, eliminating all of $H_{+1}(x)$. Thus $H_{\text{SEARCH}}(V_x) \subset H \setminus H_{+1}(x) = H_{\text{LABEL}}(x)$, and the claim holds also. □

As shown by the problem of learning intervals on the line, SEARCH can be exponentially more powerful than LABEL.

5.4 Realizable Case

We now turn to general active learning algorithms that combine SEARCH and LABEL. We focus on algorithms using *both* SEARCH and LABEL since LABEL is typically easier to implement than SEARCH and hence should be used where SEARCH has no significant advantage. (Whenever SEARCH is less expensive than LABEL, Section 5.3 suggests a transformation to a SEARCH-only algorithm.)

This section considers the realizable case, in which we assume that the hypothesis $h^* = h^*_{k^*} \in H_{k^*}$ has $\text{err}(h^*) = 0$. This means that LABEL(x) returns $h^*(x)$ for any x in the support of $D_{\mathcal{X}}$.

5.4.1 Combining LABEL and SEARCH

Our algorithm (shown as Algorithm 5.1) is called LARCH, because it combines LABEL and SEARCH. Like many selective sampling methods, LARCH uses a candidate set to determine its LABEL queries.

For concreteness, we use (a variant of) the algorithm of [CAL94], denoted by CAL, as a subroutine in LARCH. The inputs to CAL are: a candidate set V, the LABEL oracle, a target error

Algorithm 5.1 LARCH

Input: Nested hypothesis classes $H_0 \subset H_1 \subset \cdots$; oracles LABEL and SEARCH; learning parameters $\epsilon, \delta \in (0,1)$

1: **initialize** $S \leftarrow \emptyset$, (index) $k \leftarrow 0$, $\ell \leftarrow 0$
2: **for** $i = 1, 2, \ldots$ **do**
3: $\quad e \leftarrow \text{SEARCH}_{H_k}(H_k(S))$
4: $\quad$ **if** $e = \perp$ **then** # no counterexample found
5: $\quad\quad$ **if** $2^{-\ell} \leq \epsilon$ **then**
6: $\quad\quad\quad$ **return** any $h \in H_k(S)$
7: $\quad\quad$ **else**
8: $\quad\quad\quad \ell \leftarrow \ell + 1$
9: $\quad\quad$ **end if**
10: $\quad$ **else** # counterexample found
11: $\quad\quad S \leftarrow S \cup \{e\}$
12: $\quad\quad k \leftarrow \min\{k' : H_{k'}(S) \neq \emptyset\}$
13: $\quad$ **end if**
14: $\quad S \leftarrow S \cup \text{CAL}(H_k(S), \text{LABEL}, 2^{-\ell}, \delta/(i^2+i))$
15: **end for**

rate, and a failure probability; and its output is a set of labeled examples (implicitly defining a new candidate set). CAL is described in Section 5.7; its essential properties are specified in Lemma 5.1.

LARCH differs from LABEL-only active learners (like CAL) by first calling SEARCH in Step 3. If SEARCH returns $\perp$, LARCH checks to see if the last call to CAL resulted in a small-enough error, halting if so in Step 6, and decreasing the allowed error rate if not in Step 8. If SEARCH instead returns a counterexample, the hypothesis class H_k must be impoverished, so in Step 12, LARCH increases the complexity of the hypothesis class to the minimum complexity sufficient to correctly classify all known labeled examples in S. After the SEARCH, CAL is called in Step 14 to discover a sufficiently low-error (or at least low-disagreement) candidate set with high probability.

When LARCH advances to index k (for any $k \leq k^*$), its set of labeled examples S may imply a candidate set $H_k(S) \subseteq H_k$ that can be actively-learned more efficiently than the whole of H_k. In our analysis, we quantify this through the disagreement coefficient of $H_k(S)$, which may be markedly smaller than that of the full H_k.

The following theorem bounds the oracle query complexity of Algorithm 5.1 for learning with both SEARCH and LABEL in the realizable setting. The proof is in section 5.4.2.

Theorem 5.1. *Assume that* $\text{err}(h^*) = 0$. *For each* $k' \geq 0$, *let* $\theta_{k'}(\cdot)$ *be the disagreement coefficient of* $H_{k'}(S_{[k']})$, *where* $S_{[k']}$ *is the set of labeled examples* S *in* LARCH *at the first time that* $k \geq k'$.

Fix any $\epsilon, \delta \in (0,1)$. If LARCH *is run with inputs hypothesis classes $\{H_k\}_{k=0}^{\infty}$, oracles* LABEL *and* SEARCH*, and learning parameters ϵ, δ, then with probability at least $1-\delta$:* LARCH *halts after at most $k^* + \log_2(1/\epsilon)$ for-loop iterations and returns a classifier with error rate at most ϵ; furthermore, it draws at most $\tilde{O}(k^* d_{k^*}/\epsilon)$ unlabeled examples from $D_{\mathcal{X}}$, makes at most $k^* + \log_2(1/\epsilon)$ queries to* SEARCH*, and at most $\tilde{O}\left(\left(k^* + \log(1/\epsilon)\right) \cdot (\max_{k' \le k^*} \theta_{k'}(\epsilon)) \cdot d_{k^*} \cdot \log^2(1/\epsilon)\right)$ queries to* LABEL*.*

Union-of-intervals example. We now show an implication of Theorem 5.1 in the case where the target hypothesis h^* is the union of non-trivial intervals in $\mathcal{X} := [0,1]$, assuming that $D_{\mathcal{X}}$ is uniform. For $k \ge 0$, let H_k be the hypothesis class of the union of up to k intervals in $[0,1]$ with H_0 containing only the always-negative hypothesis. (Thus, h^* is the union of k^* non-empty intervals.) The disagreement coefficient of H_1 is $\Omega(1/\epsilon)$, and hence LABEL-only active learners like CAL are not very effective at learning with such classes. However, the first SEARCH query by LARCH provides a counterexample to H_0, which must be a positive example $(x_1, +1)$. Hence, $H_1(S_{[1]})$ (where $S_{[1]}$ is defined in Theorem 5.1) is the class of intervals that contain x_1 with disagreement coefficient $\theta_1 \le 4$.

Now consider the inductive case. Just before LARCH advances its index to a value k (for any $k \le k^*$), SEARCH returns a counterexample $(x, h^*(x))$ to the candidate set; every hypothesis in this candidate set (which could be empty) is a union of fewer than k intervals. If the candidate set is empty, then S must already contain positive examples from at least k different intervals in h^* and at least $k-1$ negative examples separating them. If the candidate set is not empty, then the point x is either a positive example belonging to a previously uncovered interval in h^* or a negative example splitting an existing interval. In either case, $S_{[k]}$ contains positive examples from at least k distinct intervals separated by at least $k-1$ negative examples. The disagreement coefficient of the set of unions of k intervals consistent with $S_{[k]}$ is at most $4k$, independent of ϵ.

The VC dimension of H_k is $O(k)$, so Theorem 5.1 implies that with high probability, LARCH makes at most $k^* + \log(1/\epsilon)$ queries to SEARCH and $\tilde{O}((k^*)^3 \log(1/\epsilon) + (k^*)^2 \log^3(1/\epsilon))$ queries to LABEL.

5.4.2 Proof of Theorem 5.1

The proof of Theorem 5.1 uses the following lemma regarding the CAL subroutine, proved in Section 5.7. It is similar to a result of [Han09], but an important difference here is that the

input candidate set V is not assumed to contain h^*.

Lemma 5.1. *Assume* $\textsc{Label}(x) = h^*(x)$ *for every* x *in the support of* $D_{\mathcal{X}}$*. For any hypothesis set* $V \subseteq \mathcal{Y}^{\mathcal{X}}$ *with VC dimension* $d < \infty$*, and any* $\epsilon, \delta \in (0,1)$*, the following holds with probability at least* $1-\delta$*.* $\mathrm{CAL}(V, \textsc{Label}, \epsilon, \delta)$ *returns labeled examples* $T \subseteq \{(x, h^*(x)) : x \in \mathcal{X}\}$ *such that for any* h *in* $V(T)$*,* $\Pr_{(x,y)\sim D}[h(x) \neq y \ \wedge \ x \in \mathrm{Dis}(V(T))] \leq \epsilon$*; furthermore, it draws at most* $\tilde{O}(d/\epsilon)$ *unlabeled examples from* $D_{\mathcal{X}}$*, and makes at most* $\tilde{O}\left(\theta_V(\epsilon) \cdot d \cdot \log^2(1/\epsilon)\right)$ *queries to* $\textsc{Label}$*.*

We now prove Theorem 5.1. By Lemma 5.1 and a union bound, there is an event with probability at least $1 - \sum_{i \geq 1} \delta/(i^2+i) \geq 1-\delta$ such that each call to CAL made by Larch satisfies the high-probability guarantee from Lemma 5.1. We henceforth condition on this event.

We first establish the guarantee on the error rate of a hypothesis returned by Larch. By the assumed properties of Label and Search, and the properties of CAL from Lemma 5.1, the labeled examples S in Larch are always consistent with h^*. Moreover, the return property of CAL implies that at the end of any loop iteration, with the present values of S, k, and ℓ, we have $\Pr_{(x,y)\sim D}[h(x) \neq y \wedge x \in \mathrm{Dis}(H_k(S))] \leq 2^{-\ell}$ for all $h \in H_k(S)$. (The same holds trivially before the first loop iteration.) Therefore, if Larch halts and returns a hypothesis $h \in H_k(S)$, then there is no counterexample to $H_k(S)$, and $\Pr_{(x,y)\sim D}[h(x) \neq y \wedge x \in \mathrm{Dis}(H_k(S))] \leq \epsilon$. These consequences and the law of total probability imply $\mathrm{err}(h) = \Pr_{(x,y)\sim D}[h(x) \neq y \wedge x \in \mathrm{Dis}(H_k(S))] \leq \epsilon$.

We next consider the number of for-loop iterations executed by Larch. Let S_i, k_i, and ℓ_i be, respectively, the values of S, k, and ℓ at the start of the i-th for-loop iteration in Larch. We claim that if Larch does not halt in the i-th iteration, then one of k and ℓ is incremented by at least one. Clearly, if there is no counterexample to $H_{k_i}(S_i)$ and $2^{-\ell_i} > \epsilon$, then ℓ is incremented by one (Step 8). If, instead, there is a counterexample (x,y), then $H_{k_i}(S_i \cup \{(x,y)\}) = \emptyset$, and hence k is incremented to some index larger than k_i (Step 12). This proves that $k_{i+1} + \ell_{i+1} \geq k_i + \ell_i + 1$. We also have $k_i \leq k^*$, since $h^* \in H_{k^*}$ is consistent with S, and $\ell_i \leq \log_2(1/\epsilon)$, as long as Larch does not halt in for-loop iteration i. So the total number of for-loop iterations is at most $k^* + \log_2(1/\epsilon)$. Together with Lemma 5.1, this bounds the number of unlabeled examples drawn from $D_{\mathcal{X}}$.

Finally, we bound the number of queries to Search and Label. The number of queries to Search is the same as the number of for-loop iterations—this is at most $k^* + \log_2(1/\epsilon)$. By Lemma 5.1 and the fact that $V(S' \cup S'') \subseteq V(S')$ for any hypothesis space V and sets of labeled examples S', S'', the number of Label queries made by CAL in the i-th for-loop iteration is at

most $\tilde{O}(\theta_{k_i}(\epsilon) \cdot d_{k_i} \cdot \ell_i^2 \cdot \text{polylog}(i))$. The claimed bound on the number of LABEL queries made by LARCH now readily follows by taking a max over i, and using the facts that $i \leq k^*$ and $d_{k'} \leq d_{k^*}$ for all $k' \leq k$. □

5.4.3 An Improved Algorithm

LARCH is somewhat conservative in its use of SEARCH, interleaving just one SEARCH query between sequences of LABEL queries (from CAL). Often, it is advantageous to advance to higher complexity hypothesis classes quickly, as long as there is justification to do so. Counterexamples from SEARCH provide such justification, and a $\perp$ result from SEARCH also provides useful feedback about the current candidate set: outside of its disagreement region, the candidate set is in complete agreement with h^* (even if the version space does not contain h^*). Based on these observations, we propose an improved algorithm for the realizable setting, which we call SEABEL. We defer its presentation to Section 5.8. We prove the following performance guarantee for SEABEL.

Theorem 5.2. *Assume that* $\text{err}(h^*) = 0$. *Let* $\theta_k(\cdot)$ *denote the disagreement coefficient of* $V_i^{k_i}$ *at the first iteration* i *in* SEABEL *where* $k_i \geq k$. *Fix any* $\epsilon, \delta \in (0,1)$. *If* SEABEL *is run with inputs hypothesis classes* $\{H_k\}_{k=0}^{\infty}$, *oracles* SEARCH *and* LABEL, *and learning parameters* $\epsilon, \delta \in (0,1)$, *then with probability* $1-\delta$: SEABEL *halts and returns a classifier with error rate at most* ϵ; *furthermore, it draws* $\tilde{O}((d_{k^*} + \log k^*)/\epsilon)$ *unlabeled examples from* $D_{\mathcal{X}}$, *makes* $k^* + O(\log(d_{k^*}/\epsilon) + \log\log k^*)$ *queries to* SEARCH, *and* $\tilde{O}(\max_{k \leq k^*} \theta_k(2\epsilon) \cdot (d_{k^*} \log^2(1/\epsilon) + \log k^*))$ *queries to* LABEL.

It is not generally possible to directly compare Theorems 5.1 and 5.2 on account of the algorithm-dependent disagreement coefficient bounds. However, in cases where these disagreement coefficients are comparable (as in the union-of-intervals example), the SEARCH complexity in Theorem 5.2 is slightly higher (by additive log terms), but the LABEL complexity is smaller than that from Theorem 5.1 by roughly a factor of k^*. For the union-of-intervals example, SEABEL would learn target union of k^* intervals with $k^* + O(\log(k^*/\epsilon))$ queries to SEARCH and $\tilde{O}((k^*)^2 \log^2(1/\epsilon))$ queries to LABEL.

5.5 Non-Realizable Case

In this section, we consider the case where the optimal hypothesis h^* may have non-zero error rate, i.e., the non-realizable (or agnostic) setting. In this case, the algorithm LARCH, which was designed for the realizable setting, is no longer applicable. First, examples obtained by LABEL and SEARCH are of different quality: those returned by SEARCH always agree with h^*, whereas the labels given by LABEL need not agree with h^*. Moreover, the candidate sets (even when $k = k^*$) as defined by LARCH may always be empty due to the noisy labels.

Another complication arises in our SRM setting that differentiates it from the usual agnostic active learning setting. When working with a specific hypothesis class H_k in the nested sequence, we may observe high error rates because (i) the finite sample error is too high (but additional labeled examples could reduce it), or (ii) the current hypothesis class H_k is impoverished. In case (ii), the best hypothesis in H_k may have a much larger error rate than h^*, and hence lower bounds [Kää06] imply that active learning on H_k instead of H_{k^*} may be substantially more difficult.

These difficulties in the SRM setting are circumvented by an algorithm that adaptively estimates the error of h^*. The algorithm, A-LARCH (Algorithm 5.5), is presented in Section 5.9.

Theorem 5.3. *Assume* $\mathrm{err}(h^*) = \nu$. *Let* $\theta_k(\cdot)$ *denote the disagreement coefficient of* $V_i^{k_i}$ *at the first iteration* i *in A-LARCH where* $k_i \geq k$. *Fix any* $\epsilon, \delta \in (0,1)$. *If A-LARCH is run with inputs hypothesis classes* $\{H_k\}_{k=0}^{\infty}$, *oracles SEARCH and LABEL, learning parameter* δ, *and unlabeled example budget* $\tilde{O}((d_{k^*} + \log k^*)(\nu+\epsilon)/\epsilon^2)$, *then with probability* $1-\delta$: *A-LARCH returns a classifier with error rate* $\leq \nu + \epsilon$; *it makes at most* $k^* + O\left(\log(d_{k^*}/\epsilon) + \log\log k^*\right)$ *queries to SEARCH, and* $\tilde{O}\left(\max_{k \leq k^*} \theta_k(2\nu+2\epsilon) \cdot (d_{k^*} \log^2(1/\epsilon) + \log k^*) \cdot (1 + \nu^2/\epsilon^2)\right)$ *queries to LABEL.*

The proof is in Section 5.9. The LABEL query complexity is at least a factor of k^* better than that in [Han09], and sometimes exponentially better thanks to the reduced disagreement coefficient of the candidate set when consistency constraints are incorporated.

5.5.1 AA-Larch: an Opportunistic Anytime Algorithm

In many practical scenarios, termination conditions based on quantities like a target excess error rate ϵ are undesirable. The target ϵ is unknown, and we instead prefer an algorithm that performs as well as possible until a cost budget is exhausted. Fortunately, when the primary

cost being considered are LABEL queries, there are many LABEL-only active learning algorithms that readily work in such an "anytime" setting [DHM07, Han14].

The situation is more complicated when we consider both SEARCH and LABEL: we can often make substantially more progress with SEARCH queries than with LABEL queries (as the error rate of the best hypothesis in $H_{k'}$ for $k' > k$ can be far lower than in H_k). AA-LARCH (Algorithm 5.2) shows that although these queries come at a higher cost, the cost can be amortized.

Algorithm 5.2 AA-LARCH

Input: Nested hypothesis set $H_0 \subseteq H_1 \subseteq \cdots$; oracles LABEL and SEARCH; learning parameter $\delta \in (0,1)$; SEARCH-to-LABEL cost ratio τ, dataset size upper bound N.
Output: hypothesis $\tilde{h}$.

Initialize: consistency constraints $S \leftarrow \emptyset$, counter $c \leftarrow 0$, $k \leftarrow 0$, verified labeled dataset $\tilde{L} \leftarrow \emptyset$, working labeled dataset $L_0 \leftarrow \emptyset$, unlabeled examples processed $i \leftarrow 0$, $V_i \leftarrow H_k(S)$.
loop
  Reset counter $c \leftarrow 0$.
  repeat
    if ERROR-CHECK(V_i, L_i, δ_i) **then**
      $(k, S, V_i) \leftarrow$ UPGRADE-CANDIDATE-SET$(k, S, \emptyset)$
      $V_i \leftarrow$ PRUNE-CANDIDATE-SET$(V_i, \tilde{L}, \delta_i)$
      $L_i \leftarrow \tilde{L}$
      continue loop
    end if
    $i \leftarrow i + 1$
    $(L_i, c) \leftarrow$ SAMPLE-AND-LABEL$(V_{i-1}, \text{LABEL}, L_{i-1}, c)$
    $V_i \leftarrow$ PRUNE-CANDIDATE-SET(V_{i-1}, L_i, δ_i)
  until $c = \tau$ **or** $l_i = N$
  $e \leftarrow \text{SEARCH}_{H_k}(V_i)$
  if $e \neq \perp$ **then**
    $(k, S, V_i) \leftarrow$ UPGRADE-CANDIDATE-SET$(k, S, \{e\})$
    $V_i \leftarrow$ PRUNE-CANDIDATE-SET$(V_i, \tilde{L}, \delta_i)$
    $L_i \leftarrow \tilde{L}$
  else
    Update verified dataset $\tilde{L} \leftarrow L_i$.
    Store temporary solution $\tilde{h} = \arg\min_{h' \in V_i} \text{err}(h', \tilde{L})$.
  end if
end loop

AA-LARCH relies on several crucial subroutines: SAMPLE-AND-LABEL, ERROR-CHECK, PRUNE-CANDIDATE-SET and UPGRADE-CANDIDATE-SET (Algorithms 5.6, 5.7, 5.8, and 5.9). The detailed descriptions are deferred to Section 5.10. SAMPLE-AND-LABEL performs standard disagreement-based selective sampling using oracle LABEL; labels of examples in the disagreement region are queried, otherwise inferred. PRUNE-CANDIDATE-SET prunes the candidate set given the labeled examples collected, based on standard generalization error bounds.

ERROR-CHECK checks if the best hypothesis in the candidate set has large error; SEARCH is used to find a systematic mistake for the candidate set; if either event happens, AA-LARCH calls UPGRADE-CANDIDATE-SET to increase k, the level of our working hypothesis class.

Theorem 5.4. *Assume* $\mathrm{err}(h^*) = \nu$. *Let* $\theta_{k'}(\cdot)$ *denote the disagreement coefficient of* V_i *at the first iteration* i *after which* $k \geq k'$. *Fix any* $\epsilon \in (0,1)$. *Let* $n_\epsilon = \tilde{O}(\max_{k \leq k^*} \theta_k(2\nu + 2\epsilon) d_{k^*}(1 + \nu^2/\epsilon^2))$ *and define* $C_\epsilon = 2(n_\epsilon + k^*\tau)$. *Run Algorithm 5.2 with a nested sequence of hypotheses* $\{H_k\}_{k=0}^{\infty}$, *oracles* LABEL *and* SEARCH, *failure probability* δ, *cost ratio* $\tau \geq 1$, *and upper bound* $N = \tilde{O}(d_{k^*}/\epsilon^2)$. *If the cost spent is at least* C_ϵ, *then with probability* $1 - \delta$, *the current hypothesis* $\tilde{h}$ *has error at most* $\nu + \epsilon$.

The proof is in Section 5.10. A comparison to Theorem 5.3 shows that AA-LARCH is adaptive: for any cost complexity C, the excess error rate ϵ is roughly at most twice that achieved by A-LARCH.

5.6 Basic Notations Used in Proofs

Because we apply the deviation inequalities to the hypothesis classes from $\{H_k\}_{k=0}^{\infty}$, we also define:

$$\sigma_k(m, \delta) \ := \ \sigma(d_k, m, \delta), \tag{5.1}$$

where d_k is the VC dimension of H_k.

For integers $i \geq 1$ and $k \geq 0$, define

$$\delta_i := \frac{\delta}{i(i+1)}, \qquad \delta_{i,k} := \frac{\delta_i}{(k+1)(k+2)}$$

Note that $\sum_{i=1}^{\infty} \delta_i = \delta$ and $\sum_{k=0}^{\infty} \delta_{i,k} = \delta_i$.

5.7 Active Learning Algorithm CAL

In this section, we describe and analyze a variant of the LABEL-only active learning algorithm of [CAL94], which we refer to as CAL. Note that [Han09] provides a label complexity analysis of CAL in terms of the disagreement coefficient under the assumption that the LABEL oracle is consistent with some hypothesis in the hypothesis class used by CAL. We cannot use

Algorithm 5.3 CAL

Input: Hypothesis set V with VC dimension $\leq d$; oracle LABEL; learning parameters $\epsilon, \delta \in (0,1)$
Output: Labeled examples T

for $i = 1, 2, \ldots$ **do**
  $T_i \leftarrow \emptyset$
  for $j = 1, 2, \ldots, 2^i$ **do**
    $x_{i,j} \leftarrow$ independent draw from $D_{\mathcal{X}}$ (the corresponding label is hidden)
    if $x_{i,j} \in \mathrm{Dis}(V(T_{\leq i-1}))$ **then**
      $T_i \leftarrow T_i \cup \{(x_{i,j}, \textsc{Label}(x_{i,j}))\}$
    end if
  end for
  if $\sigma(d, 2^i, \delta_i/2) \leq \epsilon$ or $V(T_{\leq i}) = \emptyset$ **then**
    return $T_{\leq i}$
  end if
end for

that analysis because we call CAL as a subroutine in LARCH with sets of hypotheses V that do not necessarily contain the optimal hypothesis h^*.

5.7.1 Description of CAL

CAL takes as input a set of hypotheses V, the LABEL oracle (which always returns $h^*(x)$ when queried with a point x), and learning parameters $\epsilon, \delta \in (0,1)$.

The pseudocode for CAL is given in Algorithm 5.3 below, where we use the notation

$$U_{\leq i} \;:=\; \bigcup_{j=1}^{i} U_j$$

for any sequence of sets $(U_j)_{j \in \mathbb{N}}$.

5.7.2 Proof of Lemma 5.1

We now give the proof of Lemma 5.1.

Let $V_0 := V$ and $V_i := V(T_{\leq i})$ for each $i \geq 1$. Clearly $V_0 \supseteq V_1 \supseteq \cdots$, and hence $\mathrm{Dis}(V_0) \supseteq \mathrm{Dis}(V_1) \supseteq \cdots$ as well.

Let E_i be the event in which the following hold:

1. Every $h \in V_i$ satisfies

$$\Pr_{x \sim D_{\mathcal{X}}}[h(x) \neq h^*(x) \;\wedge\; x \in \mathrm{Dis}(V_i)] \;\leq\; \sigma(d, 2^i, \delta_i/2)\,.$$

2. The number of LABEL queries in iteration i is at most

$$2^i \mu_i + O\left(\sqrt{2^i \mu_i \log(2/\delta_i)} + \log(2/\delta_i)\right),$$

where

$$\mu_i \;:=\; \theta_{V_{i-1}}(\epsilon) \cdot 2\sigma(d, 2^{i-1}, \delta_{i-1}/2).$$

We claim that $E_0 \cap E_1 \cap \cdots \cap E_i$ holds with probability at least $1 - \sum_{i'=1}^{i} \delta_{i'} \geq 1 - \delta$. The proof is by induction. The base case is trivial, as E_0 holds deterministically. For the inductive case, we just have to show that $\Pr(E_i \mid E_0 \cap E_1 \cap \cdots \cap E_{i-1}) \geq 1 - \delta_i$.

Condition on the event $E_0 \cap E_1 \cap \cdots \cap E_{i-1}$. For all $x \notin \mathrm{Dis}(V_{i-1})$, let $V_{i-1}(x)$ denote the label assigned by every $h \in V_{i-1}$ to x. Define

$$\hat{S}_i \;:=\; \left\{(x_{i,j}, \hat{y}_{i,j}) : j \in \{1, 2, \ldots, 2^i\},\; x_{i,j} \notin \mathrm{Dis}(V_{i-1}),\; \hat{y}_{i,j} = V_{i-1}(x_{i,j})\right\}.$$

Observe that $\hat{S}_i \cup T_i$ is an iid sample of size 2^i from a distribution (call it D_{i-1}) over labeled examples (x, y), where $x \sim D_{\mathcal{X}}$ and y is given by

$$y \;:=\; \begin{cases} V_{i-1}(x) & \text{if } x \notin \mathrm{Dis}(V_{i-1}), \\ h^*(x) & \text{if } x \in \mathrm{Dis}(V_{i-1}). \end{cases}$$

In fact, for any $h \in V_{i-1}$, we have

$$\mathrm{err}_{D_{i-1}}(h) \;=\; \Pr_{(x,y) \sim D_{i-1}}[h(x) \neq y] \;=\; \Pr_{x \sim D_{\mathcal{X}}}[h(x) \neq h^*(x) \;\wedge\; x \in \mathrm{Dis}(V_{i-1})]. \tag{5.2}$$

The VC inequality (Theorem A.1) implies that, with probability at least $1 - \delta_i/2$,

$$\forall h \in V \,.\, \Big(\mathrm{err}(h, \hat{S}_i \cup T_i) \;=\; 0 \;\implies\; \mathrm{err}_{D_{i-1}}(h) \;\leq\; \sigma(d, 2^i, \delta_i/2)\Big). \tag{5.3}$$

Consider any $h \in V_i$. We have $\mathrm{err}(h, T_i) = 0$ by definition of V_i. We also have $\mathrm{err}(h, \hat{S}_i) = 0$ since

$h \in V_i \subseteq V_{i-1}$. So in the event that (5.3) holds, we have

$$\begin{aligned}\Pr_{x\sim D_{\mathcal{X}}}[h(x) \neq h^*(x) \ \wedge \ x \in \mathrm{Dis}(V_i)] \ &\leq \ \Pr_{x\sim D_{\mathcal{X}}}[h(x) \neq h^*(x) \ \wedge \ x \in \mathrm{Dis}(V_{i-1})] \\ &= \ \mathrm{err}_{D_{i-1}}(h) \\ &\leq \ \sigma(d, 2^i, \delta_i/2)\,,\end{aligned}$$

where the first inequality follows because $\mathrm{Dis}(V_i) \subseteq \mathrm{Dis}(V_{i-1})$, and the equality follows from (5.2).

Now we prove the LABEL query bound.

Claim 5.1. *On event E_{i-1} for every $h, h' \in V_{i-1}$,*

$$\Pr_{x\sim D_{\mathcal{X}}}[h(x) \neq h'(x)] \leq \ 2\sigma(d, 2^{i-1}, \delta_{i-1}/2)$$

Proof. On event E_{i-1}, every $h \in V_{i-1}$ satisfies

$$\Pr_{x\sim D_{\mathcal{X}}}[h(x) \neq h^*(x), x \in \mathrm{Dis}(V_{i-1})] \ \leq \ \sigma(d, 2^{i-1}, \delta_{i-1}/2)\,.$$

Therefore, for any $h, h' \in V_{i-1}$, we have

$$\begin{aligned}\Pr_{x\sim D_{\mathcal{X}}}[h(x) \neq h'(x)] \ &= \ \Pr_{x\sim D_{\mathcal{X}}}[h(x) \neq h'(x), x \in \mathrm{Dis}(V_{i-1})] \\ &\leq \ \Pr_{x\sim D_{\mathcal{X}}}[h(x) \neq h^*(x), x \in \mathrm{Dis}(V_{i-1})] \\ &\quad + \Pr_{x\sim D_{\mathcal{X}}}[h'(x) \neq h^*(x), x \in \mathrm{Dis}(V_{i-1})] \\ &\leq \ 2\sigma(d, 2^{i-1}, \delta_{i-1}/2)\,.\end{aligned}$$

□

Since $2\sigma(d, 2^{i-1}, \delta_{i-1}/2) \geq \epsilon$, the above claim and the definition of the disagreement coefficient imply

$$\Pr_{x\sim D_{\mathcal{X}}}[x \in \mathrm{Dis}(V_{i-1})] \ \leq \ \theta_{V_{i-1}}(\epsilon) \cdot 2\sigma(d, 2^{i-1}, \delta_{i-1}/2) \ = \ \mu_i\,.$$

Therefore, μ_i is an upper bound on the probability that LABEL is queried on $x_{i,j}$, for each

$j = 1, 2, \ldots, 2^i$. By Lemma A.2, the number of queries to LABEL is at most

$$2^i \mu_i + O\left(\sqrt{2^i \mu_i \log(2/\delta_i)} + \log(2/\delta_i)\right).$$

with probability at least $1 - \delta_i/2$. We conclude by a union bound that $\Pr(E_i \mid E_0 \cap E_1 \cap \cdots \cap E_{i-1}) \geq 1 - \delta_i$ as required.

We now show that in the event $E_0 \cap E_1 \cap \cdots$, which holds with probability at least $1 - \delta$, the required consequences from Lemma 5.1 are satisfied. The definition of σ from Equation (A.1) and the halting condition in CAL imply that the number of iterations I executed by CAL satisfies

$$\sigma(d, 2^{I-1}, \delta_{I-1}/2) \;\geq\; \epsilon\,.$$

Thus by Fact A.1,

$$2^I \;\leq\; O\left(\frac{1}{\epsilon}\left(d \log \frac{1}{\epsilon} + \log \frac{1}{\delta}\right)\right),$$

which immediately gives the required bound on the number of unlabeled points drawn from $D_{\mathcal{X}}$. Moreover, I can be bounded as

$$I \;=\; O\left(\log(d/\epsilon) + \log\log(1/\delta)\right).$$

Therefore, in the event $E_0 \cap E_1 \cap \cdots \cap E_I$, CAL returns a set of labeled examples $T := T_{\leq I}$ in which every $h \in V(T)$ satisfies

$$\Pr_{x \sim D_{\mathcal{X}}}[h(x) \neq h^*(x) \;\wedge\; x \in \mathrm{Dis}(V(T))] \;\leq\; \epsilon\,,$$

and the number of LABEL queries is bounded by

$$\begin{aligned}
&\sum_{i=1}^{I}\left(2^i\mu_i + O\left(\sqrt{2^i\mu_i\log(2/\delta_i)} + \log(2/\delta_i)\right)\right) \\
&= \sum_{i=1}^{I} O\left(2^i\cdot\left(\theta_{V_{i-1}}(\epsilon)\frac{d\log 2^i + \log(2/\delta_i)}{2^i}\right) + \log(2/\delta_i)\right) \\
&= \sum_{i=1}^{I} O\left(\theta_{V_{i-1}}(\epsilon)\cdot\left(d\cdot i + \log(1/\delta)\right)\right) \\
&= O\left(\theta_V(\epsilon)\cdot\left(d\cdot\left(\log(d/\epsilon) + \log\log(1/\delta)\right)^2 + \left(\log(d/\epsilon) + \log\log(1/\delta)\right)\cdot\log(1/\delta)\right)\right) \\
&= \tilde{O}\left(\theta_V(\epsilon)\cdot d\cdot\log^2(1/\epsilon)\right)
\end{aligned}$$

as claimed. □

5.8 An Improved Algorithm for the Realizable Case

In this section, we present an improved algorithm for using SEARCH and LABEL in the realizable section. We call this algorithm SEABEL (Algorithm 5.4).

5.8.1 Description of Seabel

SEABEL proceeds in iterations like LARCH, but takes more advantage of SEARCH. Each iteration is split into two stages: the verification stage, and the sampling stage. In the verification stage, SEABEL makes repeated calls to SEARCH to advance to as high of a complexity class as possible, until $\perp$ is returned. When $\perp$ is returned, it guarantees that whenever the latest candidate set completely agrees on an unlabeled point, then it is also in agreement with h^*, even if it does not contain h^*.

5.8.2 Proof of Theorem 5.2

Observe that T_{i+1} is an iid sample of size 2^{i+1} from a distribution—call it D_i—over labeled examples (x, y), where $x \sim D_{\mathcal{X}}$, and

$$y := \begin{cases} V_i^{k_i}(x) & \text{if } x \notin \mathrm{Dis}(V_i^{k_i}), \\ h^*(x) & \text{if } x \in \mathrm{Dis}(V_i^{k_i}), \end{cases}$$

Algorithm 5.4 SEABEL

Input: Nested hypothesis classes $H_0 \subseteq H_1 \subseteq \cdots$; oracles SEARCH and LABEL; learning parameters $\epsilon, \delta \in (0,1)$

initialize $S_0 \leftarrow \emptyset$, $k_0 \leftarrow 0$.
Draw $x_{1,1}, x_{1,2}$ at random from $D_{\mathcal{X}}$, $T_1 \leftarrow \{(x_{1,1}, \textsc{Label}(x_{1,1})), (x_{1,2}, \textsc{Label}(x_{1,2}))\}$
for iteration $i = 1, 2, \ldots$ **do**
  $S \leftarrow S_{i-1}$, $k \leftarrow \min\{k' \geq k_{i-1} : H_{k'}(S_{i-1} \cup T_i) \neq \emptyset\}$ # Verification stage (Steps 4–13)
  loop
    $e \leftarrow \textsc{Search}_{H_k}(H_k(S \cup T_i))$
    if $e \neq \perp$ **then**
      $S \leftarrow S \cup \{e\}$
      $k \leftarrow \min\{k' > k : H_{k'}(S \cup T_i) \neq \emptyset\}$
    else
      break
    end if
  end loop
  $S_i \leftarrow S$, $k_i \leftarrow k$ # Sampling stage (Steps 14–24)
  Define new candidate set $V_i^{k_i} = H_{k_i}(S_i \cup T_i)$
  $T_{i+1} \leftarrow \emptyset$
  for $j = 1, 2, \ldots, 2^{i+1}$ **do**
    $x_{i+1,j} \leftarrow$ independent draw from $D_{\mathcal{X}}$ (the corresponding label is hidden)
    if $x_{i+1,j} \in \mathrm{Dis}(V_i^{k_i})$ **then**
      $T_{i+1} \leftarrow T_{i+1} \cup \{(x_{i+1,j}, \textsc{Label}(x_{i+1,j}))\}$
    else
      $T_{i+1} \leftarrow T_{i+1} \cup \{(x_{i+1,j}, V_i^k(x_{i+1,j}))\}$
    end if
  end for
  if $\sigma_{k_i}(2^{i+1}, \delta_{i+1,k_i}) \leq \epsilon$ **then**
    return any $\hat{h} \in V_i^{k_i}(T_{i+1})$
  end if
end for

for every x in the support of $D_{\mathcal{X}}$. (T_1 is an iid sample from $D_0 := D$; set $k_0 := 0$ and $S_0 := \emptyset$.)

Lemma 5.2. *Algorithm 5.4 maintains the following invariants:*

1. *The loop in the verification stage of iteration i terminates for all $i \geq 1$.*
2. *$k_i \leq k^*$ for all $i \geq 0$.*
3. *$h^*(x) = V_i^{k_i}(x)$ for all $x \notin \mathrm{Dis}(V_i^{k_i})$ for all $i \geq 1$.*
4. *h^* is consistent with $S_i \cup T_{i+1}$ for all $i \geq 0$.*

Proof. It is easy to see that S only contains examples provided by SEARCH, and hence the labels are consistent with h^*.

Now we prove that the invariants hold by induction on i, starting with $i = 0$. For the base case, only the last invariant needs to be checked, and it is true because the labels in T_1 are obtained from LABEL.

For the inductive step, fix any $i \geq 1$, and assume that $k_{i-1} \leq k^*$, and that h^* is consistent with T_i. Now consider the verification stage in iteration i. We first prove that the loop in the verification stage will terminate and establish some properties upon termination. Observe that k and S are initially k_{i-1} and S_{i-1}, respectively. Throughout the loop, the examples added to S are obtained from SEARCH, and hence are consistent with h^*. Thus, $h^* \in H_{k^*}(S \cup T_i)$, implying $H_{k^*}(S \cup T_i) \neq \emptyset$. If $k = k^*$, then $\text{SEARCH}_{H_{k^*}}(H_{k^*}(S \cup T_i))$ would return $\perp$ and Algorithm 5.4 would exit the loop. If $\text{SEARCH}_{H_k}(H_k(S \cup T_i)) \neq \perp$, then $k < k^*$, and k cannot be increased beyond k^* since $H_{k^*}(S \cup T_i) \neq \emptyset$. Thus, the loop must terminate with $k \leq k^*$, implying $k_i \leq k^*$. Moreover, because the loop terminates with $\text{SEARCH}_{H_k}(H_k(S \cup T_i))$ returning $\perp$ (and here, $k = k_i$ and $H_k(S \cup T_i) = V_i^{k_i}$), there is no counterexample $x \in \mathcal{X}$ such that h^* disagrees with every $h \in V_i^{k_i}$. This implies that $h^*(x) = V_i^{k_i}(x)$ for all $x \notin \text{Dis}(V_i^{k_i})$.

Now consider any (x, y) added to T_{i+1} in the sampling stage. If $x \in \text{Dis}(V_i^{k_i})$, the label is obtained from LABEL, and hence is consistent with h^*; if $x \notin \text{Dis}(V_i^{k_i})$, the label is $V_i^{k_i}(x)$, which is the same as $h^*(x)$ as previously argued. So h^* is consistent with all examples in T_{i+1}, and hence also all examples in $S_i \cup T_{i+1}$. □

Let E_i be the event in which the following hold:

1. For every $k \geq 0$, every $h \in H_k$ satisfies

$$\text{err}(h, D_{i-1}) \;\leq\; \text{err}(h, T_i) + \sqrt{\text{err}(h, T_i)\sigma_k(2^i, \delta_{i,k})} + \sigma_k(2^i, \delta_{i,k})\,.$$

2. The number of LABEL queries in iteration i (to form T_{i+1}) is at most

$$2^{i+1} \Pr_{x \sim D_{\mathcal{X}}}[x \in \text{Dis}(V_i^{k_i})] + O\left(\sqrt{2^{i+1} \Pr_{x \sim D_{\mathcal{X}}}[x \in \text{Dis}(V_i^{k_i})]\log(1/\delta_i)} + \log(1/\delta_i)\right),$$

Using Theorem A.1 and Lemma A.2, along with the union bound, $\Pr(E_i) \geq 1 - \delta_i$. Define $E := \cap_{i=1}^{\infty} E_i$; a union bound implies that $\Pr(E) \geq 1 - \delta$.

We now prove Theorem 5.2, starting with the error rate guarantee. Condition on the event E. Since $k_i \leq k^*$, the definition of σ_k from (5.1), the halting condition in Algorithm 5.4,

and Fact A.1 imply that the algorithm must halt after at most I iterations, where

$$2^I \;\leq\; O\left(\frac{1}{\epsilon}\left(d_{k^*}\log\frac{1}{\epsilon}+\log\frac{k^*}{\delta}\right)\right). \tag{5.4}$$

So let i denote the iteration in which Algorithm 5.4 halts. By definition of E_{i+1}, we have

$$\begin{aligned}\mathrm{err}(\hat{h},D_i) \;&\leq\; \mathrm{err}(\hat{h},T_{i+1})+\sqrt{\mathrm{err}(\hat{h},T_{i+1})\sigma_{k_i}(2^{i+1},\delta_{i+1,k_i})}+\sigma_{k_i}(2^{i+1},\delta_{i+1,k_i})\\ &=\; \sigma_{k_i}(2^{i+1},\delta_{i+1,k_i}) \;\leq\; \epsilon\,.\end{aligned}$$

By Lemma 5.2, $h^*(x)=V_i^{k_i}(x)$ for all $x\notin \mathrm{Dis}(V_i^{k_i})$. Therefore, $D(\cdot\mid x)=D_i(\cdot\mid x)$ for every x in the support of $D_{\mathcal{X}}$, and

$$\mathrm{err}(\hat{h},D) \;=\; \mathrm{err}(\hat{h},D_i) \;\leq\; \epsilon\,.$$

Now we bound the unlabeled, Label, and Search complexities, all conditioned on event E. First, as argued above, the algorithm halts after at most I iterations, where 2^I is bounded as in (5.4). The number of unlabeled examples drawn from $D_{\mathcal{X}}$ across all iterations is within a factor of two of the number of examples drawn in the final sampling stage, which is $O(2^I)$. Thus (5.4) also gives the bound on the number of unlabeled examples drawn.

Next, we consider the Search complexity. For each iteration i, each call to Search either returns a counterexample that forces k to increment (but never past k^*, as argued in Lemma 5.2), or returns $\perp$ which causes an exit from the verification stage loop. Therefore, the total number of Search calls is at most

$$k^*+I \;=\; k^*+O\left(\log\frac{d_{k^*}}{\epsilon}+\log\log\frac{k^*}{\delta}\right).$$

Finally, we consider the Label complexity. For $i\leq I$, we first show that the candidate set $V_i^{k_i}$ is always contained in a ball of small radius (with respect to the disagreement pseudometric). Specifically, for every h,h' in $V_i^{k_i}$, $\mathrm{err}(h,T_i)=0$ and $\mathrm{err}(h,T_i)=0$. By definition of E_i, this implies that

$$\mathrm{err}(h,D_{i-1}) \;\leq\; \sigma_{k_i}(2^i,\delta_{i,k_i}) \quad\text{and}\quad \mathrm{err}(h',D_{i-1}) \;\leq\; \sigma_{k_i}(2^i,\delta_{i,k_i}).$$

Therefore, by the triangle inequality,

$$\Pr_{x\sim D}[h(x)\neq h'(x)] \;\le\; 2\sigma_{k_i}(2^i,\delta_{i,k_i}) \;\le\; 2\sigma_{k^*}(2^i,\delta_{i,k^*}).$$

Note that because $2^I \le \tilde{O}(d_{k^*}/\epsilon)$, we have $\sigma_{k^*}(2^i,\delta_{i,k^*}) \ge \epsilon/2$ for $i \le I$. Thus, the size of the disagreement region can be bounded as

$$\Pr_{x\sim D_{\mathcal{X}}}[x\in \mathrm{Dis}(V_i^{k_i})] \quad\le\quad \theta_{k_i}(\epsilon)\cdot 2\sigma_{k^*}(2^i,\delta_{i,k^*}).$$

By definition of E_i, the number of queries to Label at iteration i is at most

$$2^{i+1}\Pr_{x\sim D_{\mathcal{X}}}[x\in \mathrm{Dis}(V_i^{k_i})] + O\left(\sqrt{2^{i+1}\Pr_{x\sim D_{\mathcal{X}}}[x\in \mathrm{Dis}(V_i^{k_i})]\log(1/\delta_{i,k})} + \log(1/\delta_i)\right),$$

which is at most

$$O\left(2^i\cdot\theta_{k_i}(\epsilon)\cdot\sigma_{k^*}(2^i,\delta_{i,k^*})\right).$$

We get that the total number of Label queries by Algorithm 5.4 is bounded by

$$\begin{aligned}
&2+\sum_{i=1}^{I} O\left(2^i\cdot\theta_{k_i}(\epsilon)\cdot\sigma_{k^*}(2^i,\delta_{i,k^*})\right)\\
&=\; 2+\sum_{i=1}^{I} O\left(2^i\cdot\max_{k\le k^*}\theta_k(\epsilon)\cdot\sigma_{k^*}(2^i,\delta_{i,k^*})\right)\\
&=\; O\left(\max_{k\le k^*}\theta_k(\epsilon)\cdot\left(\sum_{i=1}^{I}2^i\sigma_{k^*}(2^i,\delta_{i,k^*})\right)\right)\\
&=\; O\left(\max_{k\le k^*}\theta_k(\epsilon)\cdot\left(\sum_{i=1}^{I}2^i\frac{d\ln(2^i)+\ln(\frac{(i^2+i)(k^*)^2}{\delta})}{2^i}\right)\right)\\
&=\; O\left(\max_{k\le k^*}\theta_k(\epsilon)\cdot\left(d_{k^*}I^2+I\log\frac{k^*}{\delta}\right)\right)\\
&=\; O\left(\max_{k\le k^*}\theta_k(\epsilon)\cdot\left(d_{k^*}\left(\log\frac{d_{k^*}}{\epsilon}+\log\log\frac{k^*}{\delta}\right)^2+\left(\log\frac{d_{k^*}}{\epsilon}+\log\log\frac{k^*}{\delta}\right)\log\frac{k^*}{\delta}\right)\right)\\
&=\; \tilde{O}\left(\max_{k\le k^*}\theta_k(\epsilon)\cdot\left(d_{k^*}\cdot\log^2\frac{1}{\epsilon}+\log k^*\right)\right)
\end{aligned}$$

as claimed. □

5.9 A-Larch: An Adaptive Agnostic Algorithm

Below we present A-LARCH, an adaptive agnostic algorithm.

5.9.1 Description of A-Larch

A-LARCH proceeds in iterations like SEABEL. Each iteration is split into three stages: the error estimation stage, the verification stage, and the sampling stage.

In the error estimation stage, A-LARCH uses a structural risk minimization approach (Steps 4–5) to compute γ_{i-1}, a (tight) upper bound on $\Pr[h^*(x) \neq y, x \in \mathrm{Dis}(V_{i-1})]$.

In the verification stage, A-LARCH makes repeated calls to SEARCH to advance to as high of a complexity class as possible, until $\perp$ is returned (Steps 6–19). When $\perp$ is returned, it guarantees that whenever the latest candidate set completely agrees on an unlabeled point, then it is also in agreement with h^*, even if it does not contain h^*.

In the sampling stage, A-LARCH performs sampling (Steps 21–29), querying and inferring labels based on disagreement over the verified candidate set $V_i^{k_i}$.

5.9.2 Proof of Theorem 5.3

Let

$$\begin{aligned} M(\nu, k^*, \epsilon, \delta) &:= \min\left\{2^n : n \in \mathbb{N}, 3\sqrt{\nu\sigma_{k^*}(2^n, \delta_{n,k^*})} + 4\sigma_{k^*}(2^n, \delta_{n,k^*}) \leq \epsilon\right\} \\ &= O\left(\frac{(d_{k^*}\log(1/\epsilon) + \log(k^*/\delta))(\nu+\epsilon)}{\epsilon^2}\right). \end{aligned}$$

where the second line is from Fact A.1.

Theorem 5.5 (Restatement of Theorem 5.3)**.** *Assume* $\mathrm{err}(h^*) = \nu$*. If Algorithm 5.5 is run with inputs hypothesis classes* $\{H_k\}_{k=0}^{\infty}$*, oracles* SEARCH *and* LABEL*, learning parameter* δ*, unlabeled examples budget* $m = M(\nu, k^*, \epsilon, \delta)$ *and the disagreement coefficient of* $H_k(S)$ *is at most* $\theta_k(\cdot)$*, then, with probability* $1-\delta$*:*

(1) The returned hypothesis $\hat{h}$ *satisfies*

$$\mathrm{err}(\hat{h}) \leq \nu + \epsilon \ .$$

Algorithm 5.5 A-LARCH

Input: Nested hypothesis set $H_0 \subseteq H_1 \subseteq \cdots$; oracles LABEL and SEARCH; learning parameter $\delta \in (0,1)$; unlabeled examples budget $m = 2^{I+2}$.

Output: hypothesis $\hat{h}$.

Initialize: $S \leftarrow \emptyset$, $k \leftarrow 0$, $k_0 \leftarrow k$.
Draw $x_{1,1}, x_{1,2}$ at random from $D_{\mathcal{X}}$, $T_1 \leftarrow \{(x_{1,1}, \textsc{Label}(x_{1,1})), (x_{1,2}, \textsc{Label}(x_{1,2}))\}$
for $i = 1, 2, \ldots, I$ **do**
  $(\hat{h}_i, \hat{k}_i) \leftarrow \arg\min_{k' \geq k, h \in H_{k'}} \left\{ \mathrm{err}(h, T_i) + \sqrt{\mathrm{err}(h, T_i)\sigma_{k'}(2^i, \delta_{i,k'})} + \sigma_{k'}(2^i, \delta_{i,k'}) \right\}$ # Error estimation stage (Steps 4–5)
  $\gamma_{i-1} \leftarrow \mathrm{err}(\hat{h}_i, T_i) + \sqrt{\mathrm{err}(\hat{h}_i, T_i)\sigma_{\hat{k}_i}(2^i, \delta_{i,\hat{k}_i})} + \sigma_{\hat{k}_i}(2^i, \delta_{i,\hat{k}_i})$
  loop # Verification stage (Steps 6–19)
    if $\min_{h \in H_k(S)} \mathrm{err}(h, T_i) > \gamma_{i-1} + \sqrt{\gamma_{i-1}\sigma_k(2^i, \delta_{i,k})} + \sigma_k(2^i, \delta_{i,k})$ **then**
      $k \leftarrow k + 1$
    else

$$V_i^k \leftarrow \Big\{ h \in H_k(S) : \mathrm{err}(h, T_i) \leq \min_{h' \in H_k(S)} \mathrm{err}(h', T_i) + 3\sqrt{\mathrm{err}(h', T_i)\sigma_k(2^i, \delta_{i,k})} + 4\sigma_k(2^i, \delta_{i,k}) \Big\}$$

      $e \leftarrow \textsc{Search}_{H_k}(V_i^k)$
      if $e \neq \perp$ **then**
        $S \leftarrow S \cup \{e\}$
        $k \leftarrow \min\{k' > k : H_{k'}(S) \neq \emptyset\}$
      else
        break
      end if
    end if
  end loop
  $S_i \leftarrow S$, $k_i \leftarrow k$
  $T_{i+1} \leftarrow \emptyset$ # Sampling stage (Steps 21–29)
  for $j = 1, 2, \ldots, 2^{i+1}$ **do**
    $x_{i+1,j} \leftarrow$ independent draw from $D_{\mathcal{X}}$ (the corresponding label is hidden)
    if $x_{i+1,j} \in \mathrm{Dis}(V_i^k)$ **then**
      $T_{i+1} \leftarrow T_{i+1} \cup \{(x_{i+1,j}, \textsc{Label}(x_{i+1,j}))\}$
    else
      $T_{i+1} \leftarrow T_{i+1} \cup \{(x_{i+1,j}, V_i^k(x_{i+1,j}))\}$
    end if
  end for
end for
return $\hat{h}_I$

(2) The total number of queries to oracle SEARCH *is at most*

$$k^* + \log m \le k^* + O\left(\log \frac{d_{k^*}}{\epsilon} + \log\log \frac{k^*}{\delta}\right).$$

(3) The total number of queries to oracle LABEL *is at most*

$$\tilde{O}\left(\max_{k \le k^*} \theta_k(2\nu + 2\epsilon) \cdot d_{k^*} \left(\log \frac{1}{\epsilon}\right)^2 \cdot \left(1 + \frac{\nu^2}{\epsilon^2}\right)\right) .$$

The proof relies on an auxiliary lemma. First, we need to introduce the following notation.

Observe that T_{i+1} is an iid sample of size 2^{i+1} from a distribution (call it D_i) over labeled examples (x, y), where $x \sim D_{\mathcal{X}}$ and the conditional distribution is

$$D_i(y \mid x) \;:=\; \begin{cases} \mathbb{1}\{y = V_i^{k_i}(x)\} & \text{if } x \notin \mathrm{Dis}(V_i^{k_i}), \\ D(y \mid x) & \text{if } x \in \mathrm{Dis}(V_i^{k_i}), \end{cases}$$

T_1 is a sample of size 2 from $D_0 = D$.

Let E_i be the event in which the following hold:

1. For every $k \ge 0$, every $h \in H_k$ satisfies

$$\begin{aligned} \mathrm{err}(h, D_{i-1}) \;&\le\; \mathrm{err}(h, T_i) + \sqrt{\mathrm{err}(h, T_i)\sigma_k(2^i, \delta_{i,k})} + \sigma_k(2^i, \delta_{i,k}), \\ \mathrm{err}(h, T_i) \;&\le\; \mathrm{err}(h, D_{i-1}) + \sqrt{\mathrm{err}(h, D_{i-1})\sigma_k(2^i, \delta_{i,k})} + \sigma_k(2^i, \delta_{i,k}). \end{aligned}$$

2. The number of LABEL queries at iteration i is at most

$$2^{i+1} \Pr_{x \sim D_{\mathcal{X}}}[x \in \mathrm{Dis}(V_i^{k_i})] + O\left(\sqrt{2^{i+1} \Pr_{x \sim D_{\mathcal{X}}}[x \in \mathrm{Dis}(V_i^{k_i})] \log(1/\delta_i)} + \log(1/\delta_i)\right).$$

Using Theorem A.1 and Lemma A.2, along with the union bound, $\Pr(E_i) \ge 1 - \delta_i$. Define $E := \cap_{i=1}^{\infty} E_i$, by union bound, $\Pr(E) \ge 1 - \delta$.

Lemma 5.3. *On event E, Algorithm 5.5 maintains the following invariants:*

1. *For all $i \geq 1$, γ_{i-1} is such that*

$$\mathrm{err}(h^*, D_{i-1}) \leq \gamma_{i-1} \leq \mathrm{err}(h^*, D_{i-1}) + 3\sqrt{\mathrm{err}(h^*, D_{i-1})\sigma_{k^*}(2^i, \delta_{i,k^*})} + 4\sigma_{k^*}(2^i, \delta_{i,k^*}).$$

2. *The loop in the verification stage of iteration i terminates for all $i \geq 1$.*

3. *$k_i \leq k^*$ for all $i \geq 0$.*

4. *$h^*(x) = V_i^{k_i}(x)$ for all $x \notin \mathrm{Dis}(V_i^{k_i})$ for all $i \geq 1$.*

5. *For all $i \geq 0$, for every hypothesis h, $\mathrm{err}(h, D_i) - \mathrm{err}(h^*, D_i) \geq \mathrm{err}(h, D) - \mathrm{err}(h^*, D)$. Therefore, h^* is the optimal hypothesis among $\cup_k H_k$ with respect to D_i.*

Proof. Throughout, we assume the event E holds.

It is easy to see that S only contains examples provided by SEARCH, and hence the labels are consistent with h^*.

Now we prove that the invariants hold by induction on i, starting with $i = 0$. For the base case, invariant 3 holds since $k_0 = 0 \leq k^*$, and invariant 5 holds since $D_0 = D$ and h^* is the optimal hypothesis in $\cup_k H_k$.

Now consider the inductive step. We first prove that invariant 1 holds.

(1) By definition of E_i, for all $k' \geq k_{i-1}$, we have for all $h \in H_{k'}$,

$$\mathrm{err}(h, D_{i-1}) \leq \mathrm{err}(h, T_i) + \sqrt{\mathrm{err}(h, T_i)\sigma_{k'}(2^i, \delta_{i,k'})} + \sigma_{k'}(2^i, \delta_{i,k'}).$$

Thus,

$$\min_{h \in H_{k'}} \mathrm{err}(h, D_{i-1}) \leq \min_{h \in H_{k'}} \mathrm{err}(h, T_i) + \sqrt{\mathrm{err}(h, T_i)\sigma_{k'}(2^i, \delta_{i,k'})} + \sigma_{k'}(2^i, \delta_{i,k'}).$$

Taking minimum over $k' \geq k_{i-1}$ on both sides, notice that h^* is the optimal hypothesis with respect to D_{i-1} and recall the definition of γ_{i-1}, we get

$$\mathrm{err}(h^*, D_{i-1}) \leq \gamma_{i-1}.$$

(2) By definition of γ_{i-1}, we have

$$\gamma_{i-1} = \min_{k' \geq k_{i-1}, h \in H_{k'}} \left\{ \mathrm{err}(h, T_i) + \sqrt{\mathrm{err}(h, T_i)\sigma_{k'}(2^i, \delta_{i,k'})} + \sigma_{k'}(2^i, \delta_{i,k'}) \right\}$$

Taking $k' = k^*$, $h = h^*$, we get

$$\gamma_{i-1} \leq \mathrm{err}(h^*, T_i) + \sqrt{\mathrm{err}(h^*, T_i)\sigma_{k^*}(2^i, \delta_{i,k^*})} + \sigma_{k^*}(2^i, \delta_{i,k^*})$$

In conjunction with the fact that by definition of E_i,

$$\mathrm{err}(h^*, T_i) \leq \mathrm{err}(h^*, D_{i-1}) + \sqrt{\mathrm{err}(h^*, D_{i-1})\sigma_{k^*}(2^i, \delta_{i,k^*})} + \sigma_{k^*}(2^i, \delta_{i,k^*})$$

We get

$$\gamma_{i-1} \leq \mathrm{err}(h^*, D_{i-1}) + 3\sqrt{\mathrm{err}(h^*, D_{i-1})\sigma_{k^*}(2^i, \delta_{i,k^*})} + 4\sigma_{k^*}(2^i, \delta_{i,k^*}).$$

Thus, invariant 1 is established.

Now consider the verification stage in iteration i. We first prove that the loop in the verification stage will terminate and establish some properties upon termination. Observe that k and S are initially k_{i-1} and S_{i-1}, respectively. Throughout the loop, the examples added to S are obtained from SEARCH, and hence are consistent with h^*. In addition, we have the following claim regarding k^*.

Claim 5.2. *If invariants 1–5 holds for iteration $i-1$, then for iteration i, the following holds:*

(a) $\min_{h \in H_{k^*}(S)} \mathrm{err}(h, T_i) \leq \gamma_{i-1} + \sqrt{\gamma_{i-1}\sigma_{k^*}(2^i, \delta_{i,k^*})} + \sigma_{k^*}(2^i, \delta_{i,k^*})$

(b) h^* *is in* $V_i^{k^*}$*, where* $V_i^{k^*}$ *is defined as:*

$$\left\{ h \in H_{k^*}(S) : \mathrm{err}(h, T_i) \leq \min_{h' \in H_{k^*}} \mathrm{err}(h', T_i) + 3\sqrt{\mathrm{err}(h', T_i)\sigma_{k^*}(2^i, \delta_{i,k^*})} + 4\sigma_{k^*}(2^i, \delta_{i,k^*}) \right\}$$

Proof. Recall that h^* is the optimal hypothesis under distribution D_{i-1}. By the definition of E_i,

$$\begin{aligned}\min_{h' \in H_{k^*}(S)} \operatorname{err}(h', T_i) &\leq \operatorname{err}(h^*, T_i) \\ &\leq \operatorname{err}(h', D_{i-1}) + \sqrt{\operatorname{err}(h', D_{i-1})\sigma(2^i, \delta_{i,k^*})} + \sigma(2^i, \delta_{i,k^*}) \\ &\leq \gamma_{i-1} + \sqrt{\gamma_{i-1}\sigma_{k^*}(2^i, \delta_{i,k^*})} + \sigma_{k^*}(2^i, \delta_{i,k^*})\end{aligned}$$

where the last inequality is from invariant 1. This proves item (a).

On the other hand, for all h' in $H_k(S)$,

$$\begin{aligned}\operatorname{err}(h^*, T_i) &\leq \operatorname{err}(h^*, D_{i-1}) + \sqrt{\operatorname{err}(h^*, D_{i-1})\sigma(2^i, \delta_{i,k^*})} + \sigma(2^i, \delta_{i,k^*}) \\ &\leq \operatorname{err}(h', D_{i-1}) + \sqrt{\operatorname{err}(h', D_{i-1})\sigma(2^i, \delta_{i,k^*})} + \sigma(2^i, \delta_{i,k^*}) \\ &\leq \operatorname{err}(h', T_i) + 3\sqrt{\operatorname{err}(h', T_i)\sigma(2^i, \delta_{i,k^*})} + 4\sigma(2^i, \delta_{i,k^*}).\end{aligned}$$

Thus, $\operatorname{err}(h^*, T_i) \leq \min_{h' \in H_k(S)} \operatorname{err}(h', T_i) + 3\sqrt{\operatorname{err}(h', T_i)\sigma(2^i, \delta_{i,k^*})} + 4\sigma(2^i, \delta_{i,k^*})$, proving item (b). □

Thus, if $k = k^*$, then Claim 5.2(a) implies that line 7 will never be satisfied. In addition, Claim 5.2(b) says that $h^* \in V_i^{k^*}$, implying $V_i^{k^*} \neq \emptyset$, then $\textsc{Search}_{H_{k^*}}(V_i^{k^*})$ would return $\perp$ and Algorithm 5.5 would exit the loop. If $\textsc{Search}_{H_k}(V_i^k) \neq \perp$, then $k < k^*$, and k cannot be increased beyond k^* since $V_i^{k^*} \neq \emptyset$. Thus, the loop must terminate with $k \leq k^*$, implying $k_i \leq k^*$. This establishes invariants 2 and 3.

Moreover, because the loop terminates with $\textsc{Search}_{H_k}(V_i^{k_i})$ returning $\perp$, there is no counterexample $x \in \mathcal{X}$ such that h^* disagrees with every $h \in V_i^{k_i}$. This implies that $h^*(x) = V_i^{k_i}(x)$ for all $x \notin \operatorname{Dis}(V_i^{k_i})$ (i.e., invariant 4). Hence, for any hypothesis h,

$$\operatorname{err}(h, D_i) \;=\; \Pr[h(x) \neq h^*(x), x \in \operatorname{Dis}(V_i^{k_i})] + \Pr[h(x) \neq y, x \notin \operatorname{Dis}(V_i^{k_i})].$$

Therefore,

$$\begin{aligned}
&\operatorname{err}(h, D_i) - \operatorname{err}(h^*, D_i) \\
&= \Pr[h(x) \neq h^*(x), x \in \operatorname{Dis}(V_i^{k_i})] + \Pr[h(x) \neq y, x \notin \operatorname{Dis}(V_i^{k_i})] \\
&\quad - \Pr[h^*(x) \neq y, x \notin \operatorname{Dis}(V_i^{k_i})] \\
&\geq \Pr[h(x) \neq y, x \in \operatorname{Dis}(V_i^{k_i})] - \Pr[h^*(x) \neq y, x \in \operatorname{Dis}(V_i^{k_i})] \\
&\quad + \Pr[h(x) \neq y, x \notin \operatorname{Dis}(V_i^{k_i})] - \Pr[h^*(x) \neq y, x \notin \operatorname{Dis}(V_i^{k_i})] \\
&= \operatorname{err}(h, D) - \operatorname{err}(h^*, D),
\end{aligned}$$

which proves invariant 5. □

Proof of Theorem 5.5. Suppose event E happens.

We first prove the error rate guarantee. Suppose iteration $i = I$ has been reached. Observe that by definition of E_I,

$$\operatorname{err}(\hat{h}_I, D_{I-1}) \leq \operatorname{err}(\hat{h}_I, T_I) + \sqrt{\operatorname{err}(\hat{h}_I, T_I)\sigma_{\hat{k}_I}(2^I, \delta_{I,\hat{k}_I})} + \sigma_{\hat{k}_I}(2^I, \delta_{I,\hat{k}_I}) = \gamma_{I-1}$$

Combining with item 1 of Lemma 5.3, which states that

$$\gamma_{I-1} \leq \operatorname{err}(h^*, D_{I-1}) + 3\sqrt{\operatorname{err}(h^*, D_{I-1})\sigma_{k^*}(2^I, \delta_{I,k^*})} + 4\sigma_{k^*}(2^I, \delta_{I,k^*})$$

we have

$$\begin{aligned}
\operatorname{err}(\hat{h}_I, D_{I-1}) - \operatorname{err}(h^*, D_{I-1}) &\leq 3\sqrt{\operatorname{err}(h^*, D_{I-1})\sigma_{k^*}(2^I, \delta_{I,k^*})} + 4\sigma_{k^*}(2^I, \delta_{I,k^*}) \\
&\leq 3\sqrt{\nu\sigma_{k^*}(2^I, \delta_{I,k^*})} + 4\sigma_{k^*}(2^I, \delta_{I,k^*}) \\
&\leq \epsilon
\end{aligned}$$

where the second inequality is from that $\operatorname{err}(h^*, D_{I-1}) \leq \operatorname{err}(h^*, D) = \nu$, the third inequality is from that $m = 2^I = M(\nu, k^*, \epsilon, \delta)$. Thus, by item 5 of Lemma 5.3,

$$\operatorname{err}(\hat{h}_I, D) - \operatorname{err}(h^*, D) \leq \operatorname{err}(\hat{h}_I, D_{I-1}) - \operatorname{err}(h^*, D_{I-1}) \leq \epsilon.$$

Next, we prove the bound on the number of SEARCH queries. From Lemma 5.3, Algo-

rithm 5.5 maintains the invariant that $k \le k^*$. For each iteration i, each call to SEARCH either returns an example forcing k to increment, or returns $\perp$ which causes an exit from the verification stage loop. Therefore, the total number of SEARCH calls is at most

$$k^* + I \le k^* + O\left(\log \frac{d_{k^*}}{\epsilon^2} + \log\log \frac{k^*}{\delta}\right).$$

Finally, we prove the bound on the number of LABEL queries. This is done in a few steps.

1. We first show that the candidate set $V_i^{k_i}$ is always contained in a ball of small radius (with respect to the disagreement pseudometric $\Pr_{x\sim D_{\mathcal{X}}}[h(x) \ne h'(x)]$). Specifically, the following holds.

   Claim 5.3. *On event E, for every $1 \le i \le I$, and all $h, h' \in V_i^{k_i}$,*

   $$\Pr_{(x,y)\sim D}[h(x) \ne h'(x)] \le 2\gamma_{i-1} + 16\sqrt{\gamma_{i-1}\sigma_{k_i}(2^i, \delta_{i,k_i})} + 30\sigma_{k_i}(2^i, \delta_{i,k_i}).$$

   Proof. First, for every h in $V_i^{k_i}$,

   $$\mathrm{err}(h, T_i) \le \min_{h' \in H_{k_i}(S)} \mathrm{err}(h', T_i) + 3\sqrt{\mathrm{err}(h', T_i)\sigma_{k_i}(2^i, \delta_{i,k_i})} + 4\sigma_{k_i}(2^i, \delta_{i,k_i}),$$

   and since the condition in step 7 is not satisfied for $k = k_i$, we know that

   $$\min_{h' \in H_{k_i}(S)} \mathrm{err}(h', T_i) \le \gamma_{i-1} + \sqrt{\gamma_{i-1}\sigma_{k_i}(2^i, \delta_{i,k_i})} + \sigma_{k_i}(2^i, \delta_{i,k_i}).$$

   Thus,

   $$\mathrm{err}(h, T_i) \le \gamma_{i-1} + 6\sqrt{\gamma_{i-1}\sigma_{k_i}(2^i, \delta_{i,k_i})} + 10\sigma_{k_i}(2^i, \delta_{i,k_i}). \tag{5.5}$$

   By definition of event E_i, we also have

   $$\mathrm{err}(h, D_{i-1}) \le \mathrm{err}(h, T_i) + \sqrt{\mathrm{err}(h, T_i)\sigma_{k_i}(2^i, \delta_{i,k_i})} + \sigma_{k_i}(2^i, \delta_{i,k_i}).$$

   Hence,

   $$\mathrm{err}(h, D_{i-1}) \le \gamma_{i-1} + 8\sqrt{\gamma_{i-1}\sigma_{k_i}(2^i, \delta_{i,k_i})} + 15\sigma_{k_i}(2^i, \delta_{i,k_i}).$$

Therefore, for any $h, h' \in V_i^{k_i}$, we have

$$\begin{aligned}\Pr_{x\sim D_{\mathcal{X}}}[h(x) \neq h'(x)] &\leq \operatorname{err}(h, D_{i-1}) + \operatorname{err}(h', D_{i-1}) \\ &\leq 2\gamma_{i-1} + 16\sqrt{\gamma_{i-1}\sigma_{k_i}(2^i, \delta_{i,k_i})} + 30\sigma_{k_i}(2^i, \delta_{i,k_i}).\end{aligned}$$

The claim follows. □

2. Next we bound the label complexity per iteration. For $1 \leq i \leq I$, by item 1 of Lemma 5.3,

$$\begin{aligned}\gamma_{i-1} &\leq \operatorname{err}(h^*, D_{i-1}) + 3\sqrt{\operatorname{err}(h^*, D_{i-1})\sigma_{k^*}(2^i, \delta_{i,k^*})} + 4\sigma_{k^*}(2^i, \delta_{i,k^*}) \\ &\leq \nu + 3\sqrt{\nu\sigma_{k^*}(2^i, \delta_{i,k^*})} + 4\sigma_{k^*}(2^i, \delta_{i,k^*}).\end{aligned}$$

Now for $h, h' \in V_i^{k_i}$,

$$\begin{aligned}\Pr_{x\sim D_{\mathcal{X}}}[h(x) \neq h'(x)] &\leq 2\gamma_{i-1} + 16\sqrt{\gamma_{i-1}\sigma_{k_i}(2^i, \delta_{i,k_i})} + 30\sigma_{k_i}(2^i, \delta_{i,k_i}) \\ &\leq 2\gamma_{i-1} + 16\sqrt{\gamma_{i-1}\sigma_{k^*}(2^i, \delta_{i,k^*})} + 30\sigma_{k_i}(2^i, \delta_{i,k^*}) \\ &\leq 2\nu + 38\sqrt{\nu\sigma_{k^*}(2^i, \delta_{i,k^*})} + 86\sigma_{k_i}(2^i, \delta_{i,k^*}).\end{aligned}$$

Thus, $V_i^{k_i}$ is contained in $\mathrm{B}_{H_{k_i}(S)}(h, 2\nu + 38\sqrt{\nu\sigma_{k^*}(2^i, \delta_{i,k^*})} + 86\sigma_{k^*}(2^i, \delta_{i,k^*}))$ for some h in $H_{k_i}(S)$.

Note that by the choice of $m = 2^I = M(\nu, k^*, \epsilon, \delta)$, $38\sqrt{\nu\sigma_{k^*}(2^i, \delta_{i,k^*})} + 86\sigma_{k^*}(2^i, \delta_{i,k^*}) \geq 2\epsilon$ for $i \leq I$. Thus, the size of the disagreement region can be bounded as

$$\begin{aligned}\Pr_{x\sim D_{\mathcal{X}}}[x \in \operatorname{Dis}(V_i)] &\leq \theta_k(2\nu + 2\epsilon) \cdot \left(2\nu + 38\sqrt{\nu\sigma_{k^*}(2^i, \delta_{i,k^*})} + 86\sigma_{k^*}(2^i, \delta_{i,k^*})\right) \\ &\leq \theta_k(2\nu + 2\epsilon) \cdot \left(21\nu + 105\sigma_{k^*}(2^i, \delta_{i,k^*})\right). \qquad (5.6)\end{aligned}$$

By definition of E_i, the number of queries to Label at iteration i is at most

$$2^{i+1}\Pr_{x\sim D_{\mathcal{X}}}[x \in \operatorname{Dis}(V_i^{k_i})] + O\left(\sqrt{2^{i+1}\Pr_{x\sim D_{\mathcal{X}}}[x \in \operatorname{Dis}(V_i^{k_i})]\log(1/\delta_{i,k^*})} + \log(1/\delta_{i,k^*})\right).$$

Combining this with (5.6) gives

$$\#\ \textsc{Label}\text{ queries in iteration } i \ = \ O\left(2^i \cdot \theta_{k_i}(2\nu+2\epsilon)\cdot(\nu+\sigma_{k^*}(2^i,\delta_{i,k^*}))\right). \tag{5.7}$$

3. From the setting of $m = 2^I = \tilde{O}(d_{k^*}(\nu+\epsilon)/\epsilon^2)$, we get that

$$I = O\left(\log\frac{d_{k^*}}{\epsilon} + \log\log\frac{k^*}{\delta}\right).$$

Now, using (5.7), we get that the total number of $\textsc{Label}$ queries by Algorithm 5.5 is bounded by

$$\begin{aligned}
&2+\sum_{i=1}^{I} O\left(2^i\cdot\theta_{k_i}(2\nu+2\epsilon)\cdot(\nu+\sigma_{k^*}(2^i,\delta_{i,k^*}))\right)\\
&= 2+\sum_{i=1}^{I} O\left(2^i\cdot\max_{k\le k^*}\theta_k(2\nu+2\epsilon)\cdot(\nu+\sigma_{k^*}(2^i,\delta_{i,k^*}))\right)\\
&= O\left(\max_{k\le k^*}\theta_k(2\nu+2\epsilon)\cdot\left(\sum_{i=1}^{I}2^i(\nu+\sigma_{k^*}(2^i,\delta_{i,k^*}))\right)\right)\\
&= O\left(\max_{k\le k^*}\theta_k(2\nu+2\epsilon)\cdot\left(\nu 2^I+\sum_{i=1}^{I}2^i\frac{d\ln(2^i)+\ln(\frac{(i^2+i)(k^*)^2}{\delta})}{2^i}\right)\right)\\
&= O\left(\max_{k\le k^*}\theta_k(2\nu+2\epsilon)\cdot\left(\nu 2^I+d_{k^*}I^2+I\log\frac{k^*}{\delta}\right)\right)\\
&= O\left(\max_{k\le k^*}\theta_k(2\nu+2\epsilon)\cdot\left(\frac{\nu^2+\epsilon\nu}{\epsilon^2}\left(d_{k^*}\log\frac{1}{\epsilon}+\log\frac{k^*}{\delta}\right)+d_{k^*}\left(\log\frac{d_{k^*}}{\epsilon}+\log\log\frac{k^*}{\delta}\right)^2\right.\right.\\
&\qquad\qquad\left.\left.+\left(\log\frac{d_{k^*}}{\epsilon}+\log\log\frac{k^*}{\delta}\right)\log\frac{k^*}{\delta}\right)\right)\\
&= \tilde{O}\left(\max_{k\le k^*}\theta_k(2\nu+2\epsilon)\cdot\left(d_{k^*}(\log\frac{1}{\epsilon})^2+\log\frac{k^*}{\delta}\right)\cdot\left(1+\frac{\nu^2}{\epsilon^2}\right)\right).
\end{aligned}$$

□

5.10 Performance Guarantees of AA-Larch

5.10.1 Detailed Description of Subroutines

Subroutine Sample-and-Label performs standard disagreement-based selective sampling. Specifically, it draws an unlabeled example x from the $D_{\mathcal{X}}$. If x is in the agreement region of candidate set V, its label is inferred as $V(x)$; otherwise, we query the Label oracle to get its label. The counter c is incremented when Label is called.

Algorithm 5.6 Sample-and-Label

Input: Candidate set $V \subset H$, oracle Label, labeled dataset L, counter c.

Output: New labeled dataset L', new counter c'.

$x \leftarrow$ independent draw from $D_{\mathcal{X}}$ (the corresponding label is hidden).
if $x \in \mathrm{Dis}(V)$ **then**
  $L' \leftarrow L \cup \{(x, \textsc{Label}(x))\}$
  $c' \leftarrow c+1$
else
  $L' \leftarrow L \cup \{(x, V(x))\}$
  $c' \leftarrow c$
end if

Subroutine Error-Check checks if the candidate set has high error, based on item 2 of Lemma 5.5 – that is, if $k = k^*$, then Error-Check should never fail. Furthermore, if candidate set V_i fails Error-Check, then V_i should have small radius – see Lemma 5.4 for details.

Algorithm 5.7 Error-Check

Input: Candidate set $V \subset H_k$, labeled dataset L of size l, confidence δ.

Output: Boolean variable b indicating if V has high error.

Let $\delta_k := \delta/((k+1)(k+2))$ for all $k \geq 0$.
$\gamma \leftarrow \min_{k' \geq k, h \in H_{k'}} \left\{ \mathrm{err}(h, L) + 2\sqrt{\mathrm{err}(h, L)\sigma_{k'}(l, \delta_{k'})} + 3\sigma_{k'}(l, \delta_{k'}) \right\}$
if $\min_{h \in V} \mathrm{err}(h, L) > \gamma + 2\sqrt{\gamma\sigma_k(l, \delta_k)} + 3\sigma_k(l, \delta_k)$ **then**
  $b \leftarrow$ **true**
else
  $b \leftarrow$ **false**
end if

Subroutine PRUNE-CANDIDATE-SET performs update on our candidate set based on standard generalization error bounds. The candidate set never eliminates the optimal hypothesis in $H_k(S)$ when working with H_k. Claim 5.4 shows that, if at step i, $k = k^*$, then $h^* \in V_i$ from then on.

Algorithm 5.8 PRUNE-CANDIDATE-SET

Input: Candidate set $V \subset H_k$, labeled dataset L of size l, confidence δ.

Output: Pruned candidate set V'.

1: Update candidate set:

$$V' \leftarrow \left\{ h \in V : \mathrm{err}(h, L) \leq \min_{h' \in V} \mathrm{err}(h', L) + 2\sqrt{\mathrm{err}(h', L)\sigma_k(l, \delta_k)} + 3\sigma_k(l, \delta_k) \right\},$$

where $\delta_k := \frac{\delta}{(k+1)(k+2)}$.

Subroutine UPGRADE-CANDIDATE-SET is called when (1) a systematic mistake of the candidate set V_i has been found by SEARCH; or (2) ERROR-CHECK detects that the error of V_i is high. In either case, k can be increased to the minimum level such that the updated $H_k(S)$ is nonempty. This still maintains the invariant that $k \leq k^*$.

Algorithm 5.9 UPGRADE-CANDIDATE-SET

Input: Current level of hypothesis class k, seed set S, seed to be added s.

Output: New level of hypothesis class k, new seed set S, updated candidate set V.

1: $S \leftarrow S \cup s$

2: $k \leftarrow \min \left\{ k' > k : H_{k'}(S) \neq \emptyset \right\}$

3: $V \leftarrow H_k(S)$

5.10.2 Proof of Theorem 5.4

We restate Theorem 5.4 here for convenience.

Theorem 5.6. *There exist constants $c_1, c_2 > 0$ such that the following holds. Assume $\mathrm{err}(h^*) = \nu$. Let $\theta_{k'}(\cdot)$ denote the disagreement coefficient of V_i at the first step i after which $k \geq k'$. Fix any $\epsilon, \delta \in (0,1)$. Let $n_\epsilon = c_1 \max_{k \leq k^*} \theta_k(2\nu + 2\epsilon)(d_{k^*} \log \frac{1}{\epsilon} + \log \frac{1}{\delta})(1 + \nu^2/\epsilon^2)$ and define $C_\epsilon = 2(n_\epsilon + k^*\tau)$. Run Algorithm 5.2 with a nested sequence of hypotheses $\{H_k\}_{k=0}^{\infty}$, oracles LABEL and SEARCH, confidence parameter δ, cost ratio $\tau \geq 1$, and upper bound $N = c_2(d_{k^*} \log \frac{1}{\epsilon} + \log \frac{1}{\delta})/\epsilon^2$.*

If the cost spent is at least C_ϵ, then with probability $1-\delta$, the current hypothesis $\tilde{h}$ has error at most $\nu+\epsilon$.

Remark. The purpose of having a bound on unlabeled examples, N, is rather technical—to deter the algorithm from getting into an infinite loop due to its blind self-confidence. Suppose that AA-Larch starts with H_0 that has a single element h. Then, without such an N-based condition, it will incorrectly infer the labels of all the unlabeled examples drawn and end up with an infinite loop between lines 4 and 14. The condition on N is very mild—any N satisfying $N = \text{poly}(d_{k^*}, 1/\epsilon)$ and $N = \Omega(d_{k^*}/\epsilon^2)$ is sufficient.

Proof of Theorem 5.6. For integer $j \geq 0$, define step j as the execution period in AA-Larch when the value of i is j.

Let $l_i = |L_i|$. Denote by L_i^D the dataset containing unlabeled examples in L_i labeled entirely by Label, i.e., $L_i^D = \big\{(x, \textsc{Label}(x)) : (x,y) \in L_i\big\}$. Note that L_i^D is an iid sample from D.

We call dataset L_i has *favorable bias*, if the following holds for any hypothesis h:

$$\text{err}(h, L_i^D) - \text{err}(h^*, L_i^D) \leq \text{err}(h, L_i) - \text{err}(h^*, L_i). \tag{5.8}$$

Let E_i be the event that the following conditions hold:

1. For every $k \geq 0$, every $h \in H_k$ satisfies

$$\text{err}(h,D) \;\leq\; \text{err}(h, L_i^D) + \sqrt{\text{err}(h, L_i^D)\sigma_k(l_i, \delta_{i,k})} + \sigma_k(l_i, \delta_{i,k}),$$

$$\text{err}(h, L_i^D) \;\leq\; \text{err}(h,D) + \sqrt{\text{err}(h,D)\sigma_k(l_i, \delta_{i,k})} + \sigma_k(l_i, \delta_{i,k}).$$

   For every $h, h' \in H_k$,

$$\begin{aligned}&(\text{err}(h, L_i^D) - \text{err}(h', L_i^D)) - (\text{err}(h,D) - \text{err}(h',D)) \\ &\qquad \leq \sqrt{d_{L_i^D}(h,h') \cdot \sigma_k(l_i, \delta_{i,k})} + \sigma_k(l_i, \delta_{i,k}).\end{aligned}$$

   where $d_{L_i^D}(h,h') = \frac{1}{l_i}\sum_{(x,y)\in L_i^D}[h(x) \neq h'(x)]$, fraction of L_i^D where h and h' disagree.

2. For every $1 \le i' < i$, the number of LABEL queries from step i' to step i is at most

$$\sum_{j=i'}^{i} \Pr_{x \sim D_{\mathcal{X}}}[x \in \mathrm{Dis}(V_{j-1})] + O\left(\sqrt{\sum_{j=i'}^{i} \Pr_{x \sim D_{\mathcal{X}}}[x \in \mathrm{Dis}(V_{j-1})] \log(1/\delta_i)} + \log(1/\delta_i)\right),$$

where V_j denotes its final value in Algorithm 5.2.

Using Theorem A.1 and Lemma A.4, along with the union bound, $\Pr(E_i) \ge 1 - \delta_i$. Define $E := \cap_{i=1}^{\infty} E_i$, by union bound, $\Pr(E) \ge 1 - \delta$. We henceforth condition on E holding.

Define

$$\begin{aligned} M(\nu, k^*, \epsilon, \delta, N) &:= \min\left\{ m \in \mathbb{N} : 8\sqrt{\nu \sigma_{k^*}(m, \delta_{m+k^*N,k^*})} + 35\sigma_{k^*}(m, \delta_{m+k^*N,k^*}) \le \epsilon \right\} \\ &\le O\left(\frac{(d_{k^*} \log(1/\epsilon) + \log(Nk^*/\delta))(\nu + \epsilon)}{\epsilon^2} \right) \end{aligned}$$

We say that an iteration of the loop is *verified* if Step 20 is triggered; all other iterations are *unverified*. Let Γ be the set of i's where x_i gets added to the final set L, and Δ be the set of i's where x_i gets discarded. It is easy to see that if i is in Γ (resp. Δ), then the i is in a verified (resp. unverified) iteration.

Define $i^* := \min\left\{ i \in \Gamma : l_i \ge M(\nu, k^*, \epsilon, \delta, N) \right\}$. Denote by k_i the final value of k after i unlabeled examples are processed.

We need to prove two claims:

1. For $i \ge i^*$, $\mathrm{err}(\tilde{h}_i) \le \nu + \epsilon$, where $\tilde{h}_i$ is the hypothesis $\tilde{h}$ stored at the end of step i.

2. The total cost spent by Algorithm 5.2 up to step i^* is at most C_ϵ.

To prove the first claim, fix any $i \ge i^*$. The stored hypothesis $\tilde{h}_i$ is updated only when $i \in \Gamma$, so it suffices to consider only $i \in \Gamma$. From Lemma 5.6, $i \le l_i + k^*N$. We also have $l_i \ge M(\nu, k^*, \epsilon, \delta, N)$. Since $\tilde{h}_i \in V_i$, Lemma 5.4 gives

$$\begin{aligned} \mathrm{err}(\tilde{h}_i) &\le \nu + 8\sqrt{\nu \sigma_{k^*}(l_i, \delta_{i,k^*})} + 35\sigma_{k^*}(l_i, \delta_{i,k^*}) \\ &\le \nu + 8\sqrt{\nu \sigma_{k^*}(l_i, \delta_{l_i+k^*N,k^*})} + 35\sigma_{k^*}(l_i, \delta_{l_i+k^*N,k^*}) \\ &\le \nu + \epsilon, \end{aligned}$$

as desired.

For the second claim, we first show that for i in Γ, the candidate set is contained in a ball of small radius (with respect to the disagreement pseudometric), thus bounding the size of its disagreement region. Lemma 5.4 shows that for $i \in \Gamma$, every hypothesis $h \in V_i$ has error at most $\nu + 8\sqrt{\nu\sigma_{k^*}(l_i, \delta_{i,k^*})} + 35\sigma_{k^*}(l_i, \delta_{i,k^*})$.

Thus, by the triangle inequality and Lemma 5.6,

$$\begin{aligned} V_i &\subseteq \mathrm{B}_{H_{k_i}}(h, 2\nu + 16\sqrt{\nu\sigma_{k^*}(l_i, \delta_{i,k^*})} + 70\sigma_{k^*}(l_i, \delta_{i,k^*})) \\ &\subseteq \mathrm{B}_{H_{k_i}}(h, 2\nu + 16\sqrt{\nu\sigma_{k^*}(l_i, \delta_{l_i+k^*N,k^*})} + 70\sigma_{k^*}(l_i, \delta_{l_i+k^*N,k^*})). \end{aligned}$$

for some h in $H_{k_i}(S)$. This shows that for $i \in \Gamma$, $i \le i^*$,

$$\begin{aligned} & \Pr_{x\sim D_{\mathcal{X}}}[x \in \mathrm{Dis}(V_i)] \\ \le\ & \theta_{k_i}(2\nu + 2\epsilon) \cdot \left(2\nu + 16\sqrt{\nu\sigma_{k^*}(l_i, \delta_{l_i+k^*N,k^*})} + 70\sigma_{k^*}(l_i, \delta_{l_i+k^*N,k^*})\right) \\ \le\ & \max_{k\le k^*} \theta_k(2\nu + 2\epsilon) \cdot \left(2\nu + 16\sqrt{\nu\sigma_{k^*}(l_i, \delta_{l_i+k^*N,k^*})} + 70\sigma_{k^*}(l_i, \delta_{l_i+k^*N,k^*})\right), \end{aligned} \tag{5.9}$$

where the first inequality is from the definition of $\theta_{k_i}(\cdot)$ and the fact that $8\sqrt{\nu\sigma_{k^*}(l_i, \delta_{l_i+k^*N,k^*})} + 35\sigma_{k^*}(l_i, \delta_{l_i+k^*N,k^*}) \ge \epsilon$ for $i \le i^*$, the second inequality is from $k_i \le k^*$.

For $i \ge 1$, let Z_i be the indicator of whether LABEL is queried with x_i in Step 12, i.e.,

$$Z_i = \mathbb{1}\{x_i \in \mathrm{Dis}(V_{i-1})\}.$$

For $0 \le k \le k_{i^*}$, define

$$\begin{aligned} i_k^0 &:= \min\left\{i \le i^* : k_i \ge k\right\}, \text{ the first step when the hypothesis class reaches } \ge k, \\ i_k &:= \max\left\{i \le i^* : k_i \le k\right\}, \text{ the last step by the end of which the hypothesis class is still } \le k, \\ i'_k &:= \max\left\{i_k^0 \le i \le i_k : k_i \le k, i \in \Gamma\right\}, \text{ the last verified step for hypothesis class } \le k \text{ (if exists).} \end{aligned}$$

We call class k *skipped* if there is no step i such that $k_i = k$. If level k is skipped, then $i_k = i_{k-1} = i_k^0 - 1$, and i'_k is undefined.

Let

$$W_k := \sum_{i=i_k^0+1}^{i'_k} Z_i$$

be the number of verified queried examples when working with hypothesis class H_k. Note that W_k/τ is the number of verified iterations when working with H_k. If level k is skipped, then $W_k := 0$.

Let

$$Y_k := \sum_{i=i'_k+1}^{i_k+1} Z_i$$

be the number of unverified queried examples when working with hypothesis class H_k. Note that $Y_k \leq \tau$, and there is at most one unverified iteration when working with H_k. If level k is skipped,then $Y_k := 0$.

Therefore, the total cost when working with H_k is at most

$$\frac{W_k}{\tau} \cdot 2\tau + Y_k + \tau \leq 2\tau + 2W_k$$

Furthermore, Claim 5.4 implies that there is no unverified iteration when working with H_{k^*}. Hence the total cost when working with H_{k^*} has a tighter upper bound, that is, $2W_{k^*}$.

As a shorthand, let $m = M(\nu, k^*, \epsilon, \delta, N)$. We now bound the total cost incurred up to

time i^* as

$$\begin{aligned}
\sum_{k=0}^{k^*-1}(2\tau+2W_k)+2W_{k^*} &= 2\tau k^*+2\sum_{k=0}^{k_{i^*}}W_k \\
&= 2\tau k^*+2\sum_{k=0}^{k_{i^*}}\sum_{i=i_k^0+1}^{i_k'}Z_i \\
&= 2\tau k^*+O\left(2\sum_{i\in\Gamma:i\le i^*}\Pr_{x\sim D_{\mathcal{X}}}[x\in \mathrm{Dis}(V_{i-1})]\right)+O\left(k^*\ln\frac{1}{\delta_{i^*}}\right) \\
&\le 2\left(\tau k^*+O\left(\sum_{l=1}^{m-1}\max_{k\le k^*}\theta_k(2\nu+2\epsilon)(\nu+\sigma_{k^*}(l,\delta_{l+k^*N,k^*}))\right)\right) \\
&\le 2\left(\tau k^*+O\left(\max_{k\le k^*}\theta_k(2\nu+2\epsilon)\sum_{l=1}^{m-1}(\nu+\sigma_{k^*}(l,\delta_{l+k^*N,k^*}))\right)\right) \\
&\le 2\left(\tau k^*+\tilde{O}\left(\max_{k\le k^*}\theta_k(2\nu+2\epsilon)d_{k^*}\left(1+\frac{\nu^2}{\epsilon^2}\right)\right)\right) \\
&\le 2\left(\tau k^*+n_\epsilon\right)=C_\epsilon,
\end{aligned}$$

where the first equality is by algebra, the second equality is from the definition of W_k, and the third equality is from the definition of E. The first inequality is from Lemma 5.4, using Equation (5.9) to bound $\Pr_{x\sim D_{\mathcal{X}}}[x\in \mathrm{Dis}(V_{i-1})]$ and noting that $\{l_i : i\in\Gamma, i\le i^*\}=[m]$. □

Now we provide the proof of our two key lemmas(Lemmas 5.4 and 5.6).

Consider the last call of PRUNE-CANDIDATE-SET in step i. Define γ_i as the value of γ in line 2 of ERROR-CHECK:

$$\gamma_i=\min_{k'\ge k_i,h\in H_{k'}}\left\{\mathrm{err}(h,L_i)+2\sqrt{\mathrm{err}(h,L_i)\sigma_{k'}(l,\delta_{i,k'})}+3\sigma_{k'}(l,\delta_{i,k'})\right\} \tag{5.10}$$

Meanwhile, from line 1 of PRUNE-CANDIDATE-SET, we have for all $h\in V_i$,

$$\mathrm{err}(h,L_i)\le\min_{h'\in V_i}\mathrm{err}(h',L_i)+2\sqrt{\mathrm{err}(h',L_i)\sigma_{k_i}(l_i,\delta_{i,k_i})}+3\sigma_{k_i}(l_i,\delta_{i,k_i})$$

where V_i denotes its final value.

Lemma 5.4. *Assume that the following conditions hold:*

1. *The dataset L_i has favorable bias, i.e. it satisfies Equation* (5.8).

2. *The candidate set V_i is such that* ERROR-CHECK(V_i, L_i, δ_i) *returns false, i.e. it has a low empirical error on L_i:*

$$\min_{h' \in V_i} \mathrm{err}(h', L_i) \leq \gamma_i + 2\sqrt{\gamma_i \sigma_{k_i}(l_i, \delta_{i,k_i})} + 3\sigma_{k_i}(l_i, \delta_{i,k_i}). \tag{5.11}$$

Then, every $h \in V_i$ is such that

$$\mathrm{err}(h) \leq \nu + 8\sqrt{\nu \sigma_{k^*}(l_i, \delta_{i,k^*})} + 35\sigma_{k^*}(l_i, \delta_{i,k^*}). \tag{5.12}$$

where V_i and L_i denote their final values, respectively. Specifically, Equation (5.12) *holds for any $h \in V_i$ such that $i \in \Gamma$ or $i+1 \in \Gamma$.*

Proof. Lemma 5.6 shows that $k_i \leq k^*$, which we will use below.

Start with Equation (5.11):

$$\min_{h' \in V_i} \mathrm{err}(h', L_i) \leq \gamma_i + 2\sqrt{\gamma_i \sigma_{k_i}(l_i, \delta_{i,k_i})} + 3\sigma_{k_i}(l_i, \delta_{i,k_i}).$$

Since $k_i \leq k^*$, $\sigma_{k_i}(l_i, \delta_{i,k_i}) \leq \sigma_{k^*}(l_i, \delta_{i,k^*})$.

From the definition of γ_i (Equation (5.10)), taking $k = k^* \geq k_i$, $h = h^* \in H_{k^*}$,

$$\gamma_i \leq \mathrm{err}(h^*, L_i) + 2\sqrt{\mathrm{err}(h^*, L_i)\sigma_{k^*}(l_i, \delta_{i,k^*})} + 3\sigma_{k^*}(l_i, \delta_{i,k^*}).$$

Plugging the latter into the former and using σ as a shorthand for $\sigma_{k^*}(l_i, \delta_{i,k^*})$, we have

$$\begin{aligned}
\min_{h' \in V_i} \mathrm{err}(h', L_i) \leq & \mathrm{err}(h^*, L_i) + 2\sqrt{\mathrm{err}(h^*, L_i)\sigma} + 3\sigma + 2\sqrt{(\mathrm{err}(h^*, L_i) + 2\sqrt{\mathrm{err}(h^*, L_i)\sigma} + 3\sigma)\sigma} \\
& + 3\sigma \\
\leq & \mathrm{err}(h^*, L_i) + 2\sqrt{\mathrm{err}(h^*, L_i)\sigma} + 6\sigma + 2(\sqrt{\mathrm{err}(h^*, L_i)\sigma} + \sqrt{3}\sigma) \\
\leq & \mathrm{err}(h^*, L_i) + 4\sqrt{\mathrm{err}(h^*, L_i)\sigma} + 10\sigma\,.
\end{aligned}$$

Fix any $h \in V_i$. By construction,

$$\mathrm{err}(h, L_i) \leq \min_{h' \in V_i} \mathrm{err}(h', L_i) + 2\sqrt{\mathrm{err}(h', L_i)\sigma_{k_i}(l_i, \delta_{i,k_i})} + 3\sigma_{k_i}(l_i, \delta_{i,k_i}).$$

Plugging the former into the latter (recalling that $\sigma_{k_i}(l_i, \delta_{i,k_i}) \leq \sigma$) gives

$$\begin{aligned}\operatorname{err}(h, L_i) - \operatorname{err}(h^*, L_i) \leq & 4\sqrt{\operatorname{err}(h^*, L_i)\sigma} + 10\sigma + 2(\sqrt{\operatorname{err}(h^*, L_i)\sigma} + \sqrt{10}\sigma) + 3\sigma \\ \leq & 6\sqrt{\operatorname{err}(h^*, L_i)\sigma} + 20\sigma.\end{aligned}$$

Combined with Equation (5.8), we have,

$$\operatorname{err}(h, L_i^D) - \operatorname{err}(h^*, L_i^D) \leq 6\sqrt{\operatorname{err}(h^*, L_i)\sigma} + 20\sigma.$$

Since $\operatorname{err}(h^*, L_i) \leq \operatorname{err}(h^*, L_i^D)$,

$$\operatorname{err}(h, L_i^D) - \operatorname{err}(h^*, L_i^D) \leq 6\sqrt{\operatorname{err}(h^*, L_i^D)\sigma} + 20\sigma.$$

From the definition of E_i,

$$\operatorname{err}(h^*, L_i^D) \leq \nu + \sqrt{\nu\sigma} + \sigma.$$

$$\operatorname{err}(h) \leq \operatorname{err}(h, L_i^D) + \sqrt{\operatorname{err}(h, L_i^D)\sigma} + \sigma.$$

Plugging in and simplifying algebraically gives

$$\operatorname{err}(h) \leq \nu + 8\sqrt{\nu\sigma} + 35\sigma.$$

Now, if $i \in \Gamma$, the dataset L_i has favorable bias from lemma 5.7; if $i \notin \Gamma$ and $i+1 \in \Gamma$, the final value of L_i equals some L_j for some $j \in \Gamma$, therefore also has favorable bias.

Meanwhile, if $i \in \Gamma$, Algorithm 5.2 fails ERROR-CHECK(V_i, L_i, δ_i) for $k = k_i$. If $i \notin \Gamma$ and $i+1 \in \Gamma$, then $i+1$ is the start of some verified iteration, i.e. $i+1 = i_k^0$ for some k. Hence the final value of V_i also fails ERROR-CHECK(V_i, L_i, δ_i) for $k = k_i$. In both cases, Equation (5.11) holds.

Therefore, if $i \in \Gamma$ or $i+1 \in \Gamma$, then Equation (5.12) holds for every h in V_i. □

Lemma 5.5. *For step i, suppose L_i has favorable bias, i.e. Equation* (5.8) *holds. Then for any k and any $h \in H_k$,*

$$\operatorname{err}(h^*, L_i) - \operatorname{err}(h, L_i) \leq 2\sqrt{\operatorname{err}(h, L_i)\sigma_{\bar{k}}(l_i, \delta_{i,\bar{k}})} + 3\sigma_{\bar{k}}(l_i, \delta_{i,\bar{k}}),$$

where $\bar{k} = \max(k^*, k)$. *Specifically:*

1. *for any* $h \in H_{k^*}$,

$$\mathrm{err}(h^*, L_i) - \mathrm{err}(h, L_i) \leq 2\sqrt{\mathrm{err}(h, L_i)\sigma_{k^*}(l_i, \delta_{i,k^*})} + 3\sigma_{k^*}(l_i, \delta_{i,k^*}), \tag{5.13}$$

2. *The empirical error of* h^* *on* L_i *can be bounded as follows:*

$$\mathrm{err}(h^*, L_i) \leq \gamma_i + 2\sqrt{\gamma_i \sigma_{k^*}(l_i, \delta_{i,k^*})} + 3\sigma_{k^*}(l_i, \delta_{i,k^*}) \tag{5.14}$$

Proof. Fix any k and $h \in H_k$. Since $\bar{k} \geq k$, $\sigma_k(l_i, \delta_{i,k}) \leq \sigma_{\bar{k}}(l_i, \delta_{i,\bar{k}})$. Similarly, $\sigma_{k^*}(l_i, \delta_{i,k^*}) \leq \sigma_{\bar{k}}(l_i, \delta_{i,\bar{k}})$. Using the shorthand $\sigma := \sigma_{\bar{k}}(l_i, \delta_{i,\bar{k}})$ and noting that $h, h^* \in H_{\bar{k}}$,

$$\begin{aligned}
\mathrm{err}(h^*, L_i) - \mathrm{err}(h, L_i) &\leq \mathrm{err}(h^*, L_i^D) - \mathrm{err}(h, L_i^D) \\
&\leq \sqrt{d_{L_i^D}(h^*, h) \cdot \sigma} + \sigma \\
&\leq \sqrt{(\mathrm{err}(h^*, L_i) + \mathrm{err}(h, L_i)) \cdot \sigma} + \sigma. \\
&\leq \sqrt{\mathrm{err}(h^*, L_i)\sigma} + \sqrt{\mathrm{err}(h, L_i)\sigma} + \sigma.
\end{aligned}$$

where the first inequality is from Equation (5.8), the second inequality is from the definition of E_i and the optimality of h^*, and the third inequality is from the triangle inequality. Letting $A = \mathrm{err}(h^*, L_i)$, $B = \mathrm{err}(h, L_i)$, and $C = B + \sqrt{B\sigma} + \sigma$, we can rewrite the above inequality as $A \leq C + \sqrt{A\sigma}$. Solving the resulting quadratic equation in terms of A, we have $A \leq C + \sigma + \sqrt{C\sigma}$, or

$$\begin{aligned}
A &\leq B + \sqrt{B\sigma} + 2\sigma + \sqrt{\sigma(B + \sqrt{B\sigma} + \sigma)} \\
&\leq B + \sqrt{B\sigma} + 2\sigma + \sqrt{\sigma}(\sqrt{B} + \sqrt{\sigma}) \\
&\leq B + 2\sqrt{B\sigma} + 3\sigma,
\end{aligned}$$

or

$$\mathrm{err}(h^*, L_i) \leq \mathrm{err}(h, L_i) + 2\sqrt{\mathrm{err}(h, L_i)\sigma} + 3\sigma.$$

Specifically:

1. Taking $k = k^*$, we get that Equation (5.13) holds for any $h \in H_{k^*}$, establishing item 1.

2. Define

$$(\hat{k}_i, \hat{h}_i) := \arg \min_{k' \geq k^*, h \in H_{k'}} \left\{ \mathrm{err}(h, L_i) + 2\sqrt{\mathrm{err}(h, L_i)\sigma_{k'}(l_i, \delta_{i,k'})} + 3\sigma_{k'}(l_i, \delta_{i,k'}) \right\}.$$

In this notation, $\gamma_i = \mathrm{err}(\hat{h}_i, L_i) + 2\sqrt{\mathrm{err}(\hat{h}_i, L_i)\sigma_{\hat{k}_i}(l_i, \delta_{i,\hat{k}_i})} + 3\sigma_{\hat{k}_i}(l_i, \delta_{i,\hat{k}_i})$. We have

$$\begin{aligned}
&\gamma_i + 2\sqrt{\gamma_i \sigma_{k^*}(l_i, \delta_{i,k^*})} + 3\sigma_{k^*}(l_i, \delta_{i,k^*}) \\
\geq &\mathrm{err}(\hat{h}_i, L_i) + 2\sqrt{\mathrm{err}(\hat{h}_i, L_i)\sigma_{\bar{k}}(l_i, \delta_{i,\bar{k}})} + 3\sigma_{\bar{k}}(l_i, \delta_{i,\bar{k}}) \\
\geq &\mathrm{err}(h^*, L_i),
\end{aligned}$$

where $\bar{k} = \max(k^*, \hat{k}_i)$ and the last inequality comes from applying Lemma 5.5 for $h' = \hat{h}_i \in H_{\hat{k}_i}$ and $\bar{k}$. This establishes Equation (5.14), proving item 2. □

Lemma 5.6. *At any step of AA-LARCH, $k \leq k^*$. Consequently, for every i, $i \leq l_i + k^* N$.*

Proof. We prove the lemma in two steps.

1. Notice that there are two places where k is incremented in AA-LARCH, line 6 and line 17. If $k < k^*$, neither line would increment it beyond k^* as $h^* \in H_{k^*}$ and h^* is consistent with S. If $k = k^*$, Claim 5.4 below shows that k will stay at k^*. This proves the first part of the claim.

2. An iteration becomes unverified only if k gets incremented, and Algorithm 5.2 maintains the invariant that $k_i \leq k^*$. Thus, the number of unverified iterations is at most k^*. In addition, each newly sampled set is of size at most N. So the number of unverified examples is at most $k^* N$.

   Hence, i—the total number of examples processed up to step i—equals the sum of the number of verified examples l_i, plus the number of unverified examples, which is at most $k^* N$. This proves the second part of the claim. □

We show a technical claim used in the proof of Lemma 5.6 which guarantees that, on event E, when k has reached k^*, it will remain k^* from then on. Recall that k_i is defined as the final value of k at the end of step i; $i_k^0 = \min\{i : k_i \geq k\}$ is the step at the end of which the working hypothesis space reaches level $\geq k$.

Claim 5.4. *If $i^0_{k^*}$ is finite, then the following hold for all $i \geq i^0_{k^*}$:*

(C1) *L_i has favorable bias.*

(C2) *Step i terminates with $k_i = k^*$.*

(C3) *$h^* \in V_i$.*

Above, L_i and V_i denote their final values in AA-LARCH.

Proof. By induction on i. **Base Case.** Let $i = i^0_{k^*}$. Consider the execution of AA-LARCH at the start of step $i^0_{k^*}$ (line 11). Since by definition of i_k, the final value of k at step $i^0_{k^*} - 1$ is $< k^*$, at step $i^0_{k^*}$, line 5 or line 16 is triggered. Hence the dataset $L_{i^0_{k^*}}$ equals some verified labeled dataset L stored by AA-LARCH, i.e. L_j for some $j \in \Gamma$. Thus, applying Lemma 5.7, Claim C1 holds.

We focus on the moment in step $i = i^0_{k^*}$ when k increases to k^* in subprocedure UPGRADE-CANDIDATE-SET(line 6 or 17). Now consider the temporary $V_{i^0_{k^*}}$ computed in the next line (PRUNE-CANDIDATE-SET). Item 2 of Lemma 5.5 implies that the candidate set $V_{i^0_{k^*}}$ is such that ERROR-CHECK($V_{i^0_{k^*}}, L_{i^0_{k^*}}, \delta_{i^0_{k^*}}$) returns false. Therefore the final value of k in step $i^0_{k^*}$ is exactly k^*. Claim C2 follows.

Claim C2 implies the temporary $V_{i^0_{k^*}}$ is final. Item 1 of Lemma 5.5 implies that $h^* \in V_{i^0_{k^*}}$, establishing Claim C3.

Inductive Case. Now consider $i \geq i^0_{k^*} + 1$. The inductive hypothesis says that Claims C1–3 hold for step $i-1$.

Claim C1 follows from Claim C3 in step $i-1$. Indeed, the newly added x_i either comes from the agreement region of V_{i-1}, in which case label y_i agrees with $h^*(x_i)$, or is from the disagreement region of V_{i-1}, in which case the inferred label y_i is queried from LABEL. Following the same reasoning as the proof of Lemma 5.7, Claim C1 is true.

Claims C2 and C3 follows the same reasoning as the proof for the base case. □

Lemma 5.7. *If i is in Γ, then L_i has favorable bias. That is, for any hypothesis h,*

$$\mathrm{err}(h, L_i^D) - \mathrm{err}(h^*, L_i^D) \leq \mathrm{err}(h, L_i) - \mathrm{err}(h^*, L_i).$$

Proof. We can split L_i^D into two subsets, the subset where L_i^D agrees with L_i and the subset $Q_i^D = \{(x,y) \in L_i^D : h^*(x) \neq y\}$ where L_i^D disagrees with L_i. On the former subset, L_i^D is identical to L_i, thus we just need to show that

$$\mathrm{err}(h, Q_i^D) - \mathrm{err}(h^*, Q_i^D) \leq \mathrm{err}(h, Q_i) - \mathrm{err}(h^*, Q_i),$$

where $Q_i = \{(x,y) : (x,-y) \in Q_i^D\}$. Since $\mathrm{err}(h^*, Q_i^D) = 1$ and $\mathrm{err}(h^*, Q_i) = 0$, this reduces to showing that $\mathrm{err}(h, Q_i^D) \leq 1 + \mathrm{err}(h, Q_i)$, which is easily seen to hold for any h as $\mathrm{err}(h, Q_i^D) \leq 1$ and $\mathrm{err}(h, Q_i) \geq 0$. □

5.11 Acknowledgements

This chapter is based on the material as it appears in Advances in Neural Information Processing Systems 2016 (Alina Beygelzimer, Daniel Hsu, John Langford and Chicheng Zhang, "Search Improves Label for Active Learning"). The dissertation author is the primary investigator and author of this material.

Chapter 6

Confidence-rated Prediction in the Batch Setting

6.1 Introduction

Confidence-rated prediction (also known as "classification with a reject option") refers to the learning setting where the learned classifier (also known as confidence-rated predictor), is allowed to abstain, that is, to output "I don't know". In this chapter, we study confidence-rated prediction in the batch setting, that is, we are given training and test examples in batches, and would like to output predictions on the test examples. As discussed in Section 1.2, the performance of a confidence-rated predictor is measure by two quantities: error (or mistake), the fraction of examples on which it predicts a wrong label, and abstention, the fraction of examples on which it abstains. Usually there is tradeoff between these two performance measures - if we are allowed a higher abstention rate, then error rate can be reduced.

To establish a tradeoff between the two measures, one plausible formulation is to design confidence-rated predictors that provide a guaranteed upper bound on the error. [1] In this chapter, we focus on this formulation. A natural strategy for confidence-rated prediction in this setting is as follows. Maintain a candidate set of hypotheses based on the input training examples, and

[1] Alternatively, we can design confidence-rated predictors that have a guaranteed upper bound on the abstention rate. These two formulations can sometimes be equivalent if concrete objectives are defined, using the techniques in Section 4.7.5 of [BV04].

abstain from prediction whenever there are two hypotheses in this candidate set that *disagree* on the label of a test example. This *disagreement-based* strategy was proposed by [EYW10], and was shown to have zero error in the realizable case. If a small error η is allowed, then, instead of abstaining, they suggest predicting the label assigned by a reference classifier in the candidate set with some small probability, and abstaining otherwise. However, the optimality of this algorithm is questionable, and it leaves open the question that whether better algorithms exist in this setting.

In this chapter, we address this question by providing a novel confidence-rated prediction algorithm with error guarantees. Our algorithm is fully general, in the sense that it applies to any hypothesis class and any data distribution. We show that our algorithm is *optimal* in the realizable case, in the sense that any other algorithm that guarantees error η given the input labeled examples will necessarily have the same or higher abstention rate. We show how to convert this algorithm to work in the agnostic setting such that its error with respect to the best hypothesis in the class is at most η; in this case, our algorithm still has better performance than the disagreement-based strategy adapted to this setting. Our algorithm applies in the transductive setting, and given a set of labeled and unlabeled examples drawn from a data distribution, finds a confidence-rated predictor with guaranteed error η on the unlabeled examples.

While our algorithm is computationally inefficient in general, we next show how to implement an approximate version of it and one of its variants efficiently through bootstrap sampling from the candidate sets. These approximate versions have error guarantees with respect to the bootstrapped subsample of hypotheses.

Finally, we evaluate our algorithm and one of its variants through two experimental tasks on real data. Define the coverage of a confidence rated predictor as 1 minus its abstention rate, as in [EYW10]. For the first task, we measure the coverage as a function of the worst-case error guarantee, and show that our algorithms provide significantly better coverage for the same error guarantee when compared with previous work. For our second experimental task, we measure the risk (or error to coverage ratio) where the error is evaluated with respect to the actual test labels. We show that a variant of our algorithm significantly outperforms [EYW10] in this case, and achieves error-abstention tradeoffs competitive with the algorithms of [EYW11] and [GWBV02, Muk03] which do not possess any error guarantees.

6.2 Algorithms

We present a confidence-rated prediction strategy in Algorithm 6.1, and one of its variants, Algorithm 6.2 in the transductive setting. These algorithms are fully general in the sense that they apply to any hypothesis class and any data distribution. Both are presented for the realizable case; extension to the non-realizable case is discussed in Section 6.2.4.

6.2.1 The Transductive Setting

We begin with some definitions. In the transductive setting, we assume that we are given a set S of n labeled examples $\{(x_1,y_1),\ldots,(x_n,y_n)\}$, where each x_i lies in a domain $\mathcal{X}$ and each $y_i \in \{-1,1\}$. We are also given a set $U = \{x_{n+1},\ldots,x_{n+m}\}$ of m unlabeled examples.

A confidence-rated predictor P is a function that maps each element of U to a distribution over $\{-1,1,\perp\}$, where $\perp$ indicates "I don't know". If for $x_{n+j} \in U$, $P(x_{n+j}) = (\alpha_j,\beta_j,\gamma_j)$, then the prediction of P on x_{n+j} is $+1$ with probability α_j, -1 with probability β_j and $\perp$ with probability γ_j.

In the sequel, we distinguish between confidence-rated predictors and selective classifiers, which are a special case of confidence-rated predictors. A selective classifier C is a tuple $(h,(\mu_1,\ldots,\mu_m))$, where h lies in a hypothesis class $\mathcal{H}$, and $0 \le \mu_i \le 1$ for all $i = 1,\ldots,m$. For any $x_{n+j} \in U$, the prediction of a selective classifier C on x_{n+j} is $h(x_{n+j})$ with probability μ_j and $\perp$ with probability $1-\mu_j$.

Finally, we calculate the error and the abstention of a confidence-rated predictor with respect to the uniform distribution over the unlabeled sample set U.

6.2.2 Algorithms

Given a training set S and an unlabeled dataset U, Algorithm 6.1 first constructs the candidate set V of S with respect to the hypothesis class $\mathcal{H}$. Our key observation is that once this candidate set has been constructed, finding the optimal abstention rate confidence-rated predictor which has guaranteed error $\le \eta$ can be expressed as a *linear program*. The linear program has three variables for each unlabeled example i, α_i, which denotes the probability with which we predict 1, β_i, the probability with which we predict -1, and γ_i, the probability we predict $\perp$ on this example. There is a constraint corresponding to each hypothesis h in the

candidate set, which ensures that if the true hypothesis is h, then the error of the predictor is at most η. Finally, the constraints $\alpha_i + \beta_i + \gamma_i = 1$, $\alpha_i, \beta_i, \gamma_i \geq 0$ for all i ensures $(\alpha_i, \beta_i, \gamma_i)$ is a valid probability distribution.

A similar observation can be used to construct a selective classifier; we present this construction in Algorithm 6.2. We maintain a candidate set V, and we select an arbitrary $h_0 \in V$ as the classifier h. The linear program has a variable μ_i for each example i which denotes the probability with which we do not abstain on this example, and a constraint for each hypothesis h' in the candidate set that ensures that if h' is the true hypothesis, then its error is $\leq \eta$. We note that abstention achieved by Algorithm 6.2 depends on the actual choice of the hypothesis h_0; however, Theorem 6.2 indicates that the choice of h_0 would not change the abstention of the selective classifier by more than η.

Algorithm 6.1 Confidence-rated Predictor: Realizable Case

1: **Inputs:** labeled data S, unlabeled data U, error bound η.
2: Compute candidate set V with respect to S.
3: Solve the linear program:

$$\min \sum_{i=1}^{m} \gamma_i$$

subject to:

$$\forall i,\ \alpha_i + \beta_i + \gamma_i = 1 \tag{6.1}$$

$$\forall h \in V,\ \sum_{i:h(x_{n+i})=+1} \beta_i + \sum_{i:h(x_{n+i})=-1} \alpha_i \leq \eta m \tag{6.2}$$

$$\forall i,\ \alpha_i, \beta_i, \gamma_i \geq 0 \tag{6.3}$$

4: Output the confidence-rated predictor: $\{(\alpha, \beta_i, \gamma_i), i = 1, \ldots, m\}$.

Algorithm 6.2 Selective Classifier: Realizable Case

1: **Inputs:** labeled data S, unlabeled data U, error bound η.
2: Compute candidate set V with respect to S. Pick an arbitrary $h_0 \in V$.
3: Solve the linear program:

$$\min \sum_{i=1}^{m} (1 - \mu_i)$$

subject to:

$$\forall i,\ 0 \leq \mu_i \leq 1 \tag{6.4}$$

$$\forall h \in V,\ \sum_{i:h(x_{n+i}) \neq h_0(x_{n+i})} \mu_i \leq \eta m \tag{6.5}$$

4: Output the selective classifier: $(h_0, (\mu_1, \ldots, \mu_m))$.

Observe that as both algorithms involve solving a linear program over the candidate set, they are generally computationally inefficient. In Section 6.3, we show how to implement an approximate version of these algorithms for linear classification by a bootstrapping procedure that draws samples from the candidate set, and solves the linear programs using the constraints generated by these samples.

6.2.3 Guarantees for the Realizable Case

Algorithms 6.1 and 6.2 have the following performance guarantees; detailed proofs are provided in the Section 6.4.

Theorem 6.1. *Suppose we are in the realizable case, and let P be the confidence-rated predictor output by Algorithm 6.1 on inputs S, U and η. Then,* $\mathrm{err}_U(P) \leq \eta$. *Moreover, if P' is any other confidence-rated predictor that guarantees* $\mathrm{err}_U(P') \leq \eta$ *given S and U, then* $\mathrm{abs}_U(P') \geq \mathrm{abs}_U(P)$.

Theorem 6.2. *Suppose we are in the realizable case, and let C be the selective classifier output by Algorithm 6.2 on inputs S, U and η for any arbitrary choice of h_0 in the candidate set V. Then,* $\mathrm{err}_U(C) \leq \eta$. *Moreover,* $\mathrm{abs}_U(C) \leq \mathrm{abs}_U(P) + \eta$, *where P is the predictor output by Algorithm 6.1 on input S, U and η.*

Thus, Algorithms 6.1 and 6.2 offer an error guarantee $\leq \eta$ with respect to the candidate set; even if any arbitrary classifier in the candidate set is the true hypothesis, Algorithms 6.1 and 6.2 will guarantee error $\leq \eta$. Algorithm 6.1 is also optimal in the sense that any other confidence-rated predictor which offers a similar error guarantee has equal or higher abstention.

6.2.4 The Non-Realizable Case

In the non-realizable case, in general, it is impossible to provide strong error guarantees if error is measured with respect to the true labels, without further generative assumptions. For example, for true labels generated by random classification noise, guaranteeing an error better than the noise rate will require a predictor that can predict the exact noise value! Following [EYW11], we therefore consider a different notion of error, error with respect to the best hypothesis in a class $\mathcal{H}$.

Let $\mathcal{H}$ be a hypothesis class, and let h^* be the hypothesis in $\mathcal{H}$ which minimizes the true error $\Pr_{(x,y)\sim D}[h(x) \neq y]$ with respect to the data distribution D. Then, the *error of a confidence-rated predictor P with respect to the best hypothesis in $\mathcal{H}$* is the probability that P predicts 1 when $h^*(x) = -1$ and vice versa. More precisely, $\mathrm{err}_{\mathcal{H}}(P) = \Pr[P(x) = -h^*(x)]$, where the probability is taken over the uniform distribution on the unlabeled data U as well as the randomness in the prediction of P.

In the non-realizable case, there is usually no $h \in \mathcal{H}$ consistent with the training set, and the candidate set V is empty. Therefore, to ensure guaranteed error, we use instead a subset of $\mathcal{H}$ that is very likely to include the true error minimizer h^*.

Corollary 6.1 shows that if we replace V in Algorithms 6.1 and 6.2 by a $(1-\delta)$-confidence set $C(S)$, then, the resulting predictor provides an error guarantee with probability $\geq 1-\delta$.

Corollary 6.1. *Suppose we are in the non-realizable case, and let P and C respectively be the confidence-rated predictor and selective classifier obtained by replacing the candidate set V in Algorithms 6.1 and 6.2 by a $(1-\delta)$-confidence set $C(S)$. If S consists of n iid examples drawn from the data distribution D, then, with probability $\geq 1-\delta$ with respect to D, $\mathrm{err}_{\mathcal{H}}(P) \leq \eta$ and $\mathrm{err}_{\mathcal{H}}(C) \leq \eta$.*

We cannot show a guarantee of optimal abstention in the non-realizable case. While any $(1-\delta)$-confidence set $C(S)$ will give us an error guarantee with probability $\geq 1-\delta$, to get good abstention, in general, we would like $C(S)$ to be as small as possible.

If $\hat{h}$ minimizes the empirical error on the training data, then, the following set $\hat{\mathcal{V}}(\hat{h})$, used by [EYW11], is a $(1-\delta)$-level confidence set:

$$\hat{\mathcal{V}}(\hat{h}) = \left\{ h \in \mathcal{H} : \mathrm{err}(h,S) \leq \mathrm{err}(\hat{h},S) + 2\sigma(n,\delta) \right\}$$

Depending on the hypothesis class, $\hat{\mathcal{V}}(\hat{h})$ may have complex structure and may even be disconnected. To address this, previous active learning literature uses instead $B_S(\hat{h}, 2\,\mathrm{err}(\hat{h},S) + 2\sigma(n,\delta))$, the empirical disagreement ball around $\hat{h}$ of radius $2\,\mathrm{err}(\hat{h},S) + 2\sigma(n,\delta)$. Since $\hat{\mathcal{V}}(\hat{h}) \subseteq B_S(\hat{h}, 2\,\mathrm{err}(\hat{h},S) + 2\sigma(n,\delta))$, this process preserves the error guarantees, and results in a higher abstention rate.

6.3 Implementation and Experiments

We next show how to implement Algorithms 6.1 and 6.2 through bootstrapping, and evaluate their performance. We will also be using the performance measures defined in [EYW10] in this section: the coverage of a confidence-rated predictor is 1 minus its abstention rate, and its risk is the ratio between its error and its coverage. Note that the coverage is equivalent to the abstention rate modulo scaling and translation. We consider two experimental tasks. First, we compare the coverages of Algorithms 6.1 and 6.2 on real data with disagreement-based strategy for the same error guarantee $\leq \eta$. Second, we evaluate the actual risk-coverage tradeoff achieved by our algorithms on real data, by comparing it with several existing confidence-rated prediction algorithms. All experiments are based on linear separators.

Implementation by Bootstrapping The linear programs in Algorithms 6.1 and 6.2 have a constraint for each hypothesis in the candidate set; to implement them, we draw a set of samples to approximate the candidate set (or a $1-\delta$ confidence set) by a finite hypothesis set H, and use the constraints corresponding to each hypothesis in H. The resulting algorithms provide an error guarantee of η with respect to the bootstrapped sample H.

In the realizable case, the candidate set V is convex. To sample from V, we first use linear programming to find a consistent (w,b) as a starting point, and then draw samples using the Hit and Run Markov Chain [GBNT05, LV06]. In non-realizable case, we first use an SVM solution to find a $(\hat{w},\hat{b})$ as a starting point $\hat{h}$, and sample linear classifiers in the star-shaped body $B_S(\hat{h}, 2\mathrm{err}(\hat{h},S)+C\sigma(n,\delta)), C=0.2$ using a ball walk[CDV10]. [2] In each case, we run the Markov Chain until $t=100000$, and randomly select 1000 classifiers from the trajectory. Algorithm 6.2 requires a reference classifier h_0, which we select to be the SVM solution. We use the LIBSVM implementation [CL11] of SVM with $C=10^5$.

The linear programs in Algorithms 6.1 and 6.2 usually have multiple optimal solutions for a given η; we break ties among these by selecting the one which has the best *alignment* with the SVM solution. To do this, we solve the original LP for a given η to get an optimal abstention $A(\eta)$. Next, we add a linear constraint to the LP to enforce that the abstention is equal to $A(\eta)$ and select the solution that maximizes, under these constraints, the quantity $\sum_{i=1}^{m}(\alpha_i-\beta_i)\langle w_0,x_i\rangle$ for Algorithm 6.1 and the quantity $\sum_{i=1}^{m}\mu_i|\langle w_0,x_i\rangle|$ for Algorithm 6.2. Here w_0 is the SVM solution vector. This tie-breaking process improves the performance of Algorithms 6.1 and 6.2

[2] Note the bound we use is optimistic over standard generalization bounds.

on real data.

Experiments: Coverage as a Function of Error. We compare the coverages achieved by Algorithms 6.1 and 6.2 with the baseline selective classifier proposed by [EYW10] (EYW10) as a function of the error guarantee η.

For our comparison purposes, we use linear classifiers on the Breast Cancer, KDD-Cup99 [BL13] (*normal* vs *malicious*) and subsets of the MNIST data [LC05]. kddcup, mnist0v1 and mnist6v9 are linearly separable, while the rest are not. We implement Algorithms 6.1 and 6.2 using the bootstrap sampling scheme described in Subsection 6.3. EYW10 is also implemented using the same bootstrap sample, so all algorithms offer error guarantees with respect to the bootstrapped sample set H. The reference classifier h_0 in EYW10 is chosen as the SVM solution.

Figure 5 shows the results. It is evident from the results that for the same error guarantee η, the coverages of Algorithms 6.1 and 6.2 do not differ much, while EYW10 provides much lower coverage.

Experiments: Risk-Coverage Tradeoffs. We next evaluate the actual risk-coverage tradeoffs of Algorithms 6.1 and 6.2 on real data. The risk of a confidence-rated predictor is defined as the ration between its error and its coverage. In addition to EYW10, we choose as baselines the Agnostic Selective Classification (ASC) algorithm [EYW11] and thresholding based on the distance from the decision boundary (DDB) of the SVM. ASC sorts unlabeled examples based on a *disbelief index* and abstains from prediction when this index lies below a threshold. DDB abstains from prediction on examples whose distance from the decision boundary of an SVM classifier is below a threshold. Each algorithm has a parameter which can be varied to control the risk-coverage tradeoff; we run several iterations of each with different values of these parameters, and plot the corresponding risk (as measured with respect to the actual test labels) as a function of the coverage. DDB does not offer any error guarantees, and ASC only has theoretical error guarantees for zero error with respect to h^*.

Figure 6 shows the results. EYW10 performs the worst, as it treats all points in the disagreement region as equivalent. ASC typically performs the same as DDB or better, while our Algorithm 6.2 is competitive with ASC. An interesting observation is that Algorithm 6.2 usually performs better than Algorithm 6.1 in practice, even though it is worse in theory. This may be explained as follows. Algorithm 6.1 treats all hypotheses in the candidate set the same way, and generates the predicted labels by solving an LP; on the other hand, the labels predicted by

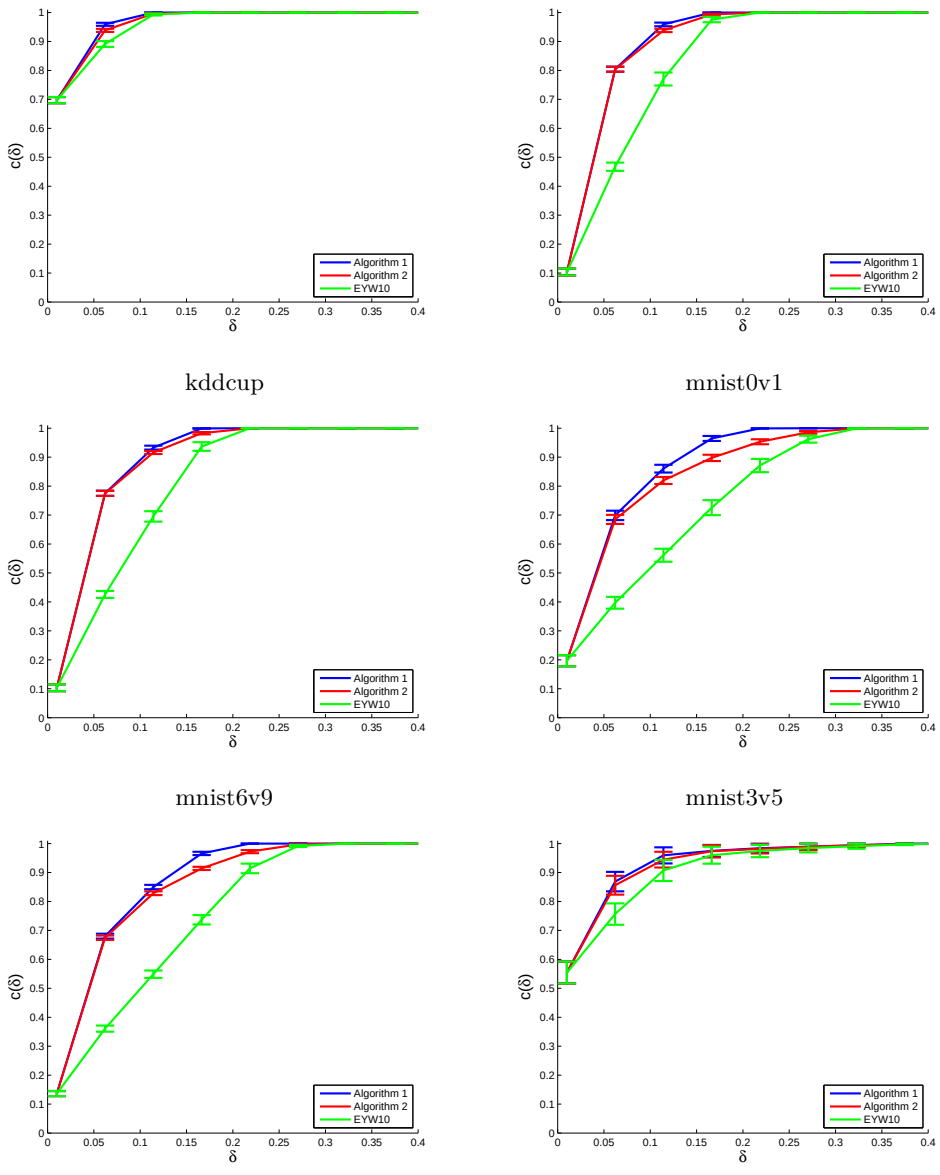

Figure 6.1: Coverage(y-axis) vs. Error guarantee η (x-axis) for the experiment datasets.

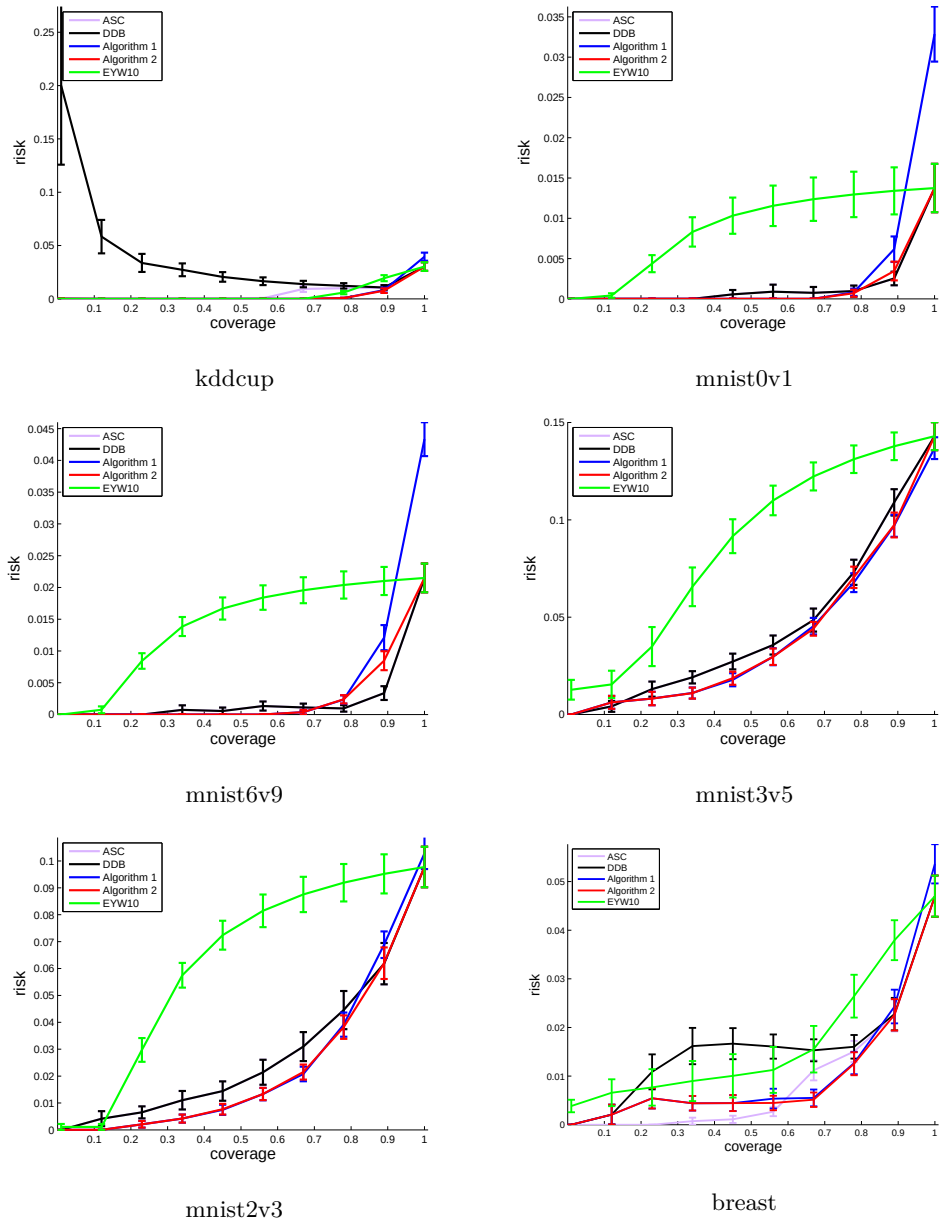

Figure 6.2: Risk(y-axis) vs. coverage(x-axis) tradeoffs for the experiment datasets. Note that ASC overlaps completely with DDB for mnist0v1, mnist2v3, mnist3v5, and mnist6v9.

Algorithm 6.2 always agree with the SVM solution, and as seen from the results on DDB, these labels work quite well in practice.

6.4 Proofs from Section 6.2

Proof of Theorem 6.1. We observe that for unlabeled example x_{n+i}, $\beta_i = \mathbb{P}[P(x_{n+i}) = -1]$ and $\alpha_i = \mathbb{P}[P(x_{n+i}) = +1]$. Suppose $h^* \in \mathcal{H}$ is the true hypothesis which has 0 error with respect to the data distribution. By the realizability assumption, $h^* \in V$. Moreover, $\text{err}_U(P) = \frac{1}{m}\sum_{i:h(x_{n+i})=1}\beta_i + \sum_{i:h(x_{n+i})=-1}\alpha_i \leq \eta$ by Algorithm 6.1. The first part of the lemma follows.

For the second part, suppose P' assigns probabilities $\{(\alpha_i', \beta_i', \gamma_i',), i = 1, \ldots, m\}$ to the unlabeled examples x_{n+i}, and suppose for the sake of contradiction that $\text{abs}_U(P') < \text{abs}_U(P)$. Then, $(\alpha_i', \beta_i', \gamma_i')$'s cannot satisfy the LP in Algorithm 6.1, and thus there exists some $h' \in V$ for which constraint (6.2) is violated. Given S, the true hypothesis that generates the data could be any $h \in V$; if this true hypothesis is h', then $\text{err}_U(P') > \eta$. □

Proof of Theorem 6.2. The first part follows from constraint (6.5), which ensures that if some $h \in V$ is the true hypothesis, then $\text{err}_U(C) \leq \eta$.

For the second part, let $\{(\alpha_i, \beta_i, \gamma_i)\}, i = 1, \ldots, m$ be the optimal solution to the LP in Algorithm 6.1. Pick any $h_0 \in V$. For all $i = 1, \ldots, m$, define $\mu_i = \alpha_i$ if $h_0(x_{n+i}) = 1$ and $\mu_i = \beta_i$ otherwise; define $\bar{\mu}_i = \alpha_i + \beta_i - \mu_i$. Thus for all i, $\alpha_i + \beta_i = \mu_i + \bar{\mu}_i$.

Observe that for all $h \in V$,

$$\begin{aligned}
& \sum_{i:h_0(x_{n+i})\neq h(x_{n+i})} \mu_i \\
= & \sum_{i:h_0(x_{n+i})=-,h(x_{n+i})=+} \mu_i + \sum_{i:h_0(x_{n+i})=+,h(x_{n+i})=-} \mu_i \\
\leq & \sum_{i:h(x_{n+i})=1} \beta_i + \sum_{i:h(x_{n+i})=-1} \alpha_i \\
\leq & \eta m
\end{aligned}$$

Thus, the μ_i's form a feasible solution to the LP in Algorithm 6.2.

Since $h_0 \in V$, from constraint (6.2), $\sum_i \bar{\mu}_i = \sum_{i:h_0(x_{n+i})=1}\beta_i + \sum_{i:h_0(x_{n+i})=-1}\alpha_i \leq \eta m$.

Thus, $\sum_i \mu_i \geq \sum_i \alpha_i + \beta_i - \eta m$. Therefore,

$$\text{abs}_U(C) = \frac{1}{m}\sum_i (1-\mu_i) \leq (\frac{1}{m}\sum_i (1-\alpha_i-\beta_i)) + \eta = \text{abs}_U(P) + \eta.$$

Therefore there exists a feasible solution to the LP in Algorithm 6.2 for which the abstention is at most $\text{abs}_U(P) + \eta$. The theorem follows. □

Proof of Corollary 6.1. Let h^* be the hypothesis in $\mathcal{H}$ which minimizes the true risk $\text{err}(h)$. From the properties of $C(S)$, if S consists of n iid examples drawn from a data distribution D, then with probability $\geq 1-\delta$, $h^* \in C(S)$. The corollary now follows as a consequence of constraint (6.2) in Algorithm 6.1 and constraint (6.5) in Algorithm 6.2. □

Chapter 7

Confidence-rated Prediction in the Online Setting

7.1 Introduction

This chapter studies the problem of online confidence-rated prediction. In settings such as online credit card fraud detection, confidence-rated prediction algorithms are useful, in the sense that when the detecting algorithm is in doubt, it can upload the transaction details to a human expert for a more careful analysis. However, different from batch confidence-rated prediction studied in Chapter 6, in this setting, the transactions come sequentially, and the algorithm needs to makes a decision as soon as a new transaction arrives. Moreover, the transactions are not generated iid - there can be many temporal and spatial localities in the data. Therefore, the algorithms in Chapter 6 does not apply in this setting. Thus, it is of importance to analyze a new model of confidence-rated prediction - that is, confidence-rated prediction in the online setting.

In this chapter, we focus on the model of [SZB10], trading off its mistakes and abstentions in the online setting. Similar to confidence-rated prediction with error guarantees in the batch setting, we consider online confidence-rated prediction where the number of mistakes must not exceed a pre-specified upper bound. While previous work has looked at designing generic online learning algorithm in this setting [LLWS11, SZB10], there are two remaining challenges. First

not much is known about the optimality of these algorithms, and in particular, about what an optimal algorithmic strategy would be for any individual hypothesis class. The second challenge is to understand what happens in a more realistic scenario where the realizability assumption does not hold. While this has been studied in a regression setting [SS11], not much is known about the classification case.

In this chapter, we address both challenges. We first provide a new combinatorial measure that, given a hypothesis class $\mathcal{H}$, captures how many abstentions are needed to ensure a certain number of mistakes. In addition, we provide an optimal algorithm that achieves this number. Our measure is closely related to the notion of Littlestone's dimension for online learning with no abstentions, and we call it the Extended Littlestone's dimension. Formalizing this notion additionally allows us to extend our algorithm to infinite hypothesis classes; while algorithms were previously known for some specific infinite classes [SZB10], no generic algorithm was known.

Next, we focus our attention on the non-realizable case. In this case, we make an l-bias assumption, which ensures that the labels are generated by a function that disagrees with some (unknown) hypothesis $h \in \mathcal{H}$ on at most l examples. We show that (at least some form of) this assumption is necessary; there exists a finite hypothesis class $\mathcal{H}$, such that when the l-bias assumption holds, any algorithm that abstains a finite number of times must make at least l mistakes. Moreover, there also exists an infinite hypothesis class $\mathcal{H}$ with Littlestone's dimension d such that any algorithm that abstains a finite number of times must make at least $l + d$ mistakes. To complement these lower bounds, we show that we can run a version of our algorithm when the l-bias assumption holds, and provide an upper bound on the number of abstentions it makes.

7.2 The Setting

We consider the problem of online classification in the model of [SZB10], where the learner is allowed to occasionally abstain from prediction. Recall from Section 3.5, the interaction between the learning algorithm and the adversary is as follows. At time t, the adversary presents an example x_t in some instance space $\mathcal{X}$. The learner makes its prediction $\hat{y}_t$, which can be either -1, $+1$, or $\perp$ (I don't know). The adversary then reveals an outcome $y_t \in \{-1, +1\}$. The interaction between the learner and the adversary continues, and the performance of the learner is measured by the total number of mistakes and abstentions made throughout the process.

To help make decisions, the learner has access to a hypothesis class $\mathcal{H}$. Each hypothesis h in $\mathcal{H}$ is a prediction rule mapping from $\mathcal{X}$ to $\{-1,+1\}$.

Basic Notations. Given two hypothesis h_1 and h_2, their *product* is defined as a new hypothesis $h_1 \cdot h_2$ which is a function that takes x as input, and outputs $h_1(x) \cdot h_2(x)$. Given two hypothesis classes $\mathcal{H}_1$ and $\mathcal{H}_2$, we define $\mathcal{H}_1 \cdot \mathcal{H}_2$ to be the class of functions achievable by taking the product between a function in $\mathcal{H}_1$ and a function in $\mathcal{H}_2$. Formally,

$$\mathcal{H}_1 \cdot \mathcal{H}_2 = \{h_1 \cdot h_2 : h_1 \in \mathcal{H}_1, h_2 \in \mathcal{H}_2\}$$

Denote by $\mathcal{C}^l$ the class of union of at most l singletons in instance domain $\mathcal{X}$. That is, hypotheses that take value $+1$ on $\mathcal{X}$, except for at most l points:

$$\mathcal{C}^l = \left\{1 - 2I(x = x_1 \vee \ldots \vee x = x_i) : x_1, \ldots, x_i \in \mathcal{X}, i \leq l\right\}$$

In this chapter, we will address both realizable and nonrealizable cases, defined below.

Realizable Case. In the realizable case, we assume there is a hypothesis h in $\mathcal{H}$ that makes no mistakes over time. Formally, a sequence $S = (x_1, y_1), \ldots, (x_n, y_n)$ is called $\mathcal{H}$-realizable, if and only if

$$\exists h \in \mathcal{H}, \quad |\left\{t : h(x_t) \neq y_t\right\}| = 0$$

Non-realizable Case. In the non-realizable case, we assume that the label of examples are generated by a function that disagrees with some hypothesis on at most l examples, which we call l-bias assumption.[1] Formally, define $\mathcal{H}^l$ as the set of classifiers where its prediction differs from some classifier in $\mathcal{H}$ on at most l points, i.e. $\mathcal{H}^l = \mathcal{H} \cdot \mathcal{C}^l$. A sequence $S = (x_1, y_1), \ldots, (x_n, y_n)$ is said to have l-bias with respect to $\mathcal{H}$, if and only if it is $\mathcal{H}^l$-realizable, i.e.

$$\exists h \in \mathcal{H}^l, \quad |\left\{t : h(x_t) \neq y_t\right\}| = 0$$

Version Space and Disagreement Region. Similar to active learning, it is often convenient to consider the set of hypotheses that agree with the labeled examples revealed so far in online confidence-rated prediction. We will also use the same notion in this chapter. Given a set of labeled examples S and a set of hypotheses V, the version space $V[S] \subseteq V$ is defined as

[1] This is similar to the l-mistake assumption in the expert problem [CFHW96, ALW06].

the set of all classifiers that classify S correctly:

$$V[S] = \left\{h \in V : \text{for all } (x,y) \in S, h(x) = y\right\}$$

At the start of time t, the version space is defined as the set of hypotheses in $\mathcal{H}$ that agree with the examples (x_1, y_1), ..., (x_{t-1}, y_{t-1}) seen so far.

We say that an example x is in the disagreement region of a hypothesis set V, denoted by $\text{Dis}(V)$, if both $V[(x,+1)]$ and $V[(x,-1)]$ are nonempty.

7.3 Extended Littlestone's Dimension

We begin with the realizable case and the definition of the Extended Littlestone's Dimension. We first define an extended mistake tree, which is a natural generalization of the mistake tree, and then use it characterize the optimal number of non-trivial rounds (abstentions + mistakes) for any algorithm in the k-SZB model. We finally present an optimal algorithm (Algorithm 7.2) for this model, and a recursive formulation of Extended Littlestone's dimension.

7.3.1 Background: Mistake Bound, Littlestone's Dimension and Standard Optimal Algorithm

[Lit87] provides a characterization of the optimal mistake bound in the realizable case, which is measured by Littlestone's dimension. We begin by describing this characterization.

Mistake Trees. Littlestone's dimension is closely related to the notion of a mistake tree. A mistake tree[2] of a hypothesis class $\mathcal{H}$ is a complete binary tree[3] , whose leaves are classifiers in $\mathcal{H}$ and whose internal nodes correspond to examples in $\mathcal{X}$. A mistake tree may have no internal nodes, in which case it only contains a leaf corresponding to a classifier h in $\mathcal{H}$ – we call it a zeroth order mistake tree. Given an internal node, the edge connecting it and its left (resp. right) child is labeled -1 (resp. $+1$).

A root to leaf path p in mistake tree T is a sequence of nodes and edges, denoted as $v_1 e_1 v_2 e_2 \ldots v_n e_n v_{n+1}$, where $v_1, \ldots, v_n$ are internal nodes in T corresponding to examples in $\mathcal{X}$, v_1 is the root node of T, each e_i is an edge in T that connects v_i and v_{i+1}, $v_{n+1} = h$ is a classifier

[2] In [Lit87] this is instead called a "complete mistake tree"; in [Sha12] this is called a $\mathcal{H}$-shattered tree.
[3] A complete binary tree is one in which every level is completely filled with nodes.

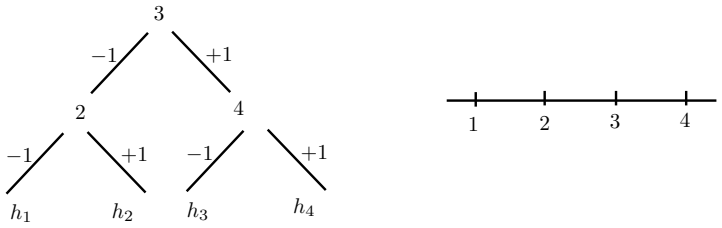

Figure 7.1: A mistake tree with respect to $\mathcal{H} = \{h_i = 2I(x \leq i) - 1 : i = 1,2,3,4\}$, a set of threshold classifiers.

in $\mathcal{H}$ corresponding to a leaf in T. For each i, edge e_i connects v_i and v_{i+1}. The length of a path $l(p)$ is defined as the number of edges in p. For each leaf, the associated classifier agrees with the internal nodes and edges along the path up to the root. That is, if each node v_i corresponds to example x_i and each edge e_i has label y_i, then h agrees with examples $\{(x_1,y_1),\ldots,(x_n,y_n)\}$. See Figure 7.1 for an illustration.

A mistake tree T succinctly represents a strategy of the adversary in response to a deterministic learner. At $t = 1$, the adversary picks the example x_1 corresponding to the root node to show to the learner. If the learner predicts $\hat{y}_1 = -1$, the adversary reveals label $y_1 = +1$, and follows the downward edge labeled $+1$; otherwise it follows the other edge. If at time $t \geq 2$, the adversary reaches a node with example x_t, then x_t is shown to the learner, and one of the downward edges adjacent to this node is followed. The interaction comes to an end when a leaf is reached. It can be seen that the adversary forces the learner to make a mistake at each node of the mistake tree; this implies that if every root-to-leaf path of the mistake tree has depth d, then the adversary can force the learner to make d mistakes using the associated strategy.

We are now ready to define Littlestone's dimension.

Definition 7.1. *The Littlestone's dimension of hypothesis class $\mathcal{H}$,* $\mathrm{Ldim}(\mathcal{H})$*, is the maximum depth of any mistake tree of $\mathcal{H}$.*

Theorem 7.1 ([Lit87])**.** *For a hypothesis class $\mathcal{H}$, the optimal mistake bound of any deterministic algorithm with respect to adversaries showing $\mathcal{H}$-realizable sequences is equal to* $\mathrm{Ldim}(\mathcal{H})$.

Standard Optimal Algorithm. Algorithm 7.1 presents the Standard Optimal Algorithm, which is an optimal deterministic algorithm for online classification in the realizable case. It maintains a version space V over time. At each time t, it predicts a label y_t such that each mistake will force the version space's Littlestone's dimension to drop by at least 1. Therefore, the number of mistakes made by Algorithm 7.1 is at most $\mathrm{Ldim}(\mathcal{H})$.

Algorithm 7.1 Standard Optimal Algorithm [Lit87]

Input: hypothesis class $\mathcal{H}$.
Initialize version space $V \leftarrow \mathcal{H}$.
for $t = 1, 2, \ldots,$ **do**
  Receive example $x_t \in \mathcal{X}$.
  Make prediction $\hat{y}_t = \arg\max_y \mathrm{Ldim}(V[(x, y)])$.
  Receive label y_t.
  if $\hat{y}_t = -y_t$ **then**
    Update version space $V \leftarrow V[(x_t, y_t)]$.
  end if
end for

Recursive Definition. For finite $\mathcal{H}$[4] one also has the following recurrence for its Littlestone's dimension:

$$\mathrm{Ldim}(\mathcal{H}) = \begin{cases} 0 & |\mathcal{H}| = 1 \\ 1 + \max_{x \in \mathrm{Dis}(\mathcal{H})} \min_{y \in \{\pm 1\}} \mathrm{Ldim}(\mathcal{H}[(x, y)]) & |\mathcal{H}| > 1 \end{cases}$$

7.3.2 Extended Littlestone's Dimension

We now define extended Littlestone's Dimension, which measures the difficulty of online learning a hypothesis class in the k-SZB model.

Extended Mistake Trees. An adversary's strategy in response to a deterministic learner in the k-SZB model can be succinctly represented by extended mistake trees. An extended mistake tree for $\mathcal{H}$ is a full[5] binary tree, whose leaves are classifiers in $\mathcal{H}$ and whose internal nodes are examples in $\mathcal{X}$. An extended mistake tree may have no internal node, in which case it only contains a leaf corresponding to a classifier h in $\mathcal{H}$ – we call it a zeroth order extended mistake tree. Unlike mistake trees, now, there are two type of edges: solid and dashed, representing mistakes and abstentions, respectively. Each node is associated with two downward solid edges, one to each child. Additionally, each node is associated with exactly one downward dashed edge connecting to one of its two children. For a downward edge of a node, whether solid or dashed, if it is connected with the node's left child, then it is labeled -1, and vice versa. Just as in mistake

[4] For infinite $\mathcal{H}$, the recurrence may not reach the base case.
[5] A full binary tree is one in which every internal node has exactly two children.

trees, for each leaf, the associated classifier agrees with the internal nodes and edges along the path up to the root. See Figure 7.2 for an illustration.

A root to leaf path p of an extended mistake tree T is a sequence of nodes and edges $v_1 e_1 v_2 e_2 \dots v_n e_n v_{n+1}$, where $v_1, \dots, v_n$ are internal nodes in T corresponding to examples in $\mathcal{X}$, v_1 is the root node of T, each e_i is an edge in T that connects v_i and v_{i+1}, $v_{n+1} = h$ is a classifier in $\mathcal{H}$ corresponding to a leaf in T. Here if there are multiple edges between v_i and v_{i+1}, any one of them can be used by p.

Given an extended mistake tree T, the associated adversarial strategy can be described as follows. At $t = 1$, the adversary chooses the example x_1 corresponding to the root node to show to the learner. If the learner predicts $\hat{y}_1 = -1$, it reveals label $y_1 = +1$, follows the downward solid edge labeled $+1$, and vice versa. Otherwise, if $\hat{y}_1 = \perp$, it reveals y_1 as the label on the dashed edge and follows the downward dashed edge. At time $t \geq 2$, if the adversary reaches a node with example x_t, then x_t is shown to the learner, and one of its adjacent downward edges is followed. The interaction comes to an end when a leaf is reached. It can be seen that with this strategy, the adversary forces every round to be nontrivial. If the depth of the leaf reached is d, then the number of nontrivial rounds is d.

As an example, the extended mistake tree in Figure 7.2 can be used by the adversary as follows. Initially $x_1 = 2$ is presented to the learner. If the learner predicts $\hat{y}_1 = -1$, the adversary reveals label $y_1 = +1$ and follows the right downward solid edge to reach node $x_2 = 3$. At time $t = 2$, the learner now shows example $x_2 = 3$; If the learner predicts $\hat{y}_2 = \perp$, the adversary reveals $y_2 = +1$ according to the label on the dashed edge and follows the edge to reach node $x_3 = 4$. At time $t = 3$, the learner shows example $x_3 = 4$; If the learner predicts $\hat{y}_2 = +1$, the adversary reveals label $y_2 = -1$ and follows the left downward solid edge to reach a leaf containing hypothesis h_3. This concludes the interaction, and the learner makes a total of 3 nontrivial rounds: 2 mistakes and 1 abstentions. Note that realizability assumption is maintained, as $h_3 \in \mathcal{H}$ agrees with the examples $(2, +1)$, $(3, +1)$, $(4, -1)$ shown. More generally, one can show that if the learner is not allowed to make any mistakes, then the adversary is able to force 3 nontrivial rounds by following this strategy. This motivates the definition below.

Definition 7.2. *We say that an extended mistake tree T is (k, m)-difficult for integers $k, m \geq 0$, if all its root to leaf paths in T using at most k solid edges have length at least m.*

For example, the extended mistake tree in Figure 7.2 is $(0, 3)$-difficult.

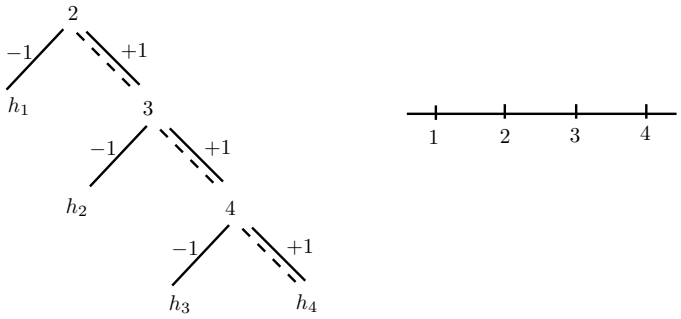

Figure 7.2: An extended mistake tree with respect to $\mathcal{H} = \{h_i = 2I(x \leq i) - 1 : i = 1,2,3,4\}$, a set of threshold classifiers.

Extended Standard Optimal Algorithm. Algorithm 7.2 presents the Extended Standard Optimal Algorithm (SOA.DK), which, as we will show, is an optimal deterministic algorithm for online prediction in the k-SZB model in the realizable case. Note that it works even when the hypothesis class $\mathcal{H}$ is infinite. Similar to the Standard Optimal Algorithm, it maintains a version space V. For a new example x_t, it predicts $\hat{y}_t \in \{-1,+1,\perp\}$ by computing function ELdim over subsets of V. The function ELdim is defined as follows.

Definition 7.3 (Extended Littlestone's Dimension)**.** *For a hypothesis class V and integer $k \geq 0$, the extended Littlestone's dimension* $\mathrm{ELdim}(V,k)$ *is defined as:*

$$\mathrm{ELdim}(V,k) := \sup\{m \in \mathbb{N} : \textit{There exists a } (k,m)\textit{-difficult extended mistake tree for } V\}$$

We remark that if for every integer m, V has a (k,m)-extended mistake tree, then $\mathrm{ELdim}(V,k) = \infty$; If $V = \emptyset$, then $\mathrm{ELdim}(V,k) = -\infty$. Since for $k' < k$, a (k,m)-difficult extended mistake tree is also (k',m)-difficult, $\mathrm{ELdim}(V,k)$ is monotonically nonincreasing with respect to k.

We first show that when $\mathrm{ELdim}(V,k)$ is high, then an adversary can force a large number of nontrivial rounds by showing a V-realizable sequence, to any deterministic algorithms that guarantees at most k mistakes.

Lemma 7.1. *Suppose we are given a hypothesis set V and integers $k \geq 0, m \geq 0$. If* $\mathrm{ELdim}(V,k) \geq m$*, then there is a strategy of the adversary that presents a V-realizable sequence and that can force any deterministic algorithm that guarantees $\leq k$ mistakes to have $\geq m$ nontrivial rounds.*

In the following lemma, we show that given a mistake budget k, if the extended Lit-

tlestone's dimension of V is small, then SOA.DK has a small number of nontrivial rounds for V-realizable sequences.

Lemma 7.2 (Performance Guarantees of SOA.DK)**.** *Suppose we are given a hypothesis class V and integers $k \geq 0, m \geq 0$. If $\mathrm{ELdim}(V,k) \leq m$, then Algorithm 7.2, when run on V with mistake budget k, achieves a (k,m)-SZB bound with respect to any adversary that shows V-realizable sequences.*

Algorithm 7.2 Extended Standard Optimal Algorithm: SOA.DK

Input: hypothesis class $\mathcal{H}$, mistake budget k.
Initialize version space $V \leftarrow \mathcal{H}$.
for $t = 1, 2, \ldots,$ **do**
  Receive example $x_t \in \mathcal{X}$.
  if $x_t \in \mathrm{Dis}(V)$ **then** # All classifiers in V predict unanimously
    Predict $\hat{y}_t = h(x_t)$, where h is an arbitrary hypothesis in V.
  else # There is disagreement among V
    if $k = 0$ **then** # Zero mistake budget, must output $\perp$
      Predict $\hat{y}_t = \perp$.
    else # Predict by minimizing the ELdim of future version space
      Compute $m_{+1} = \mathrm{ELdim}(V[(x_t,-1)], k-1)$, $m_{-1} = \mathrm{ELdim}(V[(x_t,+1)], k-1)$, and $m_{\perp} = \max(\mathrm{ELdim}(V[(x_t,-1)], k), \mathrm{ELdim}(V[(x_t,+1)], k))$
      Predict $\hat{y}_t = \arg\min\{m_y : y \in \{-1,+1,\perp\}\}$.
    end if
  end if
  Receive label y_t.
  if $\hat{y}_t = -y_t$ or $\hat{y}_t = \perp$ **then** $V \leftarrow V[(x_t, y_t)]$ **end if** # Update version space
  if $\hat{y}_t = -y_t$ **then** $k \leftarrow k-1$ **end if** # Update mistake budget
end for

An immediate consequence of Lemma 7.2 is that SOA.DK is optimal, in the sense that it has the smallest number of worst case nontrivial rounds, amongst all *deterministic* algorithms that work in k-SZB model.

Theorem 7.2 (Optimality of SOA.DK)**.** *Suppose we are given a hypothesis class $\mathcal{H}$ and integers $k \geq 0$, $m \geq 1$ such that $\mathrm{ELdim}(\mathcal{H},k) = m$. Then:*

(a) SOA.DK achieves a (k,m)-SZB bound for any adversary that shows $\mathcal{H}$-realizable sequences.

(b) There exists an adversary showing $\mathcal{H}$-realizable sequences, such that no deterministic algorithm $\mathcal{A}$ can achieve a $(k,m-1)$-SZB bound.

The following simple property relates extended Littlestone's dimension to Littlestone's dimension.

Theorem 7.3 (Relating Ldim to ELdim)**.** *Suppose we are given a hypothesis class $\mathcal{H}$. If* $\mathrm{Ldim}(\mathcal{H}) = d < \infty$*, then*

$$\mathrm{ELdim}(\mathcal{H}, d) = d$$

Recursive Definition. We provide a recursive characterization of Extended Littlestone's dimension. For finite $\mathcal{H}$[6], the following recurrence holds for its extended Littlestone's dimension:

$$\mathrm{ELdim}(\mathcal{H}, k) = \begin{cases} 0 & |\mathcal{H}| = 1 \\ 1 + \max\limits_{x \in \mathrm{Dis}(\mathcal{H})} \max\limits_{y \in \{\pm 1\}} \mathrm{ELdim}(\mathcal{H}[(x,y)], 0) & |\mathcal{H}| > 1, k = 0 \\ 1 + \max\limits_{x \in \mathrm{Dis}(\mathcal{H})} \max\limits_{y \in \{\pm 1\}} \min(\mathrm{ELdim}(\mathcal{H}[(x,y)], k-1), \mathrm{ELdim}(\mathcal{H}[(x,-y)], k)) & |\mathcal{H}| > 1, k \geq 1 \end{cases}$$

The recurrence is an immediate consequence of Lemma 7.11 in Section 7.9.

7.4 Properties of Extended Littlestone's Dimension

We next present upper bounds on the Extended Littlestone's Dimension of a hypothesis class $\mathcal{H}$. Our upper bounds depend on the *tree shattering coefficient*, a notion analogous to the growth function, which is implicit in [BPS09]. We also present some examples of Extended Littlestone's Dimension.

7.4.1 Tree Shattering Coefficient

The shattering coefficient (also known as the growth function), initially studied in [VC71], is a key notion in PAC learnability.

Definition 7.4. *Given a hypothesis $\mathcal{H}$, the shattering coefficient of $\mathcal{H}$, $\Pi(\mathcal{H}, t)$ is defined as the maximum number of labelings achievable by $\mathcal{H}$ over t points. Formally,*

$$\Pi(\mathcal{H}, t) := \max_{x_1, \ldots, x_t} \left| \left\{ (h(x_1), \ldots, h(x_t)) : h \in \mathcal{H} \right\} \right|$$

[6] Just as with Littlestone's dimension, for infinite $\mathcal{H}$, the recurrence may not reach the base case.

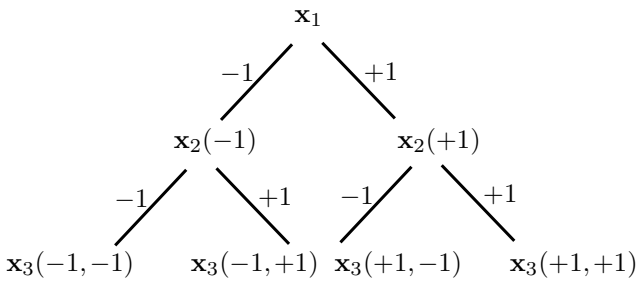

Figure 7.3: A depth-3 $\mathcal{X}$-valued tree $\mathbf{x}$.

Inspired by the shattering coefficient, in online learning, we define the notion of *tree shattering coefficient* below, implicit in [BPS09]. As we will see, this notion is crucial to online learnability in both the mistake bound and the k-SZB models. First we set up our notation by adopting the notion of trees in [RST10].

Definition 7.5 ($\mathcal{X}$-valued Trees, see [RST10])**.** *A depth-t $\mathcal{X}$-valued tree $\mathbf{x}$ is a series of mappings $(\mathbf{x}_1,\ldots,\mathbf{x}_t)$, where $\mathbf{x}_i : \{\pm 1\}^{i-1} \to \mathcal{X}$. The root of the tree $\mathbf{x}$ is the constant function $\mathbf{x}_1 \in \mathcal{X}$. For integer t, the mapping $\mathbf{x}_t(\cdot)$ takes care of the nodes in level t.*

To see why a series of mappings corresponds to a tree, we first note that a tuple $(\epsilon_1,\ldots,\epsilon_{s-1})$ in $\{\pm 1\}^{s-1}$ can be thought of as a left/right sequence of length $s-1$, where -1 stands for left and $+1$ stands for right, respectively. The node reached from the root following the path corresponding to the left/right sequence corresponds to $\mathbf{x}_s(\epsilon_1,\ldots,\epsilon_{s-1}) \in \mathcal{X}$. For example, the root node corresponds to $\mathbf{x}_1 \in \mathcal{X}$, the left child of the root corresponds to $\mathbf{x}_2(-1) \in \mathcal{X}$, the right child of the left child of the root corresponds to $\mathbf{x}_3(-1,+1) \in \mathcal{X}$, etc. See Figure 7.3 for an illustration. We slightly abuse the notation to let $\mathbf{x}_t(\epsilon)$ denote $\mathbf{x}_t(\epsilon_1,\ldots,\epsilon_{t-1})$.

Note that a $\mathcal{X}$-valued tree is not a mistake tree or an extended mistake tree, since it does not have leaves corresponding to hypotheses in $\mathcal{H}$.

Given a $\mathcal{X}$-valued tree $\mathbf{x}$, we add an extra level of edges at the bottom. Specifically for each leaf $\mathbf{x}_t(\epsilon)$, we attach a left and a right downward edge onto it, labeled -1 and $+1$ respectively. Now, consider every root to leaf path in the tree. If there is some classifier h in $\mathcal{H}$ that agrees with the path, we label the leaf with h; otherwise we label the leaf with symbol $\times$. We count the number of leaves not labeled $\times$, denoted by function $S(\mathcal{H},\mathbf{x})$. See Figure 7.4 for an example.

Definition 7.6. *For a depth-t $\mathcal{X}$-valued tree $\mathbf{x}$, and a hypothesis class $\mathcal{H}$, define function $S(\mathcal{H},\mathbf{x})$*

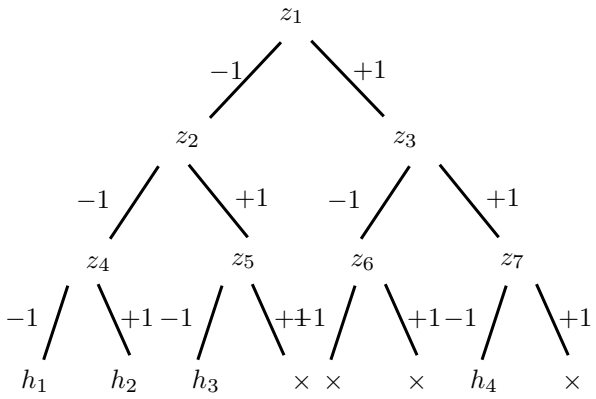

Figure 7.4: A depth-3 $\mathcal{X}$-valued tree $\mathbf{x}$, where $\mathbf{x}_1 = z_1$, $\mathbf{x}_2(-1) = z_2$, $\mathbf{x}_2(+1) = z_3$, $\mathbf{x}_3(-1,-1) = z_4$, $\mathbf{x}_3(-1,+1) = z_5$, $\mathbf{x}_3(+1,-1) = z_6$, $\mathbf{x}_3(+1,+1) = z_7$. There are 4 root to leaf paths that agrees with some hypothesis in $\mathcal{H}$ ($\times$ in a leaf indicates that no hypothesis in $\mathcal{H}$ agree with the path from root to it), i.e. $|S(\mathcal{H},\mathbf{x})| = 4$.

as the maximum number of labelings achievable by $\mathcal{H}$ on $\mathbf{x}$. Formally,

$$S(\mathcal{H},\mathbf{x}) := \left\{(\epsilon_1,\epsilon_2,\ldots,\epsilon_t) \in \{\pm 1\}^t : \exists h \in \mathcal{H}, \epsilon_1 = h(\mathbf{x}_1(\epsilon)), \epsilon_2 = h(\mathbf{x}_2(\epsilon)), \ldots, \epsilon_t = h(\mathbf{x}_t(\epsilon)),\right\}$$

Definition 7.7. *Given hypothesis class $\mathcal{H}$, and integer $t \geq 1$, the tree shattering coefficient of $\mathcal{H}$, $\mathcal{S}(\mathcal{H},t)$ is defined as the maximum number of labelings achievable by $\mathcal{H}$ over all depth-t trees. Formally,*

$$\mathcal{S}(\mathcal{H},t) := \max_{\mathbf{x}} |S(\mathcal{H},\mathbf{x})|$$

Additionally, define $\mathcal{S}(\mathcal{H},0) := 1$ if $\mathcal{H}$ is nonempty, $\mathcal{S}(\mathcal{H},0) := 0$ if $\mathcal{H}$ is empty.

In other words, given hypothesis class $\mathcal{H}$ and a depth-t tree $\mathbf{x}$ with internal nodes only, there are at most $\mathcal{S}(\mathcal{H},t)$ distinct paths in T consistent with some classifier $h \in \mathcal{H}$. Since a depth-t has at most 2^t root to leaf paths, $\mathcal{S}(\mathcal{H},t) \leq 2^t$. Note that if $\mathcal{H}$ has a depth-t mistake tree, then $\mathcal{S}(\mathcal{H},t) = 2^t$. If we constrain the trees chosen to be constant among nodes in the same depth, then the tree shattering coefficient is equivalent to the shattering coefficient. In Section 7.6, we show that the tree shattering coefficient is connected with the sequential growth function(maximal sequential zero covering number), defined in [RST10].

The following two lemmas give bounds on tree shattering coefficients, implicit in [RST10, BPS09]. For finite hypothesis class $\mathcal{H}$, its tree shattering coefficient is at most the size of $|\mathcal{H}|$.

Lemma 7.3. *If $\mathcal{H}$ is finite, then for any $t \geq 0$, $\mathcal{S}(\mathcal{H},t) \leq |\mathcal{H}|$.*

Furthermore, if an infinite hypothesis class $\mathcal{H}$ has Littlestone's dimension $d < \infty$, its tree shattering coefficient is polynomial in t, that is, $O(t^d)$.

Lemma 7.4. *If $\mathcal{H}$ has Littlestone's dimension $d < \infty$, then for any $t \geq 0$, $\mathcal{S}(\mathcal{H}, t) \leq \binom{t}{\leq d}$.*

7.4.2 Upper Bound on Extended Littlestone's Dimension

We present Theorem 7.4, the main result of this section, which upper bounds the extended Littlestone's dimension in terms of tree shattering coefficient. Intuitively, if $\mathcal{H}$ is not expressive, then it has small tree shattering coefficient, and a tighter upper bound on its extended Littlestone's dimension can be established. Note that the bound is valid even if $\mathcal{H}$ is infinite, and hence it is a strict generalization of [SZB10].

Theorem 7.4. *For any hypothesis class $\mathcal{H}$ and integer $k \geq 0$,*

$$\text{ELdim}(\mathcal{H}, k) \leq \sup \left\{ t : \binom{t}{\leq k+1} \leq \mathcal{S}(\mathcal{H}, t) \right\}$$

For finite hypothesis classes one has the following corollary.

Corollary 7.1. *For any hypothesis class $\mathcal{H}$ such that $|\mathcal{H}| < \infty$ and integer $k \geq 0$,*

$$\text{ELdim}(\mathcal{H}, k) \leq \max \left\{ t : \binom{t}{\leq k+1} \leq |\mathcal{H}| \right\}$$

Since $\binom{t}{\leq k+1} \geq (\frac{t}{k+1})^{k+1}$, this implies $\text{ELdim}(\mathcal{H}, k) \leq \max \left\{ t : (\frac{t}{k+1})^{k+1} \leq |\mathcal{H}| \right\} \leq (k+1)|\mathcal{H}|^{\frac{1}{k+1}}$, which recovers the result of [SZB10]. [7]

7.4.3 Case Study: Thresholds (Finite Class)

We give a precise characterization of the Extended Littlestone's dimension for the class of thresholds. In this case, the bound given by Theorem 7.4 is tight.

[7] Although it is implicit in [SZB10] that the result can be refined by using the optimal solution of the Egg Dropping Game [GF08, Boa04], here we give a alternative proof using a more general technique.

Consider the instance domain $\mathcal{X}$ being $\mathbb{R}$ and the hypothesis class $\mathcal{H}$ being the set of n distinct threshold functions $\left\{2I(x \leq t) - 1 : t \in \{t_1, \ldots, t_n\}\right\}$. [8]

Theorem 7.5. *Consider $\mathcal{H}$ a set of threshold classifiers $\mathcal{H} = \left\{2I(x \leq t) - 1 : t \in \{t_1, \ldots, t_n\}\right\}$. Then*

$$\text{ELdim}(\mathcal{H}, k) = \max\left\{t : \binom{t}{\leq k+1} \leq n\right\}$$

The proof of Theorem 7.5 is provided in Section 7.10. The upper bound follows immediately from Corollary 7.1. The lower bound comes from an explicit construction of optimal extended mistake trees by exploiting structure in the class of threshold classifiers.

7.4.4 Case Study: Union of Singletons (Infinite Class)

We give a precise characterization of the Extended Littlestone's dimension for the class of unions of singletons. In this case the bound given by Theorem 7.4 is tight. Consider the concept class of union of at most l singletons $\mathcal{C}^l$, with instance domain $\mathcal{X}$ such that $|\mathcal{X}| = \infty$. Note that $\text{Ldim}(\mathcal{C}^l) = l$ and $\mathcal{S}(\mathcal{C}^l, t) = \binom{t}{\leq l}$(See Lemma 7.15 for a proof). We have the following result.

Theorem 7.6. *Consider the hypothesis class $\mathcal{C}^l$, the class of union of at most l singletons. Then,*

$$\text{ELdim}(\mathcal{C}^l, k) = \sup\left\{t : \binom{t}{\leq k+1} \leq \binom{t}{\leq l}\right\} = \begin{cases} \infty, & k \leq l-1 \\ l, & k \geq l \end{cases}$$

Note that Theorem 7.6 involves infinite hypothesis classes and is broader than the results of [SZB10]. The proof of Theorem 7.6 is provided in Section 7.10. The upper bound follows immediately from Theorem 7.4. The lower bound comes from an explicit construction of optimal extended mistake trees by exploiting structures in the class of union of singleton classifiers.

[8] Note that for an infinite set of thresholds, e.g. $\mathcal{H} = \left\{2I(x \leq t) - 1 : t \in [0,1]\right\}$, $\text{Ldim}(\mathcal{H}) = \infty$, hence $\text{ELdim}(\mathcal{H}, k) = \infty$ for any finite k.

7.5 Non-Realizable Case

We now consider the non-realizable case. For the rest of the section, we assume the l-bias assumption holds, i.e. the sequence $(x_1, y_1), \ldots, (x_n, y_n)$ presented by the adversary is $\mathcal{H}^l$-realizable. Recall that $\mathcal{H}^l = \mathcal{H} \cdot \mathcal{C}^l$, the class of hypothesis that disagrees with $\mathcal{H}$ on at most l points.

7.5.1 Lower Bounds for Deterministic Prediction

A natural question is, when the l-bias assumption holds, is it possible to derive algorithms with a small number of abstentions in k-SZB model? Perhaps surprisingly, the answer depends on whether $\mathcal{H}$ is finite or not. We show next that there is a finite hypothesis class $\mathcal{H}$, such that for any $k < l$, and any integer m, any algorithm which is guaranteed to make k or less mistakes can be forced to abstain at least m times. Moreover, for any m, there is a infinite hypothesis class with Littlestone's dimension d, such that for any $k < l + d$, any algorithm that is guaranteed to make k or less mistakes can be forced to abstain at least m times.

Lower Bounds for Finite Hypothesis Classes

We first show that, for finite hypothesis classes $\mathcal{H}$, when $k < l$, no algorithm can guarantee a (k, m)-SZB bound with finite m under the l-bias assumption.

Theorem 7.7. *There exists an instance domain $\mathcal{X}$, a single-element hypothesis class $\mathcal{H}$, such that the following holds. If $k < l$, then for any integer $m \geq 0$, there exists a strategy of the adversary satisfying the l-bias assumption that forces any deterministic algorithm guaranteeing at most k mistakes to have at least $m + 1$ nontrivial rounds.*

Lower Bounds for Infinite Hypothesis Classes

We show that, given a hypothesis classes $\mathcal{H}$ with $\mathrm{Ldim}(\mathcal{H}) = d$, when $k < l + d$, no algorithm can guarantee a (k, m)-SZB bound for finite m under the l-bias assumption.

Theorem 7.8. *There exists an instance domain $\mathcal{X}$, a hypothesis class $\mathcal{H}$ with Littlestone's dimension $d < \infty$, such that the following holds. If $k < l + d$, then for any integer $m \geq 0$, there exists a strategy of the adversary satisfying the l-bias assumption that forces any deterministic algorithm guaranteeing at most k mistakes to have at least $m + 1$ nontrivial rounds.*

7.5.2 Upper Bounds

Upper Bounds for Finite Hypothesis classes

Since a sequence satisfying the l-bias assumption is $\mathcal{H}^l$-realizable, to provide an upper bound on the number of non-trivial rounds under this assumption, we need to provide an upper bound on $\text{ELdim}(\mathcal{H}^l, k)$. We now provide such upper bounds on arbitrary finite hypothesis classes $\mathcal{H}$. Note that since the hypothesis class $\mathcal{H}^l$ is infinite, this result is more general than the kind of results in [SZB10].

Lemma 7.5. *Suppose we are given a finite hypothesis class $\mathcal{H}$, integer $k \geq 0$, $l \geq 0$ such that $k \geq l$. Then,*

$$\text{ELdim}(\mathcal{H}^l, k) \leq e(k+1) \cdot |\mathcal{H}|^{\frac{1}{k+1-l}}$$

Corollary 7.2. *Suppose we are given a finite hypothesis class $\mathcal{H}$ and integers $k, l \geq 0$ such that $k \geq l$. If Algorithm SOA.DK is run with input hypothesis class $\mathcal{H}^l$ and mistake budget k, then for any adversary that shows sequences satisfying the l-bias assumption with respect to $\mathcal{H}$, SOA.DK makes at most k mistakes and has at most $e(k+1) \cdot |\mathcal{H}|^{\frac{1}{k+1-l}}$ nontrivial rounds.*

Upper Bounds for Infinite Hypothesis classes

We now derive a corresponding upper bound for infinite hypothesis classes $\mathcal{H}$ with finite Littlestone's dimension.

Lemma 7.6. *Suppose we are given a hypothesis class $\mathcal{H}$ with Littlestone's dimension $d < \infty$, integer $k \geq 0$, $l \geq 0$ such that $k \geq l + d$. Then,*

$$\text{ELdim}(\mathcal{H}^l, k) \leq (k+1) \cdot e^{\frac{2k+2}{k+1-l-d}}$$

Corollary 7.3. *Suppose we are given a hypothesis class $\mathcal{H}$ with Littlestone's dimension d and integer $k, l \geq 0$ such that $k \geq l + d$. If Algorithm SOA.DK is run with input hypothesis class $\mathcal{H}^l$ and mistake budget k, then for any adversary that shows sequences satisfying the l-bias assumption with respect to $\mathcal{H}$, SOA.DK makes at most k mistakes and has at most $(k+1) \cdot e^{\frac{2k+2}{k+1-l-d}}$ nontrivial rounds.*

7.5.3 Lower Bounds for Randomized Prediction

We show that the results in Section 7.5.1 hold even when the learner makes soft predictions.

Randomized Prediction Model. Consider the following randomized variant of online classification model. At time t, the adversary presents example x_t in $\mathcal{X}$, and the learner outputs a tuple $(p_{t,-}, p_{t,+}, 1-p_{t,-}-p_{t,+})$, with $p_{t,-} \geq 0$, $p_{t,+} \geq 0$ and $1-p_{t,-}-p_{t,+} \geq 0$. The tuple $(p_{t,-}, p_{t,+}, 1-p_{t,-}-p_{t,+})$ represents the learner's strategy of predicting $+1$ with probability $p_{t,+}$, -1 with probability $p_{t,-}$ and abstaining with probability $1-p_{t,+}-p_{t,-}$. The adversary then reveals an outcome $y_t \in \{-1,+1\}$, and the learner incurs a mistake penalty of $p_{t,+}$ if $y_t = -1$ and $p_{t,-}$ if $y_t = 1$; it also incurs an abstention penalty of $1-p_{t,+}-p_{t,-}$. When $p_{t,+}$ and $p_{t,-}$ take values in $\{0,1\}$, observe that this is equivalent to our prediction model in Section 7.2.

So given examples $(x_1, y_1), \ldots, (x_n, y_n)$, the cumulative mistake penalty upto time n is as $\sum_{t=1}^{n} I(y_t = -1)p_{t,+} + I(y_t = +1)p_{t,-}$ and the cumulative abstention penalty is $\sum_{t=1}^{n}(1-p_{t,+}-p_{t,-})$. We have the following result for finite hypothesis classes.

Theorem 7.9. *There exists an instance domain $\mathcal{X}$, a single-element hypothesis class $\mathcal{H}$, such that the following holds. If $k < l$, then for any $a \geq 0$, there exists a strategy of the adversary satisfying l-bias assumption, such that any algorithm guaranteeing a cumulative mistake penalty at most k in the randomized prediction model must have cumulative abstention penalty at least a.*

For infinite hypothesis classes with Littlestone's dimension d, we have the following result.

Theorem 7.10. *There exists an instance domain $\mathcal{X}$ and a hypothesis class $\mathcal{H}$ with Littlestone's dimension d such that the following holds. If $k < l + d$, then for any $a \geq 0$, there exists a strategy of the adversary satisfying the l-bias assumption, such that any algorithm guaranteeing a mistake penalty of at most k in the randomized prediction model must have cumulative abstention penalty at least a.*

7.6 The Relationship Between Tree Shattering Coefficient and Sequential Growth Function

In this section, we show that the tree shattering coefficient $\mathcal{S}(\mathcal{H},t)$ is at most the size of the sequential growth function(also known as maximal sequential zero covering number) of $\mathcal{H}$ [RST10]. We start with some notations.

Definition 7.8 (Sequential Zero Cover and Sequential Zero Covering Number, see [RST10])**.** *A set V of depth-t trees is a sequential zero cover of $\mathcal{H}$ on a depth-t tree $\mathbf{x}$, if*

$$\forall h \in \mathcal{H}, \forall \epsilon \in \{\pm 1\}^t, \exists \mathbf{v} \in V, s.t. \mathbf{v}_s(\epsilon) = h(\mathbf{x}_s(\epsilon)), s = 1,2,\ldots,t$$

The sequential zero covering number of a hypothesis class $\mathcal{H}$ on a given tree $\mathbf{x}$ is defined as

$$\mathcal{N}(0,\mathcal{H},\mathbf{x}) := \min\left\{|V| : V \text{ is a zero-cover of } \mathcal{H} \text{ on } \mathbf{x}\right\}$$

The maximal sequential zero covering number is the maximum sequential zero covering number of $\mathcal{H}$ over all depth-t $\mathcal{X}$-valued trees $\mathbf{x}$, that is,

$$\mathcal{N}(0,\mathcal{H},t) := \max_{\mathbf{x}} \mathcal{N}(0,\mathcal{H},\mathbf{x})$$

Theorem 7.11. *For a given hypothesis class $\mathcal{H}$ and integer $t \geq 0$,*

$$\mathcal{S}(\mathcal{H},t) \leq \mathcal{N}(0,\mathcal{H},t)$$

Proof. This is an immediate consequence of Lemma 7.7. □

Lemma 7.7. *Suppose we are given a $\mathcal{X}$-valued tree $\mathbf{x}$ and a hypothesis class $\mathcal{H}$. If V is a sequential zero cover of $\mathcal{H}$ on $\mathbf{x}$, then the size of $S(\mathcal{H},\mathbf{x})$ is at most $|V|$.*

Proof. Recall that

$$S(\mathcal{H},\mathbf{x}) = \left\{(\epsilon_1,\epsilon_2,\ldots,\epsilon_t) \in \{\pm 1\}^t : \epsilon_1 = h(\mathbf{x}_1(\epsilon)), \epsilon_2 = h(\mathbf{x}_2(\epsilon)), \ldots, \epsilon_t = h(\mathbf{x}_t(\epsilon)), h \in \mathcal{H}\right\}$$

Given an element $(\epsilon_1, \epsilon_2, \ldots, \epsilon_t)$ in $S(\mathcal{H}, \mathbf{x})$, there exists some h in $\mathcal{H}$ such that

$$\epsilon_1 = h(\mathbf{x}_1(\epsilon)), \epsilon_2 = h(\mathbf{x}_2(\epsilon)), \ldots, \epsilon_t = h(\mathbf{x}_t(\epsilon))$$

Since V is a zero-cover of $\mathcal{H}$, there exists a depth-t tree $\mathbf{v} = (\mathbf{v}_1, \ldots, \mathbf{v}_t)$ in V such that

$$\mathbf{v}_1(\epsilon) = h(\mathbf{x}_1(\epsilon)), \mathbf{v}_2(\epsilon) = h(\mathbf{x}_2(\epsilon)), \ldots, \mathbf{v}_t(\epsilon) = h(\mathbf{x}_t(\epsilon))$$

Hence,

$$\mathbf{v}_1(\epsilon) = \epsilon_1, \mathbf{v}_2(\epsilon) = \epsilon_2, \ldots, \mathbf{v}_t(\epsilon) = \epsilon_t$$

More explicitly,

$$\mathbf{v}_1 = \epsilon_1, \mathbf{v}_2(\epsilon_1) = \epsilon_2, \mathbf{v}_3(\epsilon_1, \epsilon_2) = \epsilon_3, \ldots, \mathbf{v}_t(\epsilon_1, \ldots, \epsilon_{t-1}) = \epsilon_t \tag{7.1}$$

To summarize, for every $(\epsilon_1, \epsilon_2, \ldots, \epsilon_t)$ in $S(\mathcal{H}, \mathbf{x})$, there is a tree $\mathbf{v}$ in V such that Equation (7.1) holds. Since for each tree $\mathbf{v}$ there can be at most one $(\epsilon_1, \epsilon_2, \ldots, \epsilon_t)$ such that Equation (7.1) holds, this implies that $|S(\mathcal{H}, \mathbf{x})| \leq |V|$. □

7.7 Reducing the Expert Problem to Online Classification with Finite Class

In this section, we show that the problem of Prediction with Expert Advice (abbrev. Expert Problem) with l-mistake assumption [CFHW96, ALW06] can be cast to the problem studied in this chapter, i.e. online classification with a finite hypothesis class with l-bias assumption. Specifically, in the expert problem, at each time t, the algorithm is given experts' advice $(x_{1,t}, \ldots, x_{N,t}) \in \{-1, 1\}^N$, and predicts $\hat{y}_t \in \{-1, 1, \perp\}$. Then adversary reveals label $y_t \in \{-1, 1\}$. The l-mistake assumption states that there is an expert i that makes at most l mistakes throughout the process, i.e.

$$\exists i, |\{t : x_{i,t} \neq y_t\}| \leq l$$

For $i = 1,2,\ldots,N$, define hypothesis $h_i : \mathbb{R}^{N+1} \to \mathbb{R}$ as mapping a $(N+1)$-dimensional vector to its ith coordinate. Define hypothesis class $\mathcal{H}_N := \{h_i : i = 1,\ldots,N\}$. We have the following result relating the l-mistake assumption to l-bias assumption; the intuition is to concatenate a new coordinate at the end of the experts' advice to make all the examples shown distinct.

Proposition 7.1. *The following are equivalent:*

(a) The sequence of expert advice and labels $(x_{1,t},\ldots,x_{N,t}), y_t, t = 1,2,\ldots$ satisfies l-mistake assumption.

(b) The sequence $x_t = (x_{1,t},\ldots,x_{N,t},t), y_t, t = 1,2,\ldots$ satisfies l-bias assumption with respect to $\mathcal{H}_N$.

Proof. We show the implication in both directions.

($\Rightarrow$) If $(x_{1,t},\ldots,x_{N,t}), y_t, t = 1,2,\ldots$ satisfies l-mistake assumption, then there is $i \in \{1,\ldots,N\}$ such that

$$M_i = |\left\{t : x_{i,t} \neq y_t\right\}| \leq l$$

Hence, h_i is correct on all but the rounds t in M_i, i.e. on examples $\left\{(x_{1,t},\ldots,x_{N,t},t) : t \in M_i\right\}$, which are distinct and has size at most l. Therefore, the sequence $\left\{(x_{1,t},\ldots,x_{N,t},t)\right\}, t = 1,2,..$ satisfies l-bias assumption with respect to $\mathcal{H}_N$.

($\Leftarrow$) If the sequence $(x_{1,t},\ldots,x_{N,t},t), y_t, t = 1,2,\ldots$ satisfies l-bias assumption with respect to $\mathcal{H}_N$, then there exists h_i that is correct on all but $p \leq l$ examples shown. That is, p, the size of the set

$$M_i = |\left\{t : x_{i,t} \neq y_t\right\}|$$

is at most l. This immediately implies (a). □

An immediate consequence of the above proposition is that, for an instance of the expert problem with l-mistake assumption, we can convert it to an instance of online classification in $\mathcal{H}_N$ under l-bias assumption, and apply SOA.DK on $\mathcal{H}_N$ to get mistake-abstention tradeoffs.

7.8 A Note on the Recursive Definition of ELdim

At the end of Section 7.3, we give a recursive definition on $\mathrm{ELdim}(\mathcal{H},k)$ when $k \geq 1$:

$$\begin{aligned}\mathrm{ELdim}(\mathcal{H},k) &:= \max_x \max_{y\in\{-1,+1\}} \min\big(\mathrm{ELdim}(\mathcal{H}[(x,y)],k), \mathrm{ELdim}(\mathcal{H}[(x,-y)],k-1)\big) \\ &= \max_x \max\Big(\min\big(\mathrm{ELdim}(\mathcal{H}[(x,-1)],k), \mathrm{ELdim}(\mathcal{H}[(x,+1)],k-1)\big), \\ &\qquad \min\big(\mathrm{ELdim}(\mathcal{H}[(x,+1)],k), \mathrm{ELdim}(\mathcal{H}[(x,-1)],k-1)\big)\Big) \quad (7.2)\end{aligned}$$

On the other hand, Algorithm 7.2 implicitly implies

$$\begin{aligned}\mathrm{ELdim}(\mathcal{H},k) &= \max_x \min\Big(\mathrm{ELdim}(\mathcal{H}[(x,-1)],k-1), \mathrm{ELdim}(\mathcal{H}[(x,+1)],k-1)\big), \\ &\qquad \max\big(\mathrm{ELdim}(\mathcal{H}[(x,+1)],k), \mathrm{ELdim}(\mathcal{H}[(x,-1)],k)\big)\Big) \quad (7.3)\end{aligned}$$

In this section we show that these two definition are indeed equivalent. First we need a simple observation.

Lemma 7.8. *If A,B,C are real numbers, then* $\min(\max(A,B),C) = \max(\min(A,C),\min(B,C))$.

Lemma 7.9. *The right hand sides of Equations* (7.2) *and* (7.3) *are equal.*

Proof. Fix x in $\mathcal{X}$. We have:

$$\begin{aligned}
& \min\Big(\mathrm{ELdim}(\mathcal{H}[(x,-1)],k-1), \mathrm{ELdim}(\mathcal{H}[(x,+1)],k-1)\Big), \\
& \quad \max\big(\mathrm{ELdim}(\mathcal{H}[(x,+1)],k), \mathrm{ELdim}(\mathcal{H}[(x,-1)],k)\big)\Big) \\
= \quad & \min\Bigg(\mathrm{ELdim}(\mathcal{H}[(x,-1)],k-1), \\
& \quad \min\Big(\mathrm{ELdim}(\mathcal{H}[(x,+1)],k-1)\Big), \max\big(\mathrm{ELdim}(\mathcal{H}[(x,+1)],k), \mathrm{ELdim}(\mathcal{H}[(x,-1)],k)\big)\Big)\Bigg) \\
= \quad & \min\Bigg(\mathrm{ELdim}(\mathcal{H}[(x,-1)],k-1), \\
& \quad \max\Big(\mathrm{ELdim}(\mathcal{H}[(x,+1)],k), \min\big(\mathrm{ELdim}(\mathcal{H}[(x,-1)],k), \mathrm{ELdim}(\mathcal{H}[(x,+1)],k-1)\big)\Big)\Bigg) \\
= \quad & \max\Bigg(\min\Big(\mathrm{ELdim}(\mathcal{H}[(x,-1)],k-1), \mathrm{ELdim}(\mathcal{H}[(x,+1)],k)\Big), \\
& \quad \min\Big(\mathrm{ELdim}(\mathcal{H}[(x,-1)],k), \mathrm{ELdim}(\mathcal{H}[(x,+1)],k-1)\Big)\Bigg)
\end{aligned}$$

where the first equality is from the associativity of min; the second equality is from Lemma 7.8 and $\mathrm{ELdim}(\mathcal{H}[(x,+1)],k-1) \geq \mathrm{ELdim}(\mathcal{H}[(x,+1)],k)$; the third equality is from Lemma 7.8 and $\mathrm{ELdim}(\mathcal{H}[(x,-1)],k-1) \geq \mathrm{ELdim}(\mathcal{H}[(x,-1)],k)$. Taking the maximum over $x \in \mathcal{X}$ proves the lemma. □

7.9 Proofs from Section 7.3

We first provide some auxiliary lemmas regarding properties of extended mistake trees and extended Littlestone's dimension. This will serve as the basis of the proof of Lemma 7.2.

We state a property about subtrees of a (k,m)-difficult extended mistake tree.

Lemma 7.10 (Recursive Property of Extended Mistake Trees)**.** *Suppose we are given hypothesis class $\mathcal{H}$ that has an extended mistake tree T with root x, left subtree T_{-1}, right subtree T_{+1} and integers $k \geq 0, m \geq 1$. For the root node x, denote by e_l its downward left solid edge, e_r its downward right solid edge, and e_d its downward dashed edge. Denote by y the label of e_d.*

(i) The following statements are equivalent: (a) T is $(0,m)$-difficult. (b) T_y is $(0,m-1)$-difficult.

(ii) For $k \geq 1$, the following statements are equivalent: (a) T is (k,m)-difficult. (b) T_{-y} is $(k-1,m-1)$-difficult, and T_y is $(k,m-1)$-difficult.

Proof of Lemma 7.10. Without loss of generality, suppose $y = +1$. The case of $y = -1$ can be shown symmetrically.

Proof of item (i): We show the implication in both directions.

($\Rightarrow$) Consider a root to leaf path p in T_{+1} that uses no solid edges. Now consider path p_+, the result of prepending the root node x and the downward dashed edge from root x to its right child onto p, i.e. $p_+ = xe_dp$. It can be seen that p_+ uses no solid edges, and $l(p_+) = l(p)+1$. Since T is $(0,m)$-difficult, $l(p_+) \geq m$, therefore $l(p) \geq m-1$, thus showing T_{+1} is $(0,m-1)$-difficult.

($\Leftarrow$) Consider a root to leaf path $p = v_1e_1v_2e_2 \ldots v_ne_nv_{n+1}$ in T that uses no solid edges. The first edge of p must be the downward dashed edge e_d. Define path p_- as the result of deleting the first node $v_1 = x$ and the first edge e_1 from p, i.e. $p_- = v_2e_2 \ldots v_ne_nv_{n+1}$. Since T_{-1} is $(0,m-1)$-difficult, we get that $l(p_-) \geq m-1$. Therefore $l(p) = l(p_-)+1 \geq m$. Therefore, any path p in T that uses no solid edges must be of length at least m. Thus, T is $(0,m)$-difficult.

Proof of item (ii): We show the implication in both directions.

($\Rightarrow$) (1) Consider a root to leaf path p in T_{-1} that uses at most $k-1$ solid edges. Now consider path p_+, the result of prepending the root node x and the downward edge from root x to its left child onto p, i.e. $p_+ = xe_lp$. It can be seen that p_+ uses at most k solid edges, and $l(p_+) = l(p)+1$. Since T is (k,m)-difficult, $l(p_+) \geq m$, therefore $l(p) \geq m-1$, thus showing T_{-1} is $(k-1,m-1)$-difficult.

(2) Consider a root to leaf path p in T_{+1} that uses at most k solid edges. Now consider path p_+, the result of prepending the root node x and the downward dashed edge from root x to its right child onto p, i.e. $p_+ = xe_dp$. It can be seen that p_+ uses at most k solid edges, and $l(p_+) = l(p)+1$. Since T is (k,m)-difficult, $l(p_+) \geq m$, therefore $l(p) \geq m-1$, thus showing T_{+1} is $(k,m-1)$-difficult.

($\Leftarrow$) Consider a root to leaf path $p = v_1e_1v_2e_2 \ldots v_ne_nv_{n+1}$ in T that uses at most k solid edges. Define path p_- as the result of deleting the first node $v_1 = x$ the first edge e_1 from p, i.e. $p_- = v_2e_2 \ldots v_ne_nv_{n+1}$.

(1) If the first edge of p is a downward edge from root x to its left child, then p_- is a root to leaf path in T_{-1}, and uses at most $k-1$ solid edges, since the first edge e_1 has to be a solid edge. Since T_{+1} is $(k-1,m-1)$-difficult, we get that $l(p_-) \geq m-1$.

(2) If the first edge of p is the downward edge from root x to its right child, then p_- is a root to leaf path in T_{+1}, and uses at most k solid edges. Since T_{-1} is $(k,m-1)$-difficult, we get that $l(p_-) \geq m-1$.

In both cases, $l(p_-) \geq m-1$. Hence $l(p) = l(p_-)+1 \geq m$. In summary, any path p in T that uses at most k solid edges must be of length m. Thus, T is (k,m)-difficult.

□

Built upon Lemma 7.10, we obtain the following result regarding $\mathcal{H}$'s extended Littlestone's dimension.

Lemma 7.11 (Recursive Property of ELdim)**.** *Suppose we are given hypothesis class $\mathcal{H}$ and integers $k \geq 0, m \geq 1$.*

(i) The following statements are equivalent: (a) $\text{ELdim}(\mathcal{H},0)$ *is at least m. (b) There exists (x,y) such that x is in* $\text{Dis}(\mathcal{H})$*, and* $\text{ELdim}(\mathcal{H}[(x,y)],0)$ *is at least $m-1$.*

(ii) If $k \geq 1$, the following statements are equivalent: (a) $\text{ELdim}(\mathcal{H},k)$ *is at least m. (b) There exists (x,y) such that both* $\text{ELdim}(\mathcal{H}[(x,y)],k)$ *and* $\text{ELdim}(\mathcal{H}[(x,-y)],k-1)$ *are at least $m-1$.*

Proof of Lemma 7.11.

Proof of item (i): We show the implication in both directions.

($\Rightarrow$) Suppose $\text{ELdim}(\mathcal{H},0) \geq m$. Then $\mathcal{H}$ has a $(0,m)$-difficult mistake tree T. Let x be the root of T, and $y \in \{-1,+1\}$ be the label of the root's downward dashed edge. Since T is a full binary tree, there must be leaves in both the left subtree and the right subtree of the root, i.e. there exist h_1, h_2 in $\mathcal{H}$, $h_1(x) = -1$ and $h_2(x) = -1$. Thus, x is in $\text{Dis}(\mathcal{H})$. By Lemma 7.10, T_y is a $(0,m-1)$-difficult extended mistake tree with respect to $\mathcal{H}[(x,y)]$. The result follows.

($\Leftarrow$) Suppose there exists an example (x,y) such that $x \in \text{Dis}(\mathcal{H})$ and $\text{ELdim}(\mathcal{H}[(x,y)],0) \geq m-1$. Then, $\mathcal{H}[(x,y)]$ has a $(0,m-1)$-difficult extended mistake tree T_y and $\mathcal{H}[(x,-y)]$ has a

zeroth order mistake tree T_{-y}. Construct a new tree T, where its root is x, and its subtrees are T_y and T_{-y} respectively. The dashed downward edge is connected to the subtree T_y. By Lemma 7.10, T is a $(0,m)$-difficult extended mistake tree with respect to $\mathcal{H}$. The result follows.

Proof of item (ii): We show the implication in both directions.

($\Rightarrow$) Suppose $\text{ELdim}(\mathcal{H},k) \geq m$. Then $\mathcal{H}$ has a (k,m)-difficult mistake tree T. Let x be the root of T, and $y \in \{-1,+1\}$ be the root's downward dashed edge label. By Lemma 7.10, T_{-y} is a $(k-1,m-1)$-difficult extended mistake tree with respect to $\mathcal{H}[(x,-y)]$, and T_y is a $(k,m-1)$-difficult extended mistake tree with respect to $\mathcal{H}[(x,y)]$. The result follows.

($\Leftarrow$) Suppose there exists an example (x,y) such that both $\text{ELdim}(\mathcal{H}[(x,y)],k) \geq m-1$ and $\text{ELdim}(\mathcal{H}[(x,-y)],k-1) \geq m-1$. Then, $\mathcal{H}[(x,y)]$ has a $(k,m-1)$-difficult extended mistake tree T_y and $\mathcal{H}[(x,-y)]$ has a $(k-1,m-1)$-difficult extended mistake tree T_{-y}. Now construct a new tree T, where its root is x, and its subtrees are T_y and T_{-y} respectively. The dashed downward edge is connected to the subtree T_y. By Lemma 7.10, T is a (k,m)-difficult extended mistake tree with respect to $\mathcal{H}$. The result follows.

□

Proof of Lemma 7.1. Since $\text{ELdim}(\mathcal{H},k) \geq m$, there is a (k,m)-difficult extended mistake tree $T_{\mathcal{H}}$ with respect to $\mathcal{H}$. We consider the the strategy of the adversary associated with $T_{\mathcal{H}}$. Now consider any deterministic learning algorithm $\mathcal{A}$ that guarantees at most k mistakes. Since $\mathcal{A}$ is deterministic, the interaction between $\mathcal{A}$ and the adversary follows some path p from root to leaf. The number of mistakes is equal to the number of solid edges in p, and the number of abstentions is equal to the number of dashed edges in the p. Since $\mathcal{A}$ guarantees k mistakes, p must contain at most k solid edges, thus it must be of length at least m, as $T_{\mathcal{H}}$ is (k,m)-difficult. Therefore, the number of nontrivial rounds of $\mathcal{A}$ is at least m. □

Proof of Lemma 7.2. We prove the lemma by joint induction on (k,m). **Base Case.** Consider pairs (k,m), where $k=0$ or $m=0$.

(1) For $m=0$ and $k \geq 0$, if there is no $(k,1)$-difficult extended mistake tree, then for all $x \in \mathcal{X}$, V predicts unanimously on x. Otherwise, there are two hypotheses h_1 and h_2 and an example x such that $h_1(x)=-1$ and $h_2(x)=+1$. Consider extended mistake tree T as follows. T has

x as its root, and h_1 and h_2 are leaves directly connecting to the root, where h_1 is on the left and h_2 is on the right. The downward dashed edge is connected to the right, i.e. has label $+1$. It can be seen that T is $(k,1)$-difficult for any $k \geq 0$. Therefore, Algorithm 7.2 always predicts correctly, and there will be no nontrivial rounds subsequently.

(2) For $k = 0$ and $m \geq 0$, we show the result by induction on m. The base case $m = 0$ has been shown in (1). For the inductive case, assume the inductive hypothesis holds for $m' \leq m-1$. Now, given a hypothesis class V such that $\mathrm{ELdim}(V,0)$ is at most m. Consider the first nontrivial round t when running Algorithm 7.2 with version space V. The example x_t must be in $\mathrm{Dis}(V)$, and the algorithm outputs $\hat{y}_t = \perp$. We claim that the resulting version space $V[(x_t, y_t)]$ is such that $\mathrm{ELdim}(V[(x_t,y_t)],0) \leq m-1$. Indeed, suppose $\mathrm{ELdim}(V[(x_t,y_t)],0) \geq m$, then by Lemma 7.11, $\mathrm{ELdim}(V,0) \geq m+1$, which is a contradiction.

Note that from time $t+1$ on, the adversary is only allowed to show $V[(x_t,y_t)]$-realizable sequences. By inductive hypothesis, Algorithm 7.2 runs on $V[(x_t,y_t)]$ and achieves $(0,m-1)$-SZB bound from time $t+1$ on. Therefore, Algorithm 7.2 achieves $(0,m)$-SZB bound throughout the process.

Inductive Case. Consider pairs (k,m) where $k \geq 1$ and $m \geq 1$. Assume for all $k' \leq k$, $m' \leq m$ and $k'+m' \leq k+m-1$, the inductive hypothesis holds. Now, consider a hypothesis class V such that $\mathrm{ELdim}(V,k) \leq m$. Consider the first nontrivial round t when we run Algorithm 7.2 on V. The example x_t must be in $\mathrm{Dis}(V)$. According to Algorithm 7.2's prediction $\hat{y}_t$, we consider three cases separately,

Case 1: $\hat{y}_t = -1$. In this case, since round t is nontrivial, $y_t = -\hat{y}_t = +1$. We claim that $\mathrm{ELdim}(V[(x_t,+1)],k-1) \leq m-1$. Indeed, assume (for the sake of contradiction) that $m_{-1} \geq \mathrm{ELdim}(V[(x_t,+1)],k-1) \geq m$. By definition of Algorithm 7.2, $\mathrm{ELdim}(V[(x_t,-1)],k-1) = m_{+1} \geq m_{-1} \geq m$. Hence, for any $y \in \{-1,+1\}$, $\mathrm{ELdim}(V[(x_t,y)],k-1) \geq m$.

Also by definition of Algorithm 7.2, $\max(\mathrm{ELdim}(V[(x_t,+1)],k), \mathrm{ELdim}(V[(x_t,-1)],k)) = m_\perp \geq m$. Thus, there exists some $\hat{y} \in \{-1,+1\}$ such that

$$\mathrm{ELdim}(V[(x_t,\hat{y})],k) \geq m$$

Therefore, for $\hat{y}$, we have $\mathrm{ELdim}(V[(x_t,\hat{y})],k) \geq m$ and $\mathrm{ELdim}(V[(x_t,-\hat{y})],k-1) \geq m$. By Lemma 7.11, $\mathrm{ELdim}(V,k) \geq m+1$, which is a contradiction.

Note that from time $t+1$ on, the adversary is only allowed to show $V[(x_t, y_t)]$-realizable sequences. By inductive hypothesis, Algorithm 7.2 runs on $V[(x_t, y_t)]$ with mistake budget $k-1$ and achieves $(k-1, m-1)$-SZB bound from round $t+1$ on. Therefore, Algorithm 7.2 achieves (k,m)-SZB bound throughout the process.

Case 2: $\hat{y}_t = +1$**.** This case is symmetric to Case 1.

Case 3: $\hat{y}_t = \perp$**.** We first claim that $\mathrm{ELdim}(V[(x_t, -1)], k) \leq m-1$. Indeed, assume (for the sake of contradiction) that $\mathrm{ELdim}(V[(x_t, -1)], k) \geq m$. By definition of Algorithm 7.2, $m_{-1} \geq m_{\perp} \geq m$, that is

$$\mathrm{ELdim}(V[(x_t, +1)], k-1) \geq m$$

By Lemma 7.11, $\mathrm{ELdim}(V, k) \geq m+1$, contradiction. Symmetrically, we can also deduce that $\mathrm{ELdim}(V[(x_t, +1)], k)$ is at most $m-1$.

Hence, irrespective of the outcome $y_t \in \{-1, +1\}$, the resulting version space $V[(x_t, y_t)]$ satisfies that $\mathrm{ELdim}(V[(x_t, y_t)], k) \leq m-1$. Note that from time $t+1$ on, the adversary is only allowed to show $V[(x_t, y_t)]$-realizable sequences. By inductive hypothesis, Algorithm 7.2 runs on $V[(x_t, y_t)]$ with mistake budget k, and achieves $(k, m-1)$-SZB bound from round $t+1$ on. Therefore, Algorithm 7.2 achieves (k,m)-SZB bound throughout the process.

In summary, Algorithm 7.2, when run on V, achieves (k,m)-SZB bound. This completes the induction. □

Proof of Theorem 7.2. (a) This is an immediate consequence of Lemma 7.2.

(b) By Lemma 7.1, there is a strategy of the adversary such that any deterministic learner guaranteeing at most k mistakes must have at least m nontrivial rounds. Therefore, no deterministic learner can achieve a $(k, m-1)$-SZB bound.

□

Proof of Theorem 7.3. Recall that $\mathrm{Ldim}(\mathcal{H}) = d < \infty$. We show the equality by showing inequalities in both sides.

(1) We first show $\mathrm{ELdim}(\mathcal{H}, d) \leq d$. Indeed, SOA is guaranteed to make at most d mistakes and no abstentions for $\mathcal{H}$-realizable sequences. This has a total of at most d nontrivial rounds. Now, by Lemma 7.1, if $\mathrm{ELdim}(\mathcal{H}, d) \geq d+1$, SOA must have at least $d+1$ nontrivial rounds, contradiction.

(2) On the other hand, since $\mathrm{Ldim}(\mathcal{H}) = d$, there is a depth-$d$ mistake tree T with respect to $\mathcal{H}$. Consider the following modification of T: for each internal node, add a dashed downward edge to its right child. It can be seen that the resulting tree, $\tilde{T}$, is a (d,d)-difficult extended mistake tree. Therefore $\mathrm{ELdim}(\mathcal{H},d) \geq d$.

In summary, $\mathrm{ELdim}(\mathcal{H},d) = d$. □

7.10 Proofs from Section 7.4

Proof of Lemma 7.3. For any depth-t tree $\mathbf{x}$, note that

$$|S(\mathcal{H},\mathbf{x})| \leq |\mathcal{H}|$$

Therefore,

$$\mathcal{S}(\mathcal{H},t) = \max_{\mathbf{x}} |S(\mathcal{H},\mathbf{x})| \leq |\mathcal{H}|.$$ □

Lemma 7.12 (Recursive Formula)**.** *For a hypothesis class $\mathcal{H}$ and $t \geq 1$, we have*

$$\mathcal{S}(\mathcal{H},t) = \max_{x\in\mathcal{X}} (\mathcal{S}(\mathcal{H}[(x,-1)],t-1) + \mathcal{S}(\mathcal{H}[(x,+1)],t-1))$$

We need the following notation of subtrees to give the proof of Lemma 7.12.

Definition 7.9 (Subtrees, see [RST10])**.** *Given a depth-t tree $\mathbf{x}$, the left subtree $\mathbf{x}^l$ of $\mathbf{x}$ at the root is defined as $t-1$ mappings $(\mathbf{x}_1^l,\ldots,\mathbf{x}_{t-1}^l)$, where $\mathbf{x}_i^l(\epsilon) = \mathbf{x}(\{-1\}\times\epsilon)$, for $\epsilon \in \{\pm 1\}^{t-1}$. The right subtree $\mathbf{x}^r$ of $\mathbf{x}$ at the root is defined as $t-1$ mappings $(\mathbf{x}_1^r,\ldots,\mathbf{x}_{t-1}^r)$, where $\mathbf{x}_i^r(\epsilon) = \mathbf{x}(\{+1\}\times\epsilon)$, for $\epsilon \in \{\pm 1\}^{t-1}$.*

Proof of Lemma 7.12. Consider the definition of $\mathcal{S}(\mathcal{H},t)$:

$$\max_{\mathbf{x}} |\left\{(\epsilon_1,\epsilon_2,\ldots,\epsilon_t) \in \{\pm 1\}^t : \epsilon_1 = h(\mathbf{x}_1(\epsilon)), \epsilon_2 = h(\mathbf{x}_2(\epsilon)),\ldots,\epsilon_t = h(\mathbf{x}_t(\epsilon)), h \in \mathcal{H}\right\}|$$

This can be alternatively written as

$$\max_{\mathbf{x}} |\Big\{(-1,\sigma_1,\ldots,\sigma_{t-1}) \in \{\pm 1\}^{t-1} : \sigma_1 = h(\mathbf{x}_1^l(\sigma)),\ldots,\sigma_t = h(\mathbf{x}_{t-1}^l(\sigma)), h \in \mathcal{H}[(\mathbf{x}_1,-1)]\Big\}$$
$$\cup \Big\{(+1,\sigma_1,\ldots,\sigma_{t-1}) \in \{\pm 1\}^{t-1} : \sigma_2 = h(\mathbf{x}_1^r(\sigma)),\ldots,\sigma_t = h(\mathbf{x}_{t-1}^r(\sigma)), h \in \mathcal{H}[(\mathbf{x}_1,+1)]\Big\}|$$

The above is equal to $\max_{\mathbf{x}_1 \in \mathcal{X}} F(\mathbf{x}_1)$, where $F(\mathbf{x}_1)$ equals

$$\max_{\mathbf{x}^l} |\Big\{(-1,\sigma_1,\ldots,\sigma_{t-1}) \in \{\pm 1\}^{t-1} : \sigma_1 = h(\mathbf{x}_1^l(\sigma)),\ldots,\sigma_{t-1} = h(\mathbf{x}_{t-1}^l(\sigma)), h \in \mathcal{H}[(\mathbf{x}_1,-1)]\Big\}|$$

plus

$$\max_{\mathbf{x}^r} |\Big\{(+1,\sigma_1,\ldots,\sigma_{t-1}) \in \{\pm 1\}^{t-1} : \sigma_1 = h(\mathbf{x}_1^l(\sigma)),\ldots,\sigma_{t-1} = h(\mathbf{x}_{t-1}^l(\sigma)), h \in \mathcal{H}[(\mathbf{x}_1,+1)]\Big\}|$$

Note that the above is precisely $\mathcal{S}(\mathcal{H}[(x_1,-1)],t-1)+\mathcal{S}(\mathcal{H}[(x_1,+1)],t-1)$. The lemma follows.

□

Now we are ready to prove Lemma 7.4.

Proof of Lemma 7.4. We prove the result by joint induction on (t,d). **Base Case:** Consider $t=0$ or $d=0$. If $t=0$, then $\mathcal{S}(\mathcal{H},0) \le 1 = \binom{0}{\le d}$. If $d=0$, then $\mathcal{S}(\mathcal{H},t) \le 1 = \binom{t}{\le 0}$.

Inductive Case: For $t \ge 1$ and $d \ge 1$, assume the result holds for (t',d') such that $t' \le t$, $d' \le d$ and $t'+d' \le t+d-1$. First by Lemma 7.12, for some x in $\mathcal{X}$, $\mathcal{S}(\mathcal{H},t) \le \mathcal{S}(\mathcal{H}[(x,-1)],t-1)+\mathcal{S}(\mathcal{H}[(x,+1)],t-1)$.

Second, Since $\mathrm{Ldim}(\mathcal{H}) = d$, for x, there exists $y \in \{-1,+1\}$ such that $\mathrm{Ldim}(\mathcal{H}[(x,y)]) \le d-1$ and $\mathrm{Ldim}(\mathcal{H}[(x,-y)])) \le d$. Hence by inductive hypothesis, there exists $y \in \{-1,+1\}$ such that $\mathcal{S}(\mathcal{H}[(x,y)],t-1) \le \binom{t-1}{\le d-1}$ and $\mathcal{S}(\mathcal{H}[(x,-y)],t-1) \le \binom{t-1}{\le d}$. Therefore

$$\mathcal{S}(\mathcal{H},t) \le \mathcal{S}(\mathcal{H}[(x,-1)],t-1)+\mathcal{S}(\mathcal{H}[(x,+1)],t-1) \le \binom{t-1}{\le d-1} + \binom{t-1}{\le d} \le \binom{t}{\le d}$$

This completes the induction. □

Proof of Theorem 7.4. For any integer m, if $m \le \mathrm{ELdim}(\mathcal{H},k)$, then by Lemma 7.13,

$$\mathcal{S}(\mathcal{H},m) \ge \binom{m}{\le k+1}$$

This implies that

$$m \leq \sup\left\{t : \binom{t}{\leq k+1} \leq \mathcal{S}(\mathcal{H},t)\right\}$$

Taking $m = \mathrm{ELdim}(\mathcal{H},k)$, we get the theorem. □

Lemma 7.13. *Suppose k,t are nonnegative integers. If $\mathrm{ELdim}(\mathcal{H},k) \geq t$, then $\mathcal{S}(\mathcal{H},t) \geq \binom{t}{\leq k+1}$.*

Proof of Lemma 7.13. By joint induction on (k,t). **Base Case:** We consider (k,t) pairs where $k = 0$ or $t = 0$.

(1) For $t = 0$, $\mathrm{ELdim}(\mathcal{H},k) \geq 0$ implies that $\mathcal{H}$ is nonempty. Thus, $\mathcal{S}(\mathcal{H},0) = 1 \geq \binom{0}{\leq k+1}$.

(2) For $k = 0$, we prove the result by induction on t. The case of $t = 0$ has been shown in (1). For the inductive case, by Lemma 7.11, there exists (x,y) such that $x \in \mathrm{Dis}(\mathcal{H})$ and $\mathrm{ELdim}(\mathcal{H}[(x,y)],0) \geq t-1$. Thus, by inductive hypothesis, $\mathcal{S}(\mathcal{H}[(x,y)],t-1) \geq t$. Also, since $\mathcal{H}[(x,-y)]$ is nonempty, we get $\mathcal{S}(\mathcal{H}[(x,-y)],t-1) \geq 1$. Thus,

$$\mathcal{S}(\mathcal{H},t) \geq \mathcal{S}(\mathcal{H}[(x,y)],t-1) + \mathcal{S}(\mathcal{H}[(x,-y)],t-1) \geq t+1$$

This completes the proof for $k = 0$.

Inductive Case: For $t \geq 1$ and $k \geq 1$, suppose the inductive hypothesis holds for any (k',t') such that $k' \leq k$, $t' \leq t$, $k'+t' \leq k+t+1$.

Now suppose $\mathrm{ELdim}(\mathcal{H},k) \geq t$. By Lemma 7.11, there exists (x,y) such that the following hold simultaneously: $\mathrm{ELdim}(\mathcal{H}[(x,y)],k) \geq t-1$ and $\mathrm{ELdim}(\mathcal{H}[(x,-y)],k-1) \geq t-1$. Thus by inductive hypothesis, $\mathcal{S}(\mathcal{H}[(x,y)],t-1) \geq \binom{t-1}{\leq k+1}$ and $\mathcal{S}(\mathcal{H}[(x,-y)],t-1) \geq \binom{t-1}{\leq k}$. Therefore,

$$\mathcal{S}(\mathcal{H},t) \geq \mathcal{S}(\mathcal{H}[(x,y)],t-1) + \mathcal{S}(\mathcal{H}[(x,-y)],t-1) \geq \binom{t}{\leq k+1}$$

This completes the induction. □

Proof of Theorem 7.5. Note that $\mathcal{S}(\mathcal{H},t) \leq |\mathcal{H}| = n$, therefore by Lemma 7.13, $\mathrm{ELdim}(\mathcal{H},k) \leq \max\left\{t : \binom{t}{\leq k+1} \leq n\right\}$.

On the other hand, Lemma 7.14 implies that for all m such that $\binom{m}{\leq k+1} \leq n$, there is a

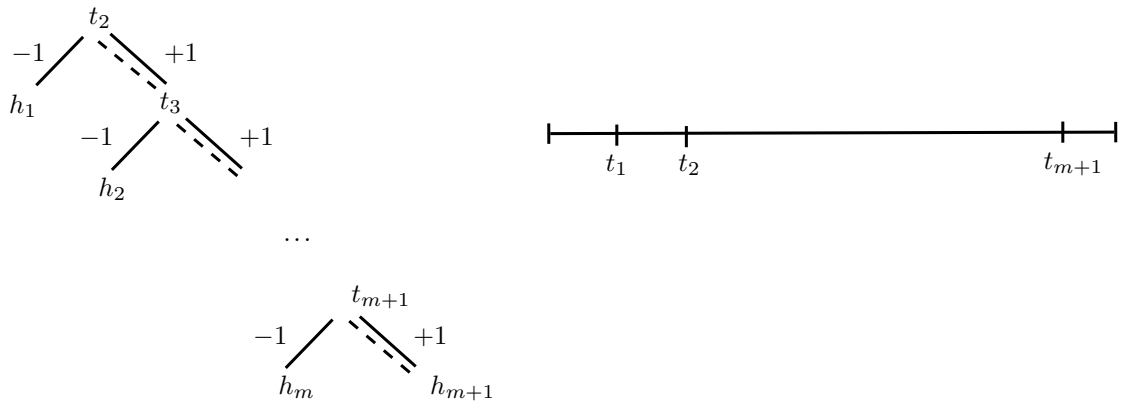

Figure 7.5: Construction of $T_{0,m}$, an extended mistake tree given parameters $k = 0$ and $m \geq 0$. For each i, h_i is defined as $h_i(x) := 2I(x \leq t_i) - 1$.

(m,k)-difficult extended mistake tree with respect to $\mathcal{H}$. Hence,

$$\text{ELdim}(\mathcal{H},k) \geq \max\left\{t : \binom{t}{\leq k+1} \leq n\right\}$$

Combining the lower and upper bound, we get the theorem. □

Lemma 7.14. *Consider the set of threshold classifiers* $\mathcal{H} = \{2I(x \leq t) - 1 : t \in \{t_1, \ldots, t_n\}\}$. *If integers* $k \geq 0$ *and* $m \geq 0$ *are such that* $\binom{m}{k+1} \leq n$, *then* $\mathcal{H}$ *has a* (k,m)*-difficult mistake tree.*

Proof. We prove the lemma by joint induction on (k,m).

Base Case: Consider $k = 0$ or $m = 0$.

(1) For $k = 0$, $\binom{m}{\leq k} = m+1$. We show a construction of $T_{0,m}$, a $(0,m)$-difficult extended mistake tree in Figure 7.5. It can be seen that the resulting tree $T_{0,m}$ is $(0,m)$-difficult, as the only root to leaf path using no solid edges corresponds to examples $(t_2, +1)$, …, $(t_{m+1}, +1)$, which has length m.

(2) For $m = 0$ and integer k, $\binom{m}{\leq k} = 1$. The zeroth order extended mistake tree containing h_{t_1} is a $(k,0)$-difficult extended mistake tree.

Inductive Case: For $k \geq 1$ and $m \geq 1$, assume the inductive hypothesis holds for (k',m') such that $k' \leq k$, $m' \leq m$ and $k' + m' \leq k + m - 1$.

We now construct $T_{k,m}$, a (k,m)-difficult extended mistake tree, using hypotheses in $\mathcal{H}$. Let $r_- = \binom{m-1}{\leq k}$, $r_+ = \binom{m-1}{\leq k+1}$. Consider hypothesis class $\mathcal{H}_- = \left\{2I(x \leq t) - 1 : t \in \left\{t_1, \ldots, t_{r_-}\right\}\right\}$

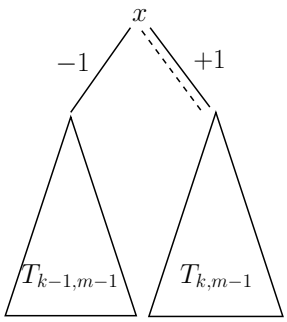

Figure 7.6: Construction of $T_{k,m}$, an extended mistake tree given parameters $k \geq 1$ and $m \geq 1$, from $T_{k,m-1}$ and $T_{k-1,m-1}$.

and $\mathcal{H}_+ = \left\{ 2I(x \leq t) - 1 : t \in \left\{ t_{r_-+1}, \ldots, t_{r_-+r_+} \right\} \right\}$. Note that $r_- + r_+ \leq \binom{m-1}{\leq k} + \binom{m-1}{\leq k-1} \leq \binom{m}{\leq k} \leq n$, thus $\mathcal{H}_-$ and $\mathcal{H}_+$ are well defined.

Since $|\mathcal{H}_-| \geq \binom{m-1}{\leq k}$, by inductive hypothesis, there is a $(k-1, m-1)$ difficult extended mistake tree $T_{k-1,m-1}$ with respect to $\mathcal{H}_-$. Similarly, since $|\mathcal{H}_+| \geq \binom{m-1}{\leq k+1}$, by inductive hypothesis, there is a $(k, m-1)$ difficult extended mistake tree $T_{k,m-1}$ with respect to $\mathcal{H}_+$.

Now Let x be a real number in (t_{r_-}, t_{r_-+1}), it can be seen that all hypotheses in $\mathcal{H}_-$ classifies x as -1 and all hypothesis in $\mathcal{H}_+$ classifies x as $+1$. We construct $T_{k,m}$ as in Figure 7.6, where x is at the root, and its downward left solid edge connect to $T_{k-1,m-1}$; its downward right solid edge and downward dashed edge connects to $T_{k,m-1}$. Note that $T_{k,m}$ is a valid extended mistake tree, since all hypotheses at the leaves in $T_{k-1,m-1}$ (resp. $T_{k-1,m}$) classifies x as -1 (resp. $+1$). By Lemma 7.10, $T_{k,m}$ is (k, m)-difficult. □

Lemma 7.15. *Let $\mathcal{C}^l$ be the class of unions of at most l singletons. Then*

$$\mathcal{S}(\mathcal{C}^l, t) = \binom{t}{\leq l}$$

Proof. We show the equality by showing the inequality in both directions.

(1) $S(\mathcal{C}^l, t) \leq \binom{t}{\leq l}$ From Lemma 7.4.

(2) Consider a $\mathcal{X}$-valued tree $\mathbf{x}$ with all its elements distinct. Then, consider the set

$$S(\mathcal{C}^l, \mathbf{x}) = \left\{ (\epsilon_1, \ldots, \epsilon_t) : \exists h \in \mathcal{C}^l, h(\mathbf{x}_s(\epsilon)) = \epsilon_s, s = 1, 2, \ldots, t \right\}$$

We claim that $S(\mathcal{C}^l, \mathbf{x})$ contains $\left\{\epsilon = (\epsilon_1, \ldots, \epsilon_t) : |\{s : \epsilon_s = -1\}| \leq l\right\}$. Indeed, for any element in $\left\{\epsilon = (\epsilon_1, \ldots, \epsilon_t) : |\{s : \epsilon_s = -1\}| \leq l\right\}$, the hypothesis $h = 1 - 2I(x \in \left\{\mathbf{x}_s(\epsilon) : \epsilon_s = -1\right\}) \in \mathcal{C}^l$ satisfies that $h(\mathbf{x}_s(\epsilon)) = \epsilon_s$, for $s = 1, 2, \ldots, t$. Hence $S(\mathcal{C}^l, \mathbf{x}) \geq \binom{t}{\leq l}$, implying $\mathcal{S}(\mathcal{C}^l, t) \geq \binom{t}{\leq l}$.

In summary, $S(\mathcal{C}^l, t) = \binom{t}{\leq l}$. □

Proof of Theorem 7.6. We show the equality by showing the inequality in both directions.

(1) Consider the case that $k \leq l - 1$. By Lemma 7.16, for any integer m, there is a (k, m)-difficult extended mistake tree with respect to $\mathcal{C}^l$. Thus, $\text{ELdim}(\mathcal{C}^l, k) = \infty$.

(2) Consider the case that $k \geq l$. By Lemma 7.15, $\mathcal{S}(\mathcal{C}^l, t) = \binom{t}{\leq l}$. By Theorem 7.4,

$$\text{ELdim}(\mathcal{C}^l, k) \leq \max\left\{t : \binom{t}{\leq k+1} \leq \binom{t}{\leq l}\right\} = l$$

This gives that $\text{ELdim}(\mathcal{C}^l, k) \leq l$.

On the other hand, $\mathcal{C}^l$ has a mistake tree T of depth l. Consider the following modification of T: for each internal node, add a dashed downward edge to its right child. It can be seen that the resulting tree, $\tilde{T}$, is an (l, l)-difficult extended mistake tree. Therefore T' is also a (k, l)-difficult mistake tree, which gives that $\text{ELdim}(\mathcal{C}^l, k) \geq l$.

Hence, we conclude that $\text{ELdim}(\mathcal{C}^l, k) = l$. □

Recall that $\mathcal{C}^l$ is the class of union of at most l singletons in instance domain $\mathcal{X}$. That is, hypotheses that take value $+1$ on $\mathcal{X}$, except for at most l points.

Lemma 7.16. *Suppose we are given an infinite domain $\mathcal{X}$ and an integer $l \geq 1$. Then for any integer $m \geq 0$, there exists a $(l-1, m)$-difficult extended mistake tree with respect to hypothesis class $\mathcal{C}^l$, such that all its dashed edges are labeled $+1$.*

Proof. By induction on l.

Base Case: For $l = 1$, the construction of the required extended mistake tree with respect to $\mathcal{C}^1$ is given in Figure 7.7. Note that the tree is $(0, m)$-difficult, and all its dashed edges are labeled $+1$.

Inductive Case: Suppose the inductive hypothesis holds for any $l' \leq l - 1$. Now pick an arbitrary $x \in \mathcal{X}$. Fix integer m. Consider $(\mathcal{X}_1, \mathcal{X}_2)$, a partition of $\mathcal{X} \setminus \{x\}$, where both $|\mathcal{X}_1|$ and $|\mathcal{X}_2|$ are infinite.

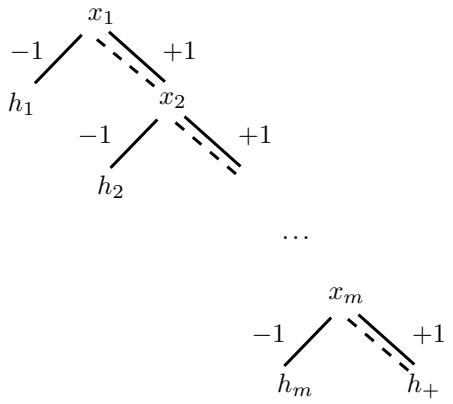

Figure 7.7: A $(0, m)$-difficult extended mistake tree with respect to $\mathcal{C}^1$. $x_1, \ldots, x_m$ are distinct elements in $\mathcal{X}$, and for each i, h_i is defined as $h_i(x) := 1 - 2I(x = x_i)$. h_+ is the constant function $+1$.

By inductive hypothesis, there is a $(l-1, m)$-difficult extended mistake tree T_+ with respect to $\mathcal{C}^l$ on domain $\mathcal{X}_1$, such that all its dashed edges are labeled $+1$. Since for any $h \in \mathcal{C}^{l-1}$, there exists $h' \in \mathcal{C}^l[(x, +1)]$ such that $h \equiv h'$ on $\mathcal{X}_1$, we can modify T_+'s leaves such that they all correspond to hypotheses in $\mathcal{C}^l[(x, +1)]$, getting a new extended mistake tree $\tilde{T}_+$.

Similarly, by inductive hypothesis, there is a $(l-2, m)$-difficult extended mistake tree T_- with respect to $\mathcal{C}^{l-1}$ on domain $\mathcal{X}_2$, such that all its dashed edges are labeled $+1$. Since for any $h \in \mathcal{C}^l$, there exists $h' \in \mathcal{C}^l[(x, -1)]$ such that $h \equiv h'$ on $\mathcal{X}_2$, we can modify T_-'s leaves such that they all correspond to hypotheses in $\mathcal{C}^l[(x, -1)]$, getting a new extended mistake tree $\tilde{T}_-$.

Now consider the extended mistake tree T rooted at x, with its left subtree as $\tilde{T}_-$ and right subtree as $\tilde{T}_+$. The dashed downward edge of root is linked to its right child, i.e. has label $+1$. Note that T is a valid extended mistake tree, since all hypotheses at the leaves in $\tilde{T}_-$ (resp. $\tilde{T}_+$) classifies x as -1 (resp. $+1$). By Lemma 7.10, T is $(l-1, m+1)$-difficult, hence $(l-1, m)$-difficult. Additionally, all its dashed edges are labeled $+1$. Since the choice of m is arbitrary, this completes the induction. □

7.11 Proofs from Section 7.5

Proof of Theorem 7.7. Let $\mathcal{X}$ be an infinite set. Let $\mathcal{H}$ be the hypothesis class containing only one hypothesis $h \equiv +1$. Note that $\mathcal{H}^l = \mathcal{C}^l$ and by Theorem 7.6, $\text{ELdim}(\mathcal{H}^l, k) = \infty$ for $k < l$. By Lemma 7.1, the theorem follows. □

Proof of Theorem 7.8. Let $\mathcal{X}$ be an infinite set. Let $\mathcal{H}$ be the $\mathcal{C}^d$, the class of unions of at most d singletons. Note that $\mathcal{H}^l = \mathcal{C}^{l+d}$ and by Theorem 7.6, $\text{ELdim}(\mathcal{C}^{l+d}, k) = \infty$ for $k < l+d$. By Lemma 7.1, the theorem follows. □

We will need the following result regarding the tree shattering coefficient of the "product" of two hypothesis classes.

Lemma 7.17. *Suppose $\mathcal{H}_1, \mathcal{H}_2$ are two hypothesis classes. If $\mathcal{H} = \mathcal{H}_1 \cdot \mathcal{H}_2$, then for all integers $t \geq 0$,*

$$\mathcal{S}(\mathcal{H}, t) \leq \mathcal{S}(\mathcal{H}_1, t) \cdot \mathcal{S}(\mathcal{H}_2, t)$$

We first show a basic property of tree shattering coefficient.

Lemma 7.18. *For hypothesis classes $\mathcal{H}_1$, $\mathcal{H}_2$, $\mathcal{S}(\mathcal{H}_1 \cup \mathcal{H}_2, t) \leq \mathcal{S}(\mathcal{H}_1, t) + \mathcal{S}(\mathcal{H}_2, t)$.*

Proof. For any depth-t tree $\mathbf{x}$, we have that

$$S(\mathcal{H}_1 \cup \mathcal{H}_2, \mathbf{x}) \subseteq S(\mathcal{H}_1, \mathbf{x}) \cup S(\mathcal{H}_2, \mathbf{x})$$

Therefore,

$$|S(\mathcal{H}_1 \cup \mathcal{H}_2, \mathbf{x})| \leq |S(\mathcal{H}_1, \mathbf{x})| + |S(\mathcal{H}_2, \mathbf{x})| \leq \mathcal{S}(\mathcal{H}_1, t) + \mathcal{S}(\mathcal{H}_2, t)$$

Since the choice of $\mathbf{x}$ is arbitrary, we get

$$\mathcal{S}(\mathcal{H}_1 \cup \mathcal{H}_2, t) \leq \mathcal{S}(\mathcal{H}_1, t) + \mathcal{S}(\mathcal{H}_2, t). \qquad \square$$

Proof of Lemma 7.17. By induction on t. **Base Case:** Consider $t = 0$. If one of $\mathcal{S}(\mathcal{H}_1, 0)$, $\mathcal{S}(\mathcal{H}_2, 0)$ is 0, this implies $\mathcal{H}_1 = \emptyset$ or $\mathcal{H}_2 = \emptyset$. Therefore, $\mathcal{H} = \emptyset$, the result holds. Otherwise, both $\mathcal{S}(\mathcal{H}_1, 0)$ and $\mathcal{S}(\mathcal{H}_2, 0)$ are at least 1. In this case $\mathcal{H}$ is nonempty, thus $1 = \mathcal{S}(\mathcal{H}, 0) \leq 2^0 = 1$, the result also hold.

Inductive Case: Given $t \geq 1$, assume the inductive hypothesis $\mathcal{S}(\mathcal{F}_1 \cdot \mathcal{F}_2, t-1) \leq \mathcal{S}(\mathcal{F}_1, t-1) \cdot \mathcal{S}(\mathcal{F}_2, t-1)$ holds for any hypothesis classes $\mathcal{F}_1$, $\mathcal{F}_2$. Fix $x \in \mathcal{X}$. Note that $\mathcal{H}[(x, +1)] =$

$(\mathcal{H}_1[(x,+1)]\cdot\mathcal{H}_2[(x,+1)])\cup(\mathcal{H}_1[(x,-1)]\cdot\mathcal{H}_2[(x,-1)])$. Therefore,

$$\begin{aligned}
& \mathcal{S}(\mathcal{H}[(x,+1)],t-1) \\
\leq\ & \mathcal{S}(\mathcal{H}_1[(x,+1)]\cdot\mathcal{H}_2[(x,+1)],t-1)+\mathcal{S}(\mathcal{H}_1[(x,-1)]\cdot\mathcal{H}_2[(x,-1)],t-1) \\
\leq\ & \mathcal{S}(\mathcal{H}_1[(x,+1)],t-1)\mathcal{S}(\mathcal{H}_2[(x,+1)],t-1)+\mathcal{S}(\mathcal{H}_1[(x,-1)],t-1)\mathcal{S}(\mathcal{H}_2[(x,-1)],t-1)
\end{aligned}$$

where the first inequality is Lemma 7.18, the second inequality is by inductive hypothesis.

Likewise, we have

$$\begin{aligned}
& \mathcal{S}(\mathcal{H}[(x,-1)],t-1) \\
\leq\ & \mathcal{S}(\mathcal{H}_1[(x,-1)],t-1)\mathcal{S}(\mathcal{H}_2[(x,+1)],t-1)+\mathcal{S}(\mathcal{H}_1[(x,+1)],t-1)\mathcal{S}(\mathcal{H}_2[(x,-1)],t-1)
\end{aligned}$$

Therefore,

$$\begin{aligned}
& \mathcal{S}(\mathcal{H}[(x,-1)],t-1)+\mathcal{S}(\mathcal{H}[(x,+1)],t-1) \\
\leq\ & (\mathcal{S}(\mathcal{H}_1[(x,-1)],t-1)+\mathcal{S}(\mathcal{H}_1[(x,+1)],t-1))(\mathcal{S}(\mathcal{H}_2[(x,-1)],t-1)+\mathcal{S}(\mathcal{H}_2[(x,+1)],t-1)) \\
\leq\ & \mathcal{S}(\mathcal{H}_1,t)\mathcal{S}(\mathcal{H}_2,t)
\end{aligned}$$

where the second inequality is from Lemma 7.11. Since the choice of x is arbitrary, we get

$$\mathcal{S}(\mathcal{H},t)=\max_{x\in\mathcal{X}}(\mathcal{S}(\mathcal{H}[(x,-1)],t-1)+\mathcal{S}(\mathcal{H}[(x,+1)],t-1))\leq\mathcal{S}(\mathcal{H}_1,t)\mathcal{S}(\mathcal{H}_2,t). \qquad \square$$

Proof of Lemma 7.5. Note that by Lemmas 7.4 and 7.17,

$$\mathcal{S}(\mathcal{H}^l,t)\leq\mathcal{S}(\mathcal{C}^l,t)\cdot\mathcal{S}(\mathcal{H},t)=\binom{t}{\leq l}\cdot\mathcal{S}(\mathcal{H},t)\leq|\mathcal{H}|\binom{t}{\leq l}$$

Hence,

$$\mathrm{ELdim}(\mathcal{H}^l,k)\leq\max\left\{t:\binom{t}{\leq k+1}\leq|\mathcal{H}|\binom{t}{\leq l}\right\}$$

Now, consider any t such that $t\geq 2l$ and

$$\binom{t}{\leq k+1}\leq|\mathcal{H}|\binom{t}{\leq l}$$

Since $\binom{t}{\leq k+1} \geq \binom{t}{k+1} \geq (\frac{t}{k+1})^{k+1}$, and $\binom{t}{\leq l} \leq (\frac{et}{l})^l$ for $t \geq 2l$, we get

$$(\frac{t}{k+1})^{k+1} \leq |\mathcal{H}|(\frac{et}{l})^l$$

Hence,

$$t^{k+1-l} \leq |\mathcal{H}|\frac{(k+1)^{k+1}}{l^l}$$

Since $\frac{(k+1)^{k+1}}{l^l} \leq (e(k+1))^{k+1-l}$, we get

$$t^{k+1-l} \leq |\mathcal{H}|(e(k+1))^{k+1-l}$$

That is, $t \leq e(k+1)|\mathcal{H}|^{\frac{1}{k+1-l}}$.

In summary,

$$\mathrm{ELdim}(\mathcal{H}^l,k) \leq \max(2l, e(k+1)|\mathcal{H}|^{\frac{1}{k+1-l}}) = e(k+1)|\mathcal{H}|^{\frac{1}{k+1-l}}$$

where the equality uses the fact that $k \geq l$. □

Proof of Lemma 7.6. Note that by Lemmas 7.4 and 7.17,

$$\mathcal{S}(\mathcal{H}^l,t) \leq \mathcal{S}(\mathcal{C}^l,t) \cdot \mathcal{S}(\mathcal{H},t) = \binom{t}{\leq l} \cdot \mathcal{S}(\mathcal{H},t) \leq \binom{t}{\leq d}\binom{t}{\leq l}$$

Hence,

$$\mathrm{ELdim}(\mathcal{H}^l,k) \leq \max\left\{t : \binom{t}{\leq k+1} \leq \binom{t}{\leq d}\binom{t}{\leq l}\right\}$$

Now, consider any t such that $t \geq 2l$, $t \geq 2d$ and

$$\binom{t}{\leq k+1} \leq \binom{t}{\leq d}\binom{t}{\leq l}$$

Since $\binom{t}{\leq k+1} \geq \binom{t}{k+1} \geq (\frac{t}{k+1})^{k+1}$, $\binom{t}{\leq d} \leq (\frac{et}{d})^d$ for $t \geq 2d$, and $\binom{t}{\leq l} \leq (\frac{et}{l})^l$ for $t \geq 2l$, we get

$$(\frac{t}{k+1})^{k+1} \leq (\frac{et}{d})^d(\frac{et}{l})^l$$

Hence,

$$t^{k+1-l-d} \leq e^{l+d}\frac{(k+1)^{k+1}}{l^l d^d}$$

Since

$$\begin{aligned} & \frac{(k+1)^{k+1}}{l^l d^l} \\ = \; & (1+\frac{k+1-l}{l})^l (1+\frac{k+1-d}{d})^d (k+1)^{k+1-l-d} \\ \leq \; & e^{2k+2-l-d}(k+1)^{k+1-l-d} \end{aligned}$$

we get

$$t \leq (k+1) \cdot e^{\frac{2k+2}{k+1-l-d}}$$

In summary,

$$\mathrm{ELdim}(\mathcal{H}^l, k) \leq \max(2l, 2d, (k+1) \cdot e^{\frac{2k+2}{k+1-l-d}})) = (k+1) \cdot e^{\frac{2k+2}{k+1-l-d}}$$

where the equality uses the fact that $k \geq l+d$. □

Proof of Theorem 7.9. Let $\mathcal{X}$ be an infinite set. $x_1, x_2, \ldots$ is a sequence of distinct elements from $\mathcal{X}$. Let $\mathcal{H}$ be the hypothesis class containing only one hypothesis $h \equiv +1$. Note that $\mathcal{H}^l = \mathcal{C}^l$. Let $\epsilon = 1 - k/l > 0$, thus $k = l(1-\epsilon)$. Fix integer $m = \lceil \frac{2}{\epsilon}(a+l) + 2l \rceil$. By Lemma 7.16, $\mathcal{C}^l$ has a $(l-1, m)$-difficult extended mistake tree T.

We define the following strategy by the adversary based on T. At time $t = 1$, the adversary chooses the example x_1 corresponding to the root of T, and shows it to the learner. If $p_{1,+} > 1 - \epsilon$, then it reveals label $y_1 = -1$ and follows the downward solid edge labeled -1 to reach the left child of the root; otherwise it reveals label $y_1 = +1$ and follows the downward dashed edge labeled $+1$ to reach the right child of the root. At time $t \geq 2$, suppose the adversary reaches a node with example x_t, then x_t is shown to the learner, and one of the downward edges adjacent to this node is followed by the same rule. The interaction comes to an end when a leaf is reached. It can be seen that the realizability assumption is maintained.

Consider an Algorithm $\mathcal{A}$ that guarantees a cumulative mistake penalty at most k.

(1) We claim that the interaction between the learner and the adversary lasts for at least m rounds. To see this, note that $\mathcal{A}$ predicts at most $l-1$ times such that $p_{t,+} > 1 - \epsilon$. Assume this is not the case, that is,

$$|\left\{t \in [m] : p_{t,+} > 1 - \epsilon\right\}| \geq l$$

Suppose the first l times $\mathcal{A}$ predicts $p_{t,+} > 1-\epsilon$ are $1 \leq t_1 < \ldots < t_l \leq m$. Then, according to the adversary's strategy, $y_{t_1} = \ldots = y_{t_l} = -1$. Thus, the cumulative mistake penalty made by $\mathcal{A}$ up to time t_l is at least

$$\sum_{i=1}^{l} p_{t_i,+} > l(1-\epsilon) = k$$

This implies that $\mathcal{A}$ has a cumulative mistake penalty $> k$, contradiction. Therefore throughout the interaction, the number of solid edges used is at most $l-1$. Since T is $(l-1,m)$-difficult, any path that going downward from the root using $l-1$ solid edges must be of length at least m, hence the interaction between the learner and the adversary lasts for at least m rounds.

(2) We claim that over the first m rounds, there are at most $\frac{2l}{\epsilon}$ rounds such that $\mathcal{A}$ predicts $p_{t,-} > \epsilon/2$ and $p_{t,+} \leq 1-\epsilon$. Assume this is not the case, that is,

$$|\left\{t \in [m] : p_{t,-} > \epsilon/2 \wedge p_{t,+} \leq 1-\epsilon\right\}| \geq \frac{2l}{\epsilon}$$

Suppose the first $g = \lceil \frac{2l}{\epsilon} \rceil$ times $\mathcal{A}$ predicts -1 are $1 \leq s_1 < \ldots < s_g \leq m$. Then, according to the adversary's strategy, $y_{s_1} = \ldots = y_{s_g} = +1$. Thus the cumulative mistake penalty made by $\mathcal{A}$ up to time s_g is at least

$$\sum_{i=1}^{g} p_{s_i,-} > g \cdot \frac{\epsilon}{2} \geq l(1-\epsilon) = k$$

This implies that $\mathcal{A}$ has a cumulative mistake penalty $> k$ over time, contradiction.

Therefore, among the first m rounds, there are at most $l + (l + \frac{2l}{\epsilon}) = 2l + \frac{2l}{\epsilon}$ rounds such that $p_{t,+} > 1-\epsilon$ or $p_{t,-} > \epsilon/2$. Thus there are at least $(m - \frac{2l}{\epsilon} - 2l)$ rounds such that $p_{t,+} \leq 1-\epsilon$ and $p_{t,-} \leq \epsilon/2$, implying $1 - p_{t,+} - p_{t,-} \geq \epsilon/2$. Thus, the cumulative abstention penalty up to time m is at least

$$(m - \frac{2l}{\epsilon} - 2l) \cdot \frac{\epsilon}{2} \geq a. \qquad \square$$

Proof of Theorem 7.10. Let $\mathcal{X}$ be an infinite set. Let $\mathcal{H}$ be the $\mathcal{C}^d$, the class of unions of at most d singletons. Note that $\mathcal{H}^l = \mathcal{C}^{l+d}$. Hence l-bias assumption with respect to $\mathcal{H}$ is equivalent to $\mathcal{C}^{l+d}$-realizability. The rest of the proof is analogous to the proof of Theorem 7.9. $\square$

7.12 Acknowledgements

This chapter is based on the material as it appears in Conference on Learning Theory 2016 (Chicheng Zhang and Kamalika Chaudhuri, "The Extended Littlestone's Dimension for Learning with Mistakes and Abstentions"). The dissertation author is the primary investigator and author of this material.

Chapter 8

Confidence-based Active Learning

8.1 Introduction

In this chapter, we study active learning of classifiers in the agnostic setting. The primary algorithm for agnostic active learning is called disagreement-based active learning. The main idea is as follows. A set V_k of possible risk minimizers is maintained with time, and the label of an example x is queried if there exist two hypotheses h_1 and h_2 in V_k such that $h_1(x) \neq h_2(x)$. This algorithm is consistent in the agnostic setting [CAL94, BBL09, DHM07, Han07, BDL09, Han09, BHLZ10, Kol10]; however, due to the conservative label query policy, its label requirement is high. A line of work due to [BBZ07, BL13, ABL14] have provided algorithms that achieve better label complexity for linear classification on the uniform distribution over the unit sphere as well as log-concave distributions; however, their algorithms are limited to these specific cases, and it is unclear how to apply them more generally.

Thus, a major challenge in the agnostic active learning literature has been to find a general active learning strategy that applies to any hypothesis class and data distribution, is consistent in the agnostic case, and has a better label requirement than disagreement based active learning. This has been mentioned as an open problem by several works, such as [BBL09, Das11, BL13].

We provide such an algorithm in this chapter. Interestingly, the algorithm is inspired by batch confidence-rated prediction studied in Chapter 6 - we call it *confidence-based active learning*. While a line of work [EYW12, WHEY15] establishes connections between disagreement-based

active learning and confidence-rated prediction with zero error guarantee, this chapter takes a step further, establishing connections between active learning and confidence-rated predictors with nonzero error guarantees. Combining our algorithmic framework with our novel confidence-rated predictor (Algorithm 6.1), we get a general active learning algorithm consistent in the agnostic setting. We provide a characterization of the label complexity of our algorithm, and show that this is better than disagreement-based active learning in general. Finally, we show that for linear classification with respect to the uniform distribution and log-concave distributions, our bounds reduce to those of [BBZ07, BL13].

8.2 Active Learning via Confidence-rated Prediction

We present our algorithmic framework in this section. Our active learning algorithm proceeds in epochs, where the goal of epoch k is to achieve excess generalization error $\epsilon_k = 2^{-k}$, by querying a fresh batch of labels. The algorithm maintains a candidate set V_k that is guaranteed to contain the true risk minimizer.

The critical decision at each epoch is how to select a subset of unlabeled examples whose labels should be queried. We make this decision using a confidence-rated predictor P. At epoch k, we run P with candidate hypothesis set $V = V_k$ and error guarantee $\eta = O(\epsilon_k)$. Whenever P abstains, we query the label of the example. The number of labels m_k queried is adjusted so that it is enough to achieve excess generalization error ϵ_{k+1}.

An outline is described in Algorithm 8.1; we next discuss each individual component in detail.

8.2.1 Candidate Sets

At epoch k, we maintain a set V_k of candidate hypotheses guaranteed to contain the true risk minimizer $h^*(D)$ (with high probability). In the realizable case, we use a version space (See Definition 3.4) as our candidate set.

Lemma 8.1. *Suppose we run Algorithm 8.1 in the realizable case with inputs example oracle EX, labeling oracle* LABEL*, hypothesis class $\mathcal{H}$, confidence-rated predictor P, target excess error ϵ and target confidence δ. Then, with probability 1, $h^*(D) \in V_k$, for all $k = 1, 2, \ldots, k_0 + 1$.*

In the non-realizable case, the version space is usually empty; we use instead a $(1-\delta)$-

Algorithm 8.1 Active Learning Algorithm: Outline

Inputs: Example oracle EX, labeling oracle LABEL, hypothesis class $\mathcal{H}$ of VC dimension d, confidence-rated predictor P, target excess error ϵ and target confidence δ.
Set $k_0 = \lceil \log 1/\epsilon \rceil$. Initialize candidate set $V_1 = \mathcal{H}$.
for $k = 1, 2, ..k_0$ **do**
  Set $\epsilon_k = \epsilon 2^{k_0 - k + 1}$, $\delta_k = \frac{\delta}{2(k_0 - k + 1)^2}$.
  Call $\mathcal{U}$ to generate a fresh unlabeled sample $U_k = \{z_{k,1}, ..., z_{k,n_k}\}$ of size $n_k = 192(\frac{256}{\epsilon_k})^2 \left(d \ln \frac{256}{\epsilon_k} + \ln \frac{288}{\delta_k} \right)$.
  Run confidence-rated predictor P with inputs $V = V_k$, $U = U_k$ and error guarantee $\eta = \epsilon_k/64$ to get abstention probabilities $\gamma_{k,1}, \ldots, \gamma_{k,n_k}$ on the examples in U_k. These probabilities induce a distribution Γ_k on U_k. Let $\phi_k = \mathbb{P}_{x \sim U_k}[P(x) = \perp] = \frac{1}{n_k} \sum_{i=1}^{n_k} \gamma_{k,i}$.
  if in the Realizable Case **then**
    Let $m_k = \frac{768 \phi_k}{\epsilon_k} \left(d \ln \frac{768 \phi_k}{\epsilon_k} + \ln \frac{48}{\delta_k} \right)$. Draw m_k i.i.d examples from Γ_k and query $\mathcal{O}$ for labels of these examples to get a labeled data set S_k. Update V_{k+1} using S_k: $V_{k+1} := \{h \in V_k : h(x) = y, \text{ for all } (x, y) \in S_k\}$.
  else
    In the non-realizable case, use Algorithm 8.2 with inputs hypothesis set V_k, distribution Γ_k, target excess error $\frac{\epsilon_k}{8\phi_k}$, target confidence $\frac{\delta_k}{2}$, and the labeling oracle $\mathcal{O}$ to get a new hypothesis set V_{k+1}.
  end if
end for
return an arbitrary $\hat{h} \in V_{k_0+1}$.

confidence set (see Definition 3.5) for the true risk minimizer. In the non-realizable case, our candidate sets are $(1-\delta)$-confidence sets for $h^*(D)$. The precise setting of V_k is explained in Algorithm 8.2.

Lemma 8.2. *Suppose we run Algorithm 8.1 in the non-realizable case with inputs example oracle EX, labeling oracle* LABEL*, hypothesis class $\mathcal{H}$, confidence-rated predictor P, target excess error ϵ and target confidence δ. Then with probability $1 - \delta$, $h^*(D) \in V_k$, for all $k = 1, 2, \ldots, k_0 + 1$.*

8.2.2 Label Query

We next discuss our label query procedure – which examples should we query labels for, and how many labels should we query at each epoch?

Which Labels to Query? Our goal is to query the labels of the most informative examples. To choose these examples while still maintaining consistency, we use a confidence-rated predictor P with guaranteed error. The inputs to the predictor are our candidate hypothesis set V_k which contains (w.h.p) the true risk minimizer, a fresh set U_k of unlabeled examples, and an error guarantee $\eta = \epsilon_k/64$. For notation simplicity, assume the elements in U_k are distinct. The output is a sequence of abstention probabilities $\{\gamma_{k,1}, \gamma_{k,2}, \ldots, \gamma_{k,n_k}\}$, for each example in U_k. It

induces a distribution Γ_k over U_k, from which we independently draw examples for label queries.

How Many Labels to Query? The goal of epoch k is to achieve excess generalization error ϵ_k. To achieve this, passive learning requires $\tilde{O}(d/\epsilon_k)$ labeled examples in the realizable case, and $\tilde{O}(d(\nu^*(D)+\epsilon_k)/\epsilon_k^2)$ examples in the agnostic case. A key observation is that in order to achieve excess generalization error ϵ_k on D, it suffices to achieve a much larger excess generalization error $O(\epsilon_k/\phi_k)$ on the data distribution induced by Γ_k and $D_{Y|X}$, where ϕ_k is the fraction of examples on which the confidence-rated predictor abstains.

In the realizable case, we achieve this by sampling $m_k = \frac{768\phi_k}{\epsilon_k}\left(d\ln\frac{768\phi_k}{\epsilon_k} + \ln\frac{48}{\delta_k}\right)$ i.i.d examples from Γ_k, and querying their labels to get a labeled dataset S_k. Observe that as ϕ_k is the abstention probability of P with guaranteed error $\leq O(\epsilon_k)$, it is generally smaller than the measure of the disagreement region of the version space; this key fact results in improved label complexity over disagreement-based active learning. This sampling procedure has the following property:

Lemma 8.3. *Suppose we run Algorithm 8.1 in the realizable case with inputs example oracle $\mathcal{U}$, labeling oracle $\mathcal{O}$, hypothesis class $\mathcal{H}$, confidence-rated predictor P, target excess error ϵ and target confidence δ. Then with probability $1-\delta$, for all $k = 1, 2, \ldots, k_0+1$, and for all $h \in V_k$, $\mathrm{err}_D(h) \leq \epsilon_k$. In particular, the $\hat{h}$ returned at the end of the algorithm satisfies $\mathrm{err}_D(\hat{h}) \leq \epsilon$.*

The agnostic case has an added complication – in practice, the value of ν^* is not known ahead of time. Inspired by [Kol10], we use a doubling procedure similar to Algorithm 4.4(stated in Algorithm 8.2) which adaptively finds the number m_k of labeled examples to be queried and queries them. The following two lemmas illustrate its properties – that it is consistent, and that it does not use too many label queries.

Lemma 8.4. *Suppose we run Algorithm 8.2 with inputs hypothesis set V, distribution Δ, labeling oracle $\mathcal{O}$, target excess error $\tilde{\epsilon}$ and target confidence $\tilde{\delta}$. Let $\tilde{\Delta}$ be the joint distribution on $\mathcal{X}\times\mathcal{Y}$ induced by Δ and $D_{Y|X}$. Then there exists an event $\tilde{E}$, $\mathbb{P}(\tilde{E}) \geq 1-\tilde{\delta}$, such that on $\tilde{E}$, (1) Algorithm 8.2 halts and (2) the set V_{j_0} has the following properties:*

1. *If for $h \in \mathcal{H}$, $\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(h^*(\tilde{\Delta})) \leq \tilde{\epsilon}/2$, then $h \in V_{j_0}$.*
2. *On the other hand, if $h \in V_{j_0}$, then $\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(h^*(\tilde{\Delta})) \leq \tilde{\epsilon}$.*

When event $\tilde{E}$ happens, we say Algorithm 8.2 succeeds.

Lemma 8.5. *Suppose we run Algorithm 8.2 with inputs hypothesis set V, distribution Δ, labeling oracle $\mathcal{O}$, target excess error $\tilde{\epsilon}$ and target confidence $\tilde{\delta}$. There exists some absolute constant $c_1 > 0$, such that on the event that Algorithm 8.2 succeeds, $n_{j_0} \leq c_1 \left((d \ln \frac{1}{\tilde{\epsilon}} + \ln \frac{1}{\tilde{\delta}}) \frac{\nu^*(\tilde{\Delta}) + \tilde{\epsilon}}{\tilde{\epsilon}^2} \right)$. Thus the total number of labels queried is $\sum_{j=1}^{j_0} n_j \leq 2 n_{j_0} \leq 2 c_1 \left((d \ln \frac{1}{\tilde{\epsilon}} + \ln \frac{1}{\tilde{\delta}}) \frac{\nu^*(\tilde{\Delta}) + \tilde{\epsilon}}{\tilde{\epsilon}^2} \right)$.*

The following lemma is a consequence of our label query procedure in the non-realizable case.

Lemma 8.6. *Suppose we run Algorithm 8.1 in the non-realizable case with inputs example oracle $\mathcal{U}$, labeling oracle $\mathcal{O}$, hypothesis class $\mathcal{H}$, confidence-rated predictor P, target excess error ϵ and target confidence δ. Then with probability $1 - \delta$, for all $k = 1, 2, \ldots, k_0 + 1$, and for all $h \in V_k$, $\mathrm{err}_D(h) \leq \mathrm{err}_D(h^*(D)) + \epsilon_k$. In particular, the $\hat{h}$ returned at the end of the algorithm satisfies $\mathrm{err}_D(\hat{h}) \leq \mathrm{err}_D(h^*(D)) + \epsilon$.*

Algorithm 8.2 An Adaptive Algorithm for Label Query Given Target Excess Error

Inputs: Hypothesis set V of VC dimension d, Distribution Δ, Labeling oracle $\mathcal{O}$, target excess error $\tilde{\epsilon}$, target confidence $\tilde{\delta}$.
for $j = 1, 2, \ldots$ **do**
  Draw $n_j = 2^j$ i.i.d examples from Δ; query their labels from $\mathcal{O}$ to get a labeled dataset S_j. Denote $\tilde{\delta}_j := \tilde{\delta}/(j(j+1))$.
  Train an ERM classifier $\hat{h}_j \in V$ over S_j.
  Define the set V_j as follows:

$$V_j = \left\{ h \in V : \mathrm{err}_{S_j}(h) \leq \mathrm{err}_{S_j}(\hat{h}_j) + \frac{\tilde{\epsilon}}{2} + \sigma(n_j, \tilde{\delta}_j) + \sqrt{\sigma(n_j, \tilde{\delta}_j) \rho_{S_j}(h, \hat{h}_j)} \right\}$$

  if $\sup_{h \in V_j} (\sigma(n_j, \tilde{\delta}_j) + \sqrt{\sigma(n_j, \tilde{\delta}_j) \rho_{S_j}(h, \hat{h}_j)}) \leq \frac{\tilde{\epsilon}}{6}$ **then**
    $j_0 = j$, **break**
  end if
end for
return V_{j_0}.

8.3 Performance Guarantees

An essential property of any active learning algorithm is statistical consistency – that it converges to the true risk minimizer given enough labeled examples. We observe that our algorithm is consistent provided we use *any* confidence-rated predictor P with guaranteed error as a subroutine. The consistency of our algorithm is a consequence of Lemmas 8.3 and 8.6 and is shown in Theorem 8.1.

Theorem 8.1 (Statistical Consistency)**.** *Suppose we run Algorithm 8.1 with inputs example oracle $\mathcal{U}$, labeling oracle $\mathcal{O}$, hypothesis class $\mathcal{H}$, confidence-rated predictor P, target excess error ϵ and target confidence δ. Then with probability $1-\delta$, the classifier $\hat{h}$ returned by Algorithm 8.1 satisfies* $\mathrm{err}_D(\hat{h}) - \mathrm{err}_D(h^*(D)) \leq \epsilon$.

We now establish a label complexity bound for our algorithm; however, this label complexity bound applies only if we use the predictor described in Algorithm 6.1 as a subroutine.

For any hypothesis set V, data distribution D, and η, define $\mathbf{\Phi}_D(V,\eta)$ to be the minimum abstention probability of a confidence-rated predictor which guarantees that the disagreement between its predicted labels and any $h \in V$ under $D_{\mathcal{X}}$ is at most η.

Formally, $\mathbf{\Phi}_D(V,\eta) = \min\{\mathbb{E}_D\gamma(x) : \mathbb{E}_D[\mathbb{1}(h(x)=+1)\beta(x) + \mathbb{1}(h(x)=-1)\alpha(x)] \leq \eta, \forall h \in V, \gamma(x)+\alpha(x)+\beta(x) \equiv 1, \gamma(x),\alpha(x),\beta(x) \geq 0\}$. Define $\phi(r,\eta) := \mathbf{\Phi}_D(\mathrm{B}_D(h^*,r),\eta)$. The label complexity of our active learning algorithm can be stated as follows.

Theorem 8.2 (Label Complexity)**.** *Suppose we run Algorithm 8.1 with inputs example oracle $\mathcal{U}$, labeling oracle $\mathcal{O}$, hypothesis class $\mathcal{H}$, confidence-rated predictor P of Algorithm 6.1, target excess error ϵ and target confidence δ. Then there exist constants $c_3, c_4 > 0$ such that with probability $1-\delta$:*

(1) In the realizable case, the total number of labels queried by Algorithm 8.1 is at most:

$$c_3 \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} \left(d \ln \frac{\phi(\epsilon_k, \epsilon_k/256)}{\epsilon_k} + \ln\left(\frac{\lceil \log(1/\epsilon) \rceil - k + 1}{\delta} \right) \right) \frac{\phi(\epsilon_k, \epsilon_k/256)}{\epsilon_k}$$

(2) In the agnostic case, the total number of labels queried by Algorithm 8.1 is at most:

$$c_4 \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} \left(d \ln \frac{\phi(2\nu^*(D)+\epsilon_k, \epsilon_k/256)}{\epsilon_k} + \ln\left(\frac{\lceil \log(1/\epsilon) \rceil - k + 1}{\delta} \right) \right) \cdot \frac{\phi(2\nu^*(D)+\epsilon_k, \epsilon_k/256)}{\epsilon_k} \left(1 + \frac{\nu^*(D)}{\epsilon_k} \right)$$

Comparison. The label complexity of disagreement-based active learning is characterized in terms of the *disagreement coefficient*. As $\mathbb{P}(\mathrm{Dis}(\mathrm{B}_D(h^*,r))) \leq \phi(r,0)$ [EYW10], in our notation, $\theta(r) \leq \sup_{r' \geq r} \frac{\phi(r',0)}{r'}$.

In the realizable case, the label complexity of disagreement-based active learning is $\tilde{O}(\theta(\epsilon) \cdot \ln(1/\epsilon) \cdot (d \ln \theta(\epsilon) + \ln\ln(1/\epsilon)))$ [Han14]. Our label complexity bound may be simplified

to:

$$\tilde{O}\left(\ln\frac{1}{\epsilon}\cdot\sup_{k\leq\lceil\log(1/\epsilon)\rceil}\frac{\phi(\epsilon_k,\epsilon_k/256)}{\epsilon_k}\cdot\left(d\ln\left(\sup_{k\leq\lceil\log(1/\epsilon)\rceil}\frac{\phi(\epsilon_k,\epsilon_k/256)}{\epsilon_k}\right)+\ln\ln\frac{1}{\epsilon}\right)\right),$$

which is essentially the bound of [Han14] with $\theta(\epsilon)$ replaced by $\sup_{k\leq\lceil\log(1/\epsilon)\rceil}\frac{\phi(\epsilon_k,\epsilon_k/256)}{\epsilon_k}$. As enforcing a lower error guarantee requires more abstention, $\phi(r,\eta)$ is a decreasing function of η; as a result,

$$\sup_{k\leq\lceil\log(1/\epsilon)\rceil}\frac{\phi(\epsilon_k,\epsilon_k/256)}{\epsilon_k}\leq\theta(\epsilon),$$

and our label complexity is better.

In the agnostic case, for disagreement-based active-learning, [DHM07] provides a label complexity bound of $\tilde{O}(\theta(2\nu^*(D)+\epsilon)\cdot(d\frac{\nu^*(D)^2}{\epsilon^2}\ln(1/\epsilon)+d\ln^2(1/\epsilon)))$. In contrast, by Proposition 8.1 our label complexity is at most:

$$\tilde{O}\left(\sup_{k\leq\lceil\log(1/\epsilon)\rceil}\frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{2\nu^*(D)+\epsilon_k}\cdot\left(d\frac{\nu^*(D)^2}{\epsilon^2}\ln(1/\epsilon)+d\ln^2(1/\epsilon)\right)\right)$$

Again, this is essentially the bound of [DHM07] with $\theta(2\nu^*(D)+\epsilon)$ replaced by the smaller quantity

$$\sup_{k\leq\lceil\log(1/\epsilon)\rceil}\frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{2\nu^*(D)+\epsilon_k},$$

[Han14] has provided a more refined analysis of disagreement-based active learning that gives a label complexity of $\tilde{O}(\theta(\nu^*(D)+\epsilon)(\frac{\nu^*(D)^2}{\epsilon^2}+\ln\frac{1}{\epsilon})(d\ln\theta(\nu^*(D)+\epsilon)+\ln\ln\frac{1}{\epsilon}))$; observe that their dependence is still on $\theta(\nu^*(D)+\epsilon)$. We leave a more refined label complexity analysis of our algorithm for future work.

8.3.1 Tsybakov Noise Conditions

An important sub-case of learning from noisy data is learning under the Tsybakov noise conditions [Tsy04].

Definition 8.1. *(Tsybakov Noise Condition) Let $\kappa\geq1$. A labeled data distribution D over $\mathcal{X}\times\mathcal{Y}$ satisfies (C_0,κ)-Tsybakov Noise Condition with respect to a hypothesis class $\mathcal{H}$ for some constant $C_0>0$, if for all $h\in\mathcal{H}$, $\rho_D(h,h^*(D))\leq C_0(\mathrm{err}_D(h)-\mathrm{err}_D(h^*(D)))^{\frac{1}{\kappa}}$.*

The following theorem shows the performance guarantees achieved by Algorithm 8.1 under the Tsybakov noise conditions.

Theorem 8.3. *Suppose (C_0, κ)-Tsybakov Noise Condition holds for D with respect to $\mathcal{H}$. Then Algorithm 8.1 with inputs example oracle $\mathcal{U}$, labeling oracle $\mathcal{O}$, hypothesis class $\mathcal{H}$, confidence-rated predictor P of Algorithm 6.1, target excess error ϵ and target confidence δ satisfies the following properties. There exists a constant $c_5 > 0$ such that with probability $1-\delta$, the total number of labels queried by Algorithm 8.1 is at most:*

$$c_5 \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} \left(d \ln \left(\phi(C_0 \epsilon_k^{\frac{1}{\kappa}}, \epsilon_k/256) \epsilon_k^{\frac{1}{\kappa}-2} \right) + \ln \left(\frac{\lceil \log \frac{1}{\epsilon} \rceil - k + 1}{\delta} \right) \right) \phi(C_0 \epsilon_k^{\frac{1}{\kappa}}, \epsilon_k/256) \epsilon_k^{\frac{1}{\kappa}-2}$$

Comparison. For disagreement-based active learning, [Han14] provides a label complexity bound of $\tilde{O}(\theta(C_0\epsilon^{\frac{1}{\kappa}})\epsilon^{\frac{2}{\kappa}-2} \ln \frac{1}{\epsilon} (d \ln \theta(C_0 \epsilon^{\frac{1}{\kappa}}) + \ln \ln \frac{1}{\epsilon}))$. For $\kappa > 1$, by Proposition 8.2, our label complexity is at most:

$$\tilde{O}\left(\sup_{k \le \lceil \log(1/\epsilon) \rceil} \frac{\phi(C_0 \epsilon_k^{1/\kappa}, \epsilon_k/256)}{\epsilon_k^{1/\kappa}} \cdot \epsilon_k^{2/\kappa - 2} \cdot d \ln(1/\epsilon) \right),$$

For $\kappa = 1$, our label complexity is at most

$$\tilde{O}\left(\ln \frac{1}{\epsilon} \cdot \sup_{k \le \lceil \log(1/\epsilon) \rceil} \frac{\phi(C_0\epsilon_k, \epsilon_k/256)}{\epsilon_k} \cdot \left(d \ln \left(\sup_{k \le \lceil \log(1/\epsilon) \rceil} \frac{\phi(C_0\epsilon_k, \epsilon_k/256)}{\epsilon_k} \right) + \ln \ln \frac{1}{\epsilon} \right) \right).$$

In both cases, our bounds are better, as $\sup_{k \le \lceil \log(1/\epsilon) \rceil} \cdot \frac{\phi(C_0 \epsilon_k^{1/\kappa}, \epsilon_k/256)}{C_0 \epsilon_k^{1/\kappa}} \le \theta(C_0 \epsilon^{1/\kappa})$. In further work, [HY12] provides a refined analysis with a bound of $\tilde{O}(\theta(C_0\epsilon^{\frac{1}{\kappa}})\epsilon^{\frac{2}{\kappa}-2}\, d \ln \theta(C_0 \epsilon^{\frac{1}{\kappa}}))$; however, this work is not directly comparable to ours, as they need prior knowledge of C_0 and κ.

8.3.2 Case Study: Linear Classification under the Log-concave Distribution

We now consider learning linear classifiers with respect to log-concave data distribution on $\mathbb{R}^d$. In this case, for any r, the disagreement coefficient $\theta(r) \le O(\sqrt{d} \ln(1/r))$ [BL13]; however, for any $\eta > 0$, $\frac{\phi(r,\eta)}{r} \le O(\ln(r/\eta))$ (see Lemma 8.12), which is much smaller so long as η/r is not too small. This leads to the following label complexity bounds.

Corollary 8.1. *Suppose $D_{\mathcal{X}}$ is isotropic and log-concave on $\mathbb{R}^d$, and $\mathcal{H}$ is the set of homogeneous linear classifiers on $\mathbb{R}^d$. Then Algorithm 8.1 with inputs example oracle $\mathcal{U}$, labeling oracle $\mathcal{O}$, hypothesis class $\mathcal{H}$, confidence-rated predictor P of Algorithm 6.1, target excess error ϵ and target confidence δ satisfies the following properties. With probability $1-\delta$:*

(1) In the realizable case, there exists some constant $c_8 > 0$ such that the total number of labels queried is at most $O\left(\ln\frac{1}{\epsilon}\left(d+\ln\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta}\right)\right)$.

(2) In the agnostic case, there exists some constant $c_9 > 0$ such that the total number of labels queried is at most $O\left(\frac{\left(\nu^*(D)^2\right)\epsilon^2+\ln\frac{1}{\epsilon}}{\cdot}\ln\frac{\epsilon+\nu^*(D)}{\epsilon}\cdot\left(d\ln\frac{\epsilon+\nu^*(D)}{\epsilon}+\ln\frac{1}{\delta}\right)+\ln\frac{\epsilon+\nu^*(D)}{\epsilon}\cdot\ln\frac{1}{\epsilon}\ln\ln\frac{1}{\epsilon}\right)$.

(3) If (C_0,κ)-Tsybakov Noise condition holds for D with respect to $\mathcal{H}$, then there exists some constant $c_{10} > 0$ (that depends on C_0,κ) such that the total number of labels queried is at most $O\left(\epsilon^{\frac{2}{\kappa}-2}\ln\frac{1}{\epsilon}\cdot\left(d\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta}\right)\right)$.

In the realizable case, our bound matches [BL13]. For disagreement-based algorithms, the bound is $\tilde{O}\left(d^{\frac{3}{2}}\ln^2\frac{1}{\epsilon}(\ln d+\ln\ln\frac{1}{\epsilon})\right)$, which is worse by a factor of $O(\sqrt{d}\ln(1/\epsilon))$. [BL13] does not address the fully agnostic case directly; however, if $\nu^*(D)$ is known a-priori, then their algorithm can achieve roughly the same label complexity as ours.

For the Tsybakov Noise Condition with $\kappa > 1$, [BBZ07, BL13] provides a label complexity bound of $\tilde{O}\left(\epsilon^{\frac{2}{\kappa}-2}\ln^2\frac{1}{\epsilon}(d+\ln\ln\frac{1}{\epsilon})\right)$ with an algorithm that has a-priori knowledge of C_0 and κ. We get a slightly better bound. On the other hand, a disagreement based algorithm [Han14] gives a label complexity of $\tilde{O}\left(d^{\frac{3}{2}}\ln^2\frac{1}{\epsilon}\epsilon^{\frac{2}{\kappa}-2}(\ln d+\ln\ln\frac{1}{\epsilon})\right)$. Again our bound is better by factor of $\Omega(\sqrt{d})$ over disagreement-based algorithms. For $\kappa = 1$, we can tighten our label complexity to get a $\tilde{O}\left(\ln\frac{1}{\epsilon}(d+\ln\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta})\right)$ bound, which again matches [BL13], and is better than the ones provided by disagreement-based algorithm – $\tilde{O}\left(d^{\frac{3}{2}}\ln^2\frac{1}{\epsilon}(\ln d+\ln\ln\frac{1}{\epsilon})\right)$ [Han14].

8.4 Additional Notations and Concentration Lemmas

For an unlabeled sample U_k drawn iid from unlabeled distribution $D_{\mathcal{X}}$, we use $\tilde{U}_k$ to denote the joint distribution over $\mathcal{X}\times\mathcal{Y}$ induced by uniform distribution over U_k and $D_{Y|X}$. We have the following concentration result, whose proof is deferred to Section 8.8.

Lemma 8.7. *If the size of n_k of the unlabeled dataset U_k is at least $192(\frac{256}{\epsilon_k})^2(d\ln\frac{256}{\epsilon_k}+\ln\frac{288}{\delta_k})$, then with probability $1-\delta_k/4$, the following conditions hold for all $h,h'\in V_k$:*

$$|\operatorname{err}_D(h) - \operatorname{err}_{\tilde{U}_k}(h)| \leq \frac{\epsilon_k}{64} \tag{8.1}$$

$$|(\operatorname{err}_D(h) - \operatorname{err}_D(h')) - (\operatorname{err}_{\tilde{U}_k}(h) - \operatorname{err}_{\tilde{U}_k}(h'))| \leq \frac{\epsilon_k}{32} \tag{8.2}$$

$$|\rho_D(h,h') - \rho_{\tilde{U}_k}(h,h')| \leq \frac{\epsilon_k}{64} \tag{8.3}$$

Lemma 8.8. *If the size of n_k of the unlabeled dataset U_k is at least $192(\frac{256}{\epsilon_k})^2(d\ln\frac{256}{\epsilon_k} + \ln\frac{288}{\delta_k})$, then with probability $1-\delta_k/4$, the following hold:*

(1) The outputs $\{(\alpha_{k,i},\beta_{k,i},\gamma_{k,i})\}_{i=1}^{n_k}$ of any confidence-rated predictor with inputs hypothesis set V_k, unlabeled data U_k, and error bound $\epsilon_k/64$ satisfy:

$$\frac{1}{n_k}\sum_{i=1}^{n_k}[\mathbb{1}(h(x_i) \neq h'(x_i))(1-\gamma_{k,i})] \leq \frac{\epsilon_k}{32}; \tag{8.4}$$

(2) The outputs $\{(\alpha_{k,i},\beta_{k,i},\gamma_{k,i})\}_{i=1}^{n_k}$ of the confidence-rated predictor of Algorithm 6.1 with inputs hypothesis set V_k, unlabeled data U_k, and error bound $\epsilon_k/64$ satisfy:

$$\phi_k \leq \mathbf{\Phi}_D(V_k, \frac{\epsilon_k}{128}) + \frac{\epsilon_k}{256} \tag{8.5}$$

We use $\tilde{\Gamma}_k$ to denote the joint distribution over $\mathcal{X}\times\mathcal{Y}$ induced by Γ_k and $D_{Y|X}$. Denote $\gamma_k(x):\mathcal{X}\to[0,1]$, where $\gamma_k(x_i)=\gamma_{k,i}$, and 0 elsewhere. It can be seen that $\Gamma_k(\{x\}) = \frac{\gamma_k(x)}{n_k\phi_k}$ and $\tilde{\Gamma}_k(\{(x,y)\}) = \frac{\tilde{U}_k(\{(x,y)\})\gamma_k(x)}{\phi_k}$. Also, Equations (8.4) and (8.5) of Lemma 8.8 can be restated as

$$\forall h,h' \in V_k, \mathbb{E}_{\tilde{U}_k}[(1-\gamma_k(x))\mathbb{1}(h(x)\neq h'(x))] \leq \frac{\epsilon_k}{32}$$

$$\mathbb{E}_{\tilde{U}_k}[\gamma_k(x)] = \phi_k \leq \mathbf{\Phi}_D(V_k, \frac{\epsilon_k}{128}) + \frac{\epsilon_k}{256}$$

In the realizable case, define event

$$E_r = \{\text{For all } k=1,2,\ldots,k_0\text{: Equations (8.1), (8.2), (8.3), (8.4), (8.5) hold for } \tilde{U}_k \text{ and all classifiers consistent with } S_k \text{ have error at most } \frac{\epsilon_k}{8\phi_k} \text{ with respect to } \tilde{\Gamma}_k\ \}.$$

Fact 8.1. $\mathbb{P}(E_r) \geq 1-\delta$.

Proof. By Equation (A.3) of Lemma A.5, with probability $1-\delta_k/2$, if $h\in V_k$ is consistent with

S_k, then

$$\mathrm{err}_{\tilde{\Gamma}_k}(h) \leq \sigma(m_k, \delta_k/2)$$

Because $m_k = \frac{768\phi_k}{\epsilon_k}(d\ln\frac{768\phi_k}{\epsilon_k} + \ln\frac{48}{\delta_k})$, we have $\mathrm{err}_{\tilde{\Gamma}_k}(h) \leq \epsilon_k/8\phi_k$. The fact follows from combining the fact above with Lemma 8.7 and Lemma 8.8, and the union bound.

□

In the non-realizable case, define event

$E_a = \{$For all $k = 1, 2, \ldots, k_0$: Equations (8.1), (8.2), (8.3), (8.4), (8.5) hold for $\tilde{U}_k$, and Algorithm 8.2 succeeds with inputs hypothesis set $V = V_k$, distribution $\Delta = \Gamma_k$, labeling oracle $\mathcal{O}$, target excess error $\tilde{\epsilon} = \dfrac{\epsilon_k}{8\phi_k}$ and target confidence $\tilde{\delta} = \dfrac{\delta_k}{2}\}$.

Fact 8.2. $\mathbb{P}(E_a) \geq 1 - \delta$.

Proof. This is an immediate consequence of Lemma 8.7, Lemma 8.8, Lemma 8.4 and union bound. □

8.5 Proofs related to the properties of Algorithm 8.2

We first establish some properties of Algorithm 8.2. The inputs to Algorithm 8.2 are a set V of hypotheses of VC dimension d, a distribution Δ, a labeling oracle $\mathcal{O}$, a target excess error $\tilde{\epsilon}$ and a target confidence $\tilde{\delta}$.

We define the event

$$\tilde{E} = \{\text{For all } j = 1, 2, \ldots : \text{Equations (A.2)-(A.5) hold for sample } S_j \text{ with } n = n_j \text{ and } \delta = \tilde{\delta}_j \ \}$$

By union bound, $\mathbb{P}(\tilde{E}) \geq 1 - \sum_j \tilde{\delta}_j \geq 1 - \tilde{\delta}$.

Proof of Lemma 8.4. Assume $\tilde{E}$ happens. For the proof of (1), define j_{max} as the smallest integer j such that $\sigma(n_j, \tilde{\delta}_j) \leq \tilde{\epsilon}^2/144$. Since $n_{j_{max}}$ is a power of 2,

$$n_{j_{max}} \leq 2\min\left\{n = 1, 2, \ldots : \frac{8(2d\ln\frac{2en}{d} + \ln\frac{24\log n(\log n+1)}{\delta})}{n} \leq \frac{\epsilon^2}{144}\right\}$$

Thus, $n_{j_{max}} \leq 192\frac{144}{\tilde{\epsilon}^2}\left\{d\ln\frac{144}{\tilde{\epsilon}} + \ln\frac{24}{\tilde{\delta}}\right\}$. Then in round j_{max}, the stopping criterion (6) of Algorithm 8.2 is satisified; thus, Algorithm 8.2 halts with $j_0 \leq j_{max}$.

To prove (2.1), we observe that as $h^*(\tilde{\Delta})$ is the risk minimizer in V, if h satisfies $\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(h^*(\tilde{\Delta})) \leq \frac{\tilde{\epsilon}}{2}$, then $\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\hat{h}_{j_0}) \leq \frac{\tilde{\epsilon}}{2}$. By Equation (A.4) of Lemma A.5,

$$\begin{aligned}(\mathrm{err}_{S_{j_0}}(h) - \mathrm{err}_{S_{j_0}}(\hat{h}_{j_0})) &\leq (\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\hat{h}_{j_0})) + \sigma(n_{j_0}, \tilde{\delta}_{j_0}) + \sqrt{\sigma(n_{j_0}, \tilde{\delta}_{j_0})\rho_{S_{j_0}}(h, \hat{h}_{j_0})} \\ &\leq \frac{\tilde{\epsilon}}{2} + \sigma(n_{j_0}, \tilde{\delta}_{j_0}) + \sqrt{\sigma(n_{j_0}, \tilde{\delta}_{j_0})\rho_{S_{j_0}}(h, \hat{h}_{j_0})}\end{aligned}$$

Hence $h \in V_{j_0}$.

For the proof of (2.2), note first that by (2.1), in particular, $h^*(\tilde{\Delta}) \in V_{j_0}$. Hence by Equation (A.4) of Lemma A.5, and the stopping criterion Equation (6),

$$\begin{aligned}& (\mathrm{err}_{\tilde{\Delta}}(\hat{h}_{j_0}) - \mathrm{err}_{\tilde{\Delta}}(h^*(\tilde{\Delta}))) - (\mathrm{err}_{S_{j_0}}(\hat{h}_{j_0}) - \mathrm{err}_{S_{j_0}}(h^*(\tilde{\Delta}))) \\ \leq\ & \sigma(n_{j_0}, \tilde{\delta}_{j_0}) + \sqrt{\sigma(n_{j_0}, \tilde{\delta}_{j_0})\rho_{S_{j_0}}(\hat{h}_{j_0}, h^*(\tilde{\Delta}))} \\ \leq\ & \frac{\tilde{\epsilon}}{6}\end{aligned}$$

Thus,

$$\mathrm{err}_{\tilde{\Delta}}(\hat{h}_{j_0}) - \mathrm{err}_{\tilde{\Delta}}(h^*(\tilde{\Delta})) \leq \frac{\tilde{\epsilon}}{6} \tag{8.6}$$

On the other hand, if $h \in V_{j_0}$, then

$$(\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\hat{h}_{j_0})) - (\mathrm{err}_{S_{j_0}}(h) - \mathrm{err}_{S_{j_0}}(\hat{h}_{j_0})) \leq \sigma(n_{j_0}, \tilde{\delta}_{j_0}) + \sqrt{\sigma(n_{j_0}, \tilde{\delta}_{j_0})\rho_{S_{j_0}}(h, \hat{h}_{j_0})} \leq \frac{\tilde{\epsilon}}{6}$$

By definition of V_{j_0},

$$(\mathrm{err}_{S_{j_0}}(h) - \mathrm{err}_{S_{j_0}}(\hat{h}_{j_0})) \leq \sigma(n_{j_0}, \tilde{\delta}_{j_0}) + \sqrt{\sigma(n_{j_0}, \tilde{\delta}_{j_0})\rho_{S_{j_0}}(h, \hat{h}_{j_0})} + \frac{\tilde{\epsilon}}{2} \leq \frac{2\tilde{\epsilon}}{3}$$

Hence,

$$\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\hat{h}_{j_0}) \leq \frac{5\tilde{\epsilon}}{6} \tag{8.7}$$

Combining Equations (8.6) and (8.7), we have

$$\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(h^*(\tilde{\Delta})) \leq \tilde{\epsilon}$$

□

Proof of Lemma 8.5. Assume $\tilde{E}$ happens. For each j, by triangle inequality, we have that $\rho_{S_j}(\hat{h}_j, h) \leq \mathrm{err}_{S_j}(\hat{h}_j) + \mathrm{err}_{S_j}(h)$. If $h \in V_j$, then, by defintion of V_j,

$$\mathrm{err}_{S_j}(h) - \mathrm{err}_{S_j}(\hat{h}_j) \leq \frac{\tilde{\epsilon}}{2} + \sigma(n_j, \tilde{\delta}_j) + \sqrt{\sigma(n_j, \tilde{\delta}_j)\,\mathrm{err}_{S_j}(\hat{h}_j)} + \sqrt{\sigma(n_j, \tilde{\delta}_j)\,\mathrm{err}_{S_j}(h)}$$

Using the fact that $A \leq B + C\sqrt{A} \Rightarrow A \leq 2B + C^2$,

$$\mathrm{err}_{S_j}(h) \leq \tilde{\epsilon} + 2\,\mathrm{err}_{S_j}(\hat{h}_j) + 2\sqrt{\sigma(n_j, \tilde{\delta}_j)\,\mathrm{err}_{S_j}(\hat{h}_j)} + 3\sigma(n_j, \tilde{\delta}_j) \leq 3\,\mathrm{err}_{S_j}(\hat{h}_j) + 4\sigma(n_j, \tilde{\delta}_j) + \tilde{\epsilon}$$

Since

$$\mathrm{err}_{S_j}(\hat{h}_j) \leq \mathrm{err}_{S_j}(h^*(\tilde{\Delta})) \leq \nu^*(\tilde{\Delta}) + \sqrt{\sigma(n_j, \tilde{\delta}_j)\nu^*(\tilde{\Delta})} + \sigma(n_j, \tilde{\delta}_j) \leq 2\nu^*(\tilde{\Delta}) + 2\sigma(n_j, \tilde{\delta}_j),$$

by the triangle inequality, we get that for all $h \in V_j$,

$$\rho_{S_j}(h, \hat{h}_j) \leq \mathrm{err}_{S_j}(h) + \mathrm{err}_{S_j}(\hat{h}_j) \leq 8\nu^*(\tilde{\Delta}) + 12\sigma(n_j, \tilde{\delta}_j) + \tilde{\epsilon} \tag{8.8}$$

Now observe that for any j,

$$\begin{aligned}
& \sup_{h \in V_j} \sqrt{\sigma(n_j, \tilde{\delta}_j)\rho_{S_j}(h, \hat{h}_j)} + \sigma(n_j, \tilde{\delta}_j) \\
\leq\ & \sup_{h \in V_j} \max \det 2\sqrt{\sigma(n_j, \tilde{\delta}_j)\rho_{S_j}(h, \hat{h}_j)}, 2\sigma(n_j, \tilde{\delta}_j) \\
\leq\ & \max\left(2\sqrt{(8\nu^*(\tilde{\Delta}) + 12\sigma(n_j, \tilde{\delta}_j) + \tilde{\epsilon})\sigma(n_j, \tilde{\delta}_j)}, 2\sigma(n_j, \tilde{\delta}_j)\right) \\
\leq\ & \max\left(12\sqrt{2\nu^*(\tilde{\Delta})\sigma(n_j, \tilde{\delta}_j)}, \tilde{\epsilon}/6, 216\sigma(n_j, \tilde{\delta}_j)\right),
\end{aligned}$$

Where the first inequality follows from $A + B \leq 2\max(A, B)$, the second inequality follows from Equation (8.8), the third inequality follows from $\sqrt{A+B} \leq \sqrt{A} + \sqrt{B}$, $A + B + C \leq 3\max(A, B, C)$ and $\sqrt{AB} \leq \max(A, B)$.

It can be easily seen that there exists some constant $c_1 > 0$, such that taking $j_1 = \left\lceil \log\left(\frac{c_1}{2}(d \ln\frac{1}{\tilde{\epsilon}} + \ln\frac{1}{\delta})(\frac{\nu^*(\tilde{\Delta}) + \tilde{\epsilon}}{\tilde{\epsilon}^2})\right)\right\rceil$ ensures that $n_{j_1} \geq \frac{c_1}{2}\left(d \ln\frac{1}{\tilde{\epsilon}} + \ln\frac{1}{\delta}\right)\left(\frac{\nu^*(\tilde{\Delta}) + \tilde{\epsilon}}{\tilde{\epsilon}^2}\right)$; this, in turn,

suffices to make

$$\max\left(12\sqrt{2\nu^*(\tilde{\Delta})\sigma(n_j,\tilde{\delta}_j)},216\sigma(n_j,\tilde{\delta}_j)\right)\leq\tilde{\epsilon}/6$$

Hence the stopping criterion $\sup_{h\in V_j}\sqrt{\sigma(n_j,\tilde{\delta}_j)\rho_{S_j}(h,\hat{h}_j)}+\sigma(n_j,\tilde{\delta}_j)\leq\tilde{\epsilon}/6$ is satisfied in iteration j_1, and Algorithm 8.2 exits at iteration $j_0\leq j_1$, which ensures that

$$n_{j_0}\leq n_{j_1}\leq c_1\left(d\ln\frac{1}{\tilde{\epsilon}}+\ln\frac{1}{\tilde{\delta}}\right)\left(\frac{\nu^*(\tilde{\Delta})+\tilde{\epsilon}}{\tilde{\epsilon}^2}\right).$$

□

The following lemma examines the behavior of Algorithm 8.2 under the Tsybakov Noise Condition and is crucial in the proof of Theorem 8.3. We observe that even if the (C_0,κ)-Tsybakov Noise Conditions hold with respect to D, they do not necessarily hold with respect to Γ_k. In particular, it is not necessarily true that:

$$\rho_{\tilde{\Gamma}_k}(h,h^*(D))\leq C_0(\mathrm{err}_{\tilde{\Gamma}_k}(h)-\mathrm{err}_{\tilde{\Gamma}_k}(h^*(D)))^{\frac{1}{\kappa}},\forall h\in V_k$$

However, we show that an "approximate" Tsybakov Noise Condition with a significantly larger "C_0", namely Condition (8.9) is met by $\tilde{\Gamma}_k$ and V_k, with $C=\max(8C_0,4)\phi_k^{\frac{1}{\kappa}-1}$ and $\tilde{h}=h^*(D)$. In the Lemma below, we carefully track the dependence of the number of our label queries on C, since $C=\max(8C_0,4)\phi_k^{\frac{1}{\kappa}-1}$ can be $\omega(1)$ in our particular application.

Lemma 8.9. *Suppose we run Algorithm 8.2 with inputs hypothesis set V, distribution $\tilde{\Delta}$, labeling oracle $\mathcal{O}$, excess generalization error $\tilde{\epsilon}$ and confidence $\tilde{\delta}$. Then there exists some absolute constant $c_2>0$ (independent of C) such that the following holds. Suppose there exist $C>0$ and a classifier $\tilde{h}\in V$, such that*

$$\forall h\in V,\rho_{\tilde{\Delta}}(h,\tilde{h})\leq C\max(\tilde{\epsilon},\mathrm{err}_{\tilde{\Delta}}(h)-\mathrm{err}_{\tilde{\Delta}}(\tilde{h}))^{\frac{1}{\kappa}},\tag{8.9}$$

where $\tilde{\epsilon}$ is the target excess error parameter in Algorithm 8.2. Then, on the event that Algorithm 8.2 succeeds,

$$n_{j_0}\leq c_2\max\left((d\ln\frac{1}{\tilde{\epsilon}}+\ln\frac{1}{\tilde{\delta}})\tilde{\epsilon}^{-1},(d\ln(C\tilde{\epsilon}^{\frac{1}{\kappa}-2})+\ln\frac{1}{\tilde{\delta}})C\tilde{\epsilon}^{\frac{1}{\kappa}-2}\right)$$

Observe that Condition (8.9), the approximate Tsybakov Noise Condition in the statement of Lemma 8.9, is with respect to $\tilde{h}$, which is not necessarily the true risk minimizer in V

with respect to $\tilde{\Delta}$. We therefore prove Lemma 8.9 in three steps; first, in Lemma 8.10, we analyze the difference $\mathrm{err}_{\tilde{\Delta}}(\hat{h}) - \mathrm{err}_{\tilde{\Delta}}(\tilde{h})$, where $\hat{h}$ is the empirical risk minimizer. Then, in Lemma 8.11, we bound the difference $\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\tilde{h})$ for any $h \in V_j$ for some j. Finally, we combine these two lemmas to provide sample complexity bounds for the V_{j_0} output by Algorithm 8.2.

Proof of Lemma 8.9. Assume the event $\tilde{E}$ happens. Then,

Consider iteration j, by Lemma 8.11, if $h \in V_j$, then

$$\rho_{\tilde{\Delta}}(h, \hat{h}_j) \leq \rho_{\tilde{\Delta}}(h, \tilde{h}) + \rho_{\tilde{\Delta}}(\hat{h}_j, \tilde{h}) \leq \max\left(2C(36\tilde{\epsilon})^{\frac{1}{\kappa}}, 2C(52\sigma(n_j, \tilde{\delta}_j))^{\frac{1}{\kappa}}, 2C(6400C\sigma(n_j, \tilde{\delta}_j))^{\frac{1}{2\kappa-1}}\right). \tag{8.10}$$

We can write:

$$\begin{aligned}
\sup_{h \in V_j} \sigma(n_j, \tilde{\delta}_j) + \sqrt{\sigma(n_j, \tilde{\delta}_j)\rho_{S_j}(h, \hat{h}_j)} &\leq \sup_{h \in V_j} 3\sigma(n_j, \tilde{\delta}_j) + \sqrt{2\sigma(n_j, \tilde{\delta}_j)\rho_{\tilde{\Delta}}(h, \hat{h}_j)} \\
&\leq \sup_{h \in V_j} \max(6\sigma(n_j, \tilde{\delta}_j), 2\sqrt{2\sigma(n_j, \tilde{\delta}_j)\rho_{\tilde{\Delta}}(h, \hat{h}_j)}),
\end{aligned}$$

where the first inequality follows from Equation (8.13) and the second inequality follows $A + B \leq 2\max(A, B)$. We can further use Equation (8.10) to show that this is at most:

$$\begin{aligned}
&\leq \max\Big(6\sigma(n_j, \tilde{\delta}_j), (16C\sigma(n_j, \tilde{\delta}_j))^{\frac{1}{2}}(36\tilde{\epsilon})^{\frac{1}{2\kappa}}, (16C\sigma(n_j, \tilde{\delta}_j))^{\frac{1}{2}}(52\sigma(n_j, \tilde{\delta}_j))^{\frac{1}{2\kappa}}, \\
&\qquad (6400C\sigma(n_j, \tilde{\delta}_j))^{\frac{\kappa}{2\kappa-1}}\Big) \\
&\leq \max\left(6\sigma(n_j, \tilde{\delta}_j), \tilde{\epsilon}/6, (6400C\sigma(n_j, \tilde{\delta}_j))^{\frac{\kappa}{2\kappa-1}}\right)
\end{aligned}$$

Here the last inequality follows from that $(16C\sigma(n_j, \tilde{\delta}_j))^{\frac{1}{2}}(36\tilde{\epsilon})^{\frac{1}{2\kappa}} \leq \max((3456C\sigma(n_j, \tilde{\delta}_j))^{\frac{\kappa}{2\kappa-1}}, \frac{\tilde{\epsilon}}{6})$ and $(16C\sigma(n_j, \tilde{\delta}_j))^{\frac{1}{2}}(52\sigma(n_j, \tilde{\delta}_j))^{\frac{1}{2\kappa}} \leq \max((144C\sigma(n_j, \tilde{\delta}_j))^{\frac{\kappa}{2\kappa-1}}, 6\sigma(n_j, \tilde{\delta}_j))$, since $A^{\frac{2\kappa-1}{2\kappa}}B^{\frac{1}{2\kappa}} \leq \max(A, B)$.

It can be easily seen that there exists $c_2 > 0$, such that taking $j_1 = \lceil \log\frac{c_2}{2}(d\ln\frac{\max(C,1)}{\tilde{\epsilon}} + \ln\frac{1}{\delta})(C\tilde{\epsilon}^{\frac{1}{\kappa}-2} + \tilde{\epsilon}^{-1})\rceil$, so that $n_j \geq \frac{c_2}{2}(d\ln\frac{\max(C,1)}{\tilde{\epsilon}} + \ln\frac{1}{\delta})(C\tilde{\epsilon}^{\frac{1}{\kappa}-2} + \tilde{\epsilon}^{-1})$ suffices to make

$$\max\left(6\sigma(n_j, \tilde{\delta}_j), (6400C\sigma(n_j, \tilde{\delta}_j))^{\frac{\kappa}{2\kappa-1}}\right) \leq \tilde{\epsilon}/6$$

Hence the stopping criterion $\sup_{h \in V_j}\sqrt{\sigma(n_j, \tilde{\delta}_j)\rho_{S_j}(h, \hat{h}_j)} + \sigma(n_j, \tilde{\delta}_j) \leq \tilde{\epsilon}/6$ is satisfied

in iteration j_1. Thus the number of the exit iteration j_0 satisfies $j_0 \leq j_1$, and $n_{j_0} \leq n_{j_1} \leq c_2 \max\left((d\ln\frac{1}{\tilde{\epsilon}} + \ln\frac{1}{\delta})\tilde{\epsilon}^{-1}, (d\ln(C\tilde{\epsilon}^{\frac{1}{\kappa}-2}) + \ln\frac{1}{\delta})C\tilde{\epsilon}^{\frac{1}{\kappa}-2}\right)$.

□

Lemma 8.10. *Suppose there exist $C > 0$ and a classifier $\tilde{h} \in V$, such that Equation (8.9) holds. Suppose we draw a set S of n examples, denote the empirical risk minimizer over S as $\hat{h}$, then with probability $1-\delta$:*

$$\mathrm{err}_{\tilde{\Delta}}(\hat{h}) - \mathrm{err}_{\tilde{\Delta}}(\tilde{h}) \leq \max\left(2\sigma(n,\delta), (4C\sigma(n,\delta))^{\frac{\kappa}{2\kappa-1}}, 2\tilde{\epsilon}\right)$$

$$\rho_{\tilde{\Delta}}(\hat{h},\tilde{h}) \leq \max\left(C(2\sigma(n,\delta))^{\frac{1}{\kappa}}, C(4C\sigma(n,\delta))^{\frac{1}{2\kappa-1}}, C(2\tilde{\epsilon})^{\frac{1}{\kappa}}\right)$$

Proof. By Lemma A.5, with probability $1-\delta$, Equation (A.4) holds. Assume this happens.

$$\begin{aligned}
& \mathrm{err}_{\tilde{\Delta}}(\hat{h}) - \mathrm{err}_{\tilde{\Delta}}(\tilde{h}) \\
\leq\ & \sigma(n,\delta) + \sqrt{\sigma(n,\delta)\rho_{\tilde{\Delta}}(\hat{h},\tilde{h})} \\
\leq\ & 2\max\left(\sigma(n,\delta), \sqrt{\sigma(n,\delta)C(\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\tilde{h}))^{\frac{1}{\kappa}}}, \sqrt{\sigma(n,\delta)C\tilde{\epsilon}^{\frac{1}{\kappa}}}\right) \\
\leq\ & \max\left(2\sigma(n,\delta), (4C\sigma(n,\delta))^{\frac{\kappa}{2\kappa-1}}, 2\tilde{\epsilon}\right)
\end{aligned}$$

Where the first inequality is by Equation (A.4) of Lemma A.5; the second inequality follow from Equation (8.9) and $A + B \leq 2\max(A,B)$. The third inequality follows from $2\sqrt{\sigma(n,\delta)C\tilde{\epsilon}^{\frac{1}{\kappa}}} \leq \max\left(2(C\sigma(n,\delta))^{\frac{\kappa}{2\kappa-1}}, 2\tilde{\epsilon}\right)$, since $A^{\frac{2\kappa-1}{2\kappa}}B^{\frac{1}{2\kappa}} \leq \max(A,B)$. As a consequence, by Equation (8.9),

$$\rho_{\tilde{\Delta}}(\hat{h},\tilde{h}) \leq \max\left(C(2\sigma(n,\delta))^{\frac{1}{\kappa}}, C(4C\sigma(n,\delta))^{\frac{1}{2\kappa-1}}, C(2\tilde{\epsilon})^{\frac{1}{\kappa}}\right)$$

□

Lemma 8.11. *Suppose there exist a $C > 0$ and a classifier $\tilde{h} \in V$ such that Equation (8.9) holds. Suppose we draw a set S of n iid examples, and let $\hat{h}$ denote the empirical risk minimizer over S. Moreover, we define:*

$$\tilde{V} = \left(h \in V : \mathrm{err}_S(h) \leq \mathrm{err}_S(\hat{h}) + \frac{\tilde{\epsilon}}{2} + \sigma(n,\delta) + \sqrt{\sigma(n,\delta)\rho_S(h,\hat{h})}\right)$$

then with probability $1-\delta$*, for all* $h \in \tilde{V}$*,*

$$\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\tilde{h}) \leq \max\left(52\sigma(n,\delta), 36\tilde{\epsilon}, (6400C\sigma(n,\delta))^{\frac{\kappa}{2\kappa-1}}\right)$$

$$\rho_{\tilde{\Delta}}(h,\tilde{h}) \leq \max C(36\tilde{\epsilon})^{\frac{1}{\kappa}}, C(52\sigma(n,\delta))^{\frac{1}{\kappa}}, C(6400C\sigma(n,\delta))^{\frac{1}{2\kappa-1}}$$

Proof. First, by Lemma 8.10,

$$\mathrm{err}_{\tilde{\Delta}}(\hat{h}) - \mathrm{err}_{\tilde{\Delta}}(\tilde{h}) \leq \max\left(2\sigma(n,\delta), (4C\sigma(n,\delta))^{\frac{\kappa}{2\kappa-1}}, 2\tilde{\epsilon}\right) \tag{8.11}$$

$$\rho_{\tilde{\Delta}}(\hat{h},\tilde{h}) \leq \max\left(C(2\sigma(n,\delta))^{\frac{1}{\kappa}}, C(4C\sigma(n,\delta))^{\frac{1}{2\kappa-1}}, C(2\tilde{\epsilon})^{\frac{1}{\kappa}}\right) \tag{8.12}$$

Next, if $h \in \tilde{V}$, then

$$\mathrm{err}_S(h) - \mathrm{err}_S(\hat{h}) \leq \sigma(n,\delta) + \sqrt{\sigma(n,\delta)\rho_S(h,\hat{h})} + \frac{\tilde{\epsilon}}{2}$$

Combining it with Equation (A.4) of Lemma A.5, that is: $\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\hat{h}) \leq \mathrm{err}_S(h) - \mathrm{err}_S(\hat{h}) + \sqrt{\sigma(n,\delta)\rho_S(h,\hat{h})} + \sigma(n,\delta)$, we get

$$\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\hat{h}) \leq 2\sigma(n,\delta) + 2\sqrt{\sigma(n,\delta)\rho_S(h,\hat{h})} + \frac{\tilde{\epsilon}}{2}$$

By Equation (A.5) of Lemma A.5,

$$\rho_S(h,\hat{h}) \leq \rho_{\tilde{\Delta}}(h,\hat{h}) + \sqrt{\sigma(n,\delta)\rho_{\tilde{\Delta}}(h,\hat{h})} + \sigma(n,\delta) \leq 2\rho_{\tilde{\Delta}}(h,\hat{h}) + 2\sigma(n,\delta) \tag{8.13}$$

Therefore,

$$\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\hat{h}) \leq 5\sigma(n,\delta) + 3\sqrt{\sigma(n,\delta)\rho_{\tilde{\Delta}}(h,\hat{h})} + \frac{\tilde{\epsilon}}{2} \tag{8.14}$$

Hence

$$\begin{aligned}
& \mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\tilde{h}) \\
= \;& (\mathrm{err}_{\tilde{\Delta}}(h) - \mathrm{err}_{\tilde{\Delta}}(\hat{h})) + (\mathrm{err}_{\tilde{\Delta}}(\hat{h}) - \mathrm{err}_{\tilde{\Delta}}(\tilde{h})) \\
\leq \;& (4C\sigma(n,\delta))^{\frac{\kappa}{2\kappa-1}} + 7\sigma(n,\delta) + 3\tilde{\epsilon} + 3\sqrt{\sigma(n,\delta)\rho_{\tilde{\Delta}}(h,\hat{h})} \\
\leq \;& (4C\sigma(n,\delta))^{\frac{\kappa}{2\kappa-1}} + 7\sigma(n,\delta) + 3\tilde{\epsilon} + 3\sqrt{\sigma(n,\delta)\rho_{\tilde{\Delta}}(h,\tilde{h})} + 3\sqrt{\sigma(n,\delta)\rho_{\tilde{\Delta}}(\tilde{h},\hat{h})}
\end{aligned}$$

Here the first inequality follows from Equations (8.11) and (8.14) and $\max(A,B,C) \leq A+B+C$, and the second inequality follows from triangle inequality and $\sqrt{A+B} \leq \sqrt{A}+\sqrt{B}$.

From Equation (8.12), $\sigma(n,\delta)\rho_{\tilde{\Delta}}(\hat{h},\tilde{h})$ is at most:

$$\begin{aligned}
&\leq C\sigma(n,\delta)\cdot((2\tilde{\epsilon})^{1/\kappa}+(2\sigma(n,\delta))^{1/\kappa}+(4C\sigma(n,\delta))^{1/(2\kappa-1)}) \\
&\leq (4C\sigma(n,\delta))^{2\kappa/(2\kappa-1)}+C\sigma(n,\delta)((2\tilde{\epsilon})^{1/\kappa}+(2\sigma(n,\delta))^{1/\kappa}) \\
&\leq (4C\sigma(n,\delta))^{2\kappa/(2\kappa-1)}+\max\left(4\tilde{\epsilon}^2,(C\sigma(n,\delta))^{2\kappa/(2\kappa-1)}\right) \\
&\quad +\max\left(4\sigma(n,\delta)^2,(C\sigma(n,\delta))^{2\kappa/(2\kappa-1)}\right),
\end{aligned}$$

where the first step follows from Equation (8.12), the second step from algebra, and the third step from using the fact that $A^{\frac{2\kappa-1}{\kappa}}B^{\frac{1}{\kappa}} \leq \max(A^2,B^2)$. Plugging this in to the previous equation, and using $\max(A,B) \leq A+B$ and $\sqrt{A+B} \leq \sqrt{A}+\sqrt{B}$, we get that:

$$\mathrm{err}_{\tilde{\Delta}}(h)-\mathrm{err}_{\tilde{\Delta}}(\tilde{h}) \quad \leq \quad 10(4C\sigma(n,\delta))^{\kappa/(2\kappa-1)}+9\tilde{\epsilon}+13\sigma(n,\delta)+3\sqrt{\sigma(n,\delta)\rho_{\tilde{\Delta}}(h,\tilde{h})}$$

Combining this with the fact that $A+B+C+D \leq 4\max(A,B,C,D)$, we get that this is at most:

$$\leq \quad \max(40(4C\sigma(n,\delta))^{\kappa/(2\kappa-1)},36\tilde{\epsilon},52\sigma(n,\delta),12\sqrt{\sigma(n,\delta)\rho_{\tilde{\Delta}}(h,\tilde{h})})$$

Combining this with Condition (8.9), we get that this is at most:

$$\max(40(4C\sigma(n,\delta))^{\kappa/(2\kappa-1)},36\tilde{\epsilon},52\sigma(n,\delta),12\sqrt{C\sigma(n,\delta)\tilde{\epsilon}^{1/\kappa}},12\sqrt{C\sigma(n,\delta)(\mathrm{err}_{\tilde{\Delta}}(h)-\mathrm{err}_{\tilde{\Delta}}(\tilde{h}))^{1/\kappa}})$$

Using the elementary inequality that $A^{(2\kappa-1)/2\kappa}B^{1/2\kappa} \leq \max(A,B)$, we conclude that $\sqrt{C\sigma(n,\delta)\tilde{\epsilon}^{1/\kappa}} \leq \max\left(\tilde{\epsilon},(C\sigma(n,\delta))^{\kappa/(2\kappa-1)}\right)$. Also, we note that the inequality $\mathrm{err}_{\tilde{\Delta}}(h)-\mathrm{err}_{\tilde{\Delta}}(\tilde{h}) \leq 12\sqrt{C\sigma(n,\delta)(\mathrm{err}_{\tilde{\Delta}}(h)-\mathrm{err}_{\tilde{\Delta}}(\tilde{h}))^{1/\kappa}}$ implies $\mathrm{err}_{\tilde{\Delta}}(h)-\mathrm{err}_{\tilde{\Delta}}(\tilde{h}) \leq (144C\sigma(n,\delta))^{\kappa/(2\kappa-1)}$. Thus we have

$$\mathrm{err}_{\tilde{\Delta}}(h)-\mathrm{err}_{\tilde{\Delta}}(\tilde{h}) \leq \max\left(36\tilde{\epsilon},52\sigma(n,\delta),(6400C\sigma(n,\delta))^{\frac{\kappa}{2\kappa-1}}\right)$$

Invoking (8.9) again, we have that:

$$\rho_{\tilde{\Delta}}(h,\tilde{h}) \leq \max\left(C(36\tilde{\epsilon})^{\frac{1}{\kappa}},C(52\sigma(n,\delta))^{\frac{1}{\kappa}},C(6400C\sigma(n,\delta))^{\frac{1}{2\kappa-1}}\right)$$

□

8.6 Remaining Proofs from Section 8.2

Proof of Lemma 8.1. Assuming E_r happens, we prove the lemma by induction.

Base Case: For $k = 1$, clearly $h^*(D) \in V_1 = \mathcal{H}$.

Inductive Case: Assume $h^*(D) \in V_k$. As we are in the realizable case, $h^*(D)$ is consistent with the examples S_k drawn in Step 8 of Algorithm 8.1; thus $h^*(D) \in V_{k+1}$. The lemma follows. □

Proof Of Lemma 8.2. We use $\tilde{h}_k = \arg\min_{h \in V_k} \mathrm{err}_{\tilde{\Gamma}_k}(h)$ to denote the optimal classifier in V_k with respect to the distribution $\tilde{\Gamma}_k$. Assuming E_a happens, we prove the lemma by induction.

Base Case: For $k = 1$, clearly $h^*(D) \in V_1 = \mathcal{H}$.

Inductive Case: Assume $h^* \in V_k$. In order to show the inductive case, our goal is to show that:

$$\mathbb{P}_{\tilde{\Gamma}_k}[h^*(D)(x) \neq y] - \mathbb{P}_{\tilde{\Gamma}_k}[\tilde{h}_k(x) \neq y] \leq \frac{\epsilon_k}{16\phi_k} \tag{8.15}$$

If (8.15) holds, then, by (2.1) of Lemma 8.4, we know that if Algorithm 8.2 succeeds when called in iteration k of Algorithm 8.1, then, it is guaranteed that $h^* \in V_{k+1}$.

We therefore focus on showing (8.15). First, from Equation (8.2) of Lemma 8.7, we have:

$$(\mathrm{err}_{\tilde{U}_k}(h^*(D)) - \mathrm{err}_{\tilde{U}_k}(\tilde{h}_k)) - (\mathrm{err}_D(h^*(D)) - \mathrm{err}_D(\tilde{h}_k)) \leq \frac{\epsilon_k}{32}$$

As $\mathrm{err}_D(h^*(D)) \leq \mathrm{err}_D(\tilde{h}_k)$, we get:

$$\mathrm{err}_{\tilde{U}_k}(h^*(D)) \leq \mathrm{err}_{\tilde{U}_k}(\tilde{h}_k) + \frac{\epsilon_k}{32} \tag{8.16}$$

On the other hand, by Equation (8.4) of Lemma 8.8 and triangle inequality,

$$\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(\tilde{h}_k(x) \neq y)(1-\gamma_k(x))] - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)(1-\gamma_k(x))] \tag{8.17}$$

$$\leq \quad \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq \tilde{h}_k(x))(1-\gamma_k(x))] \leq \frac{\epsilon_k}{32} \tag{8.18}$$

Combining Equations (8.16) and (8.17), we get:

$$\begin{aligned}
\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)\gamma_k(x)] &= \mathrm{err}_{\tilde{U}_k}(h^*(D)(x)) - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)(1-\gamma_k(x))] \\
&\leq \mathrm{err}_{\tilde{U}_k}(\tilde{h}_k(x)) + \epsilon_k/32 - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)(1-\gamma_k(x))] \\
&\leq \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(\tilde{h}_k(x) \neq y)\gamma_k(x)] + \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(\tilde{h}(x) \neq y)(1-\gamma_k(x))] + \epsilon_k/32 \\
&\quad -\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)(1-\gamma_k(x))] \\
&\leq \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(\tilde{h}_k(x) \neq y)\gamma_k(x)] + \epsilon_k/16
\end{aligned}$$

Dividing both sides by ϕ_k, we get:

$$\mathbb{P}_{\tilde{\Gamma}_k}[h^*(D)(x) \neq y] - \mathbb{P}_{\tilde{\Gamma}_k}[\tilde{h}_k(x) \neq y] \leq \frac{\epsilon_k}{16\phi_k},$$

from which the lemma follows. □

Proof of Lemma 8.3. Assuming E_r happens, we prove the lemma by induction.

Base Case: For $k = 1$, clearly $\mathrm{err}_D(h) \leq 1 \leq \epsilon_1 = \epsilon 2^{k_0}, \forall h \in V_1 = \mathcal{H}$.

Inductive Case: Note that $\forall h, h' \in V_{k+1} \subseteq V_k$, by Equation (8.4) of Lemma 8.8, we have:

$$\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq h'(x))(1-\gamma_k(x))] \leq \frac{\epsilon_k}{8}$$

By the proof of Lemma 8.1, $h^*(D) \in V_{k+1}$ on event E_r, thus $\forall h \in V_{k+1}$,

$$\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq h^*(D)(x))(1-\gamma_k(x))] \leq \frac{\epsilon_k}{8} \tag{8.19}$$

Since any $h \in V_{k+1}$, h is consistent with S_k of size $m_k = \frac{768\phi_k}{\epsilon_k}\left(d\ln\frac{768\phi_k}{\epsilon_k} + \ln\frac{48}{\delta_k}\right)$, we have that for all $h \in V_{k+1}$,

$$\mathbb{P}_{\tilde{\Gamma}_k}[h(x) \neq h^*(D)(x)] \leq \frac{\epsilon_k}{8\phi_k}$$

That is,

$$\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq h^*(D)(x))\gamma_k(x)] \leq \frac{\epsilon_k}{8}$$

Combining this with Equation (8.19) above,

$$\mathbb{P}_{\tilde{U}_k}[h(x) \neq h^*(D)(x)] \leq \frac{\epsilon_k}{4}$$

By Equation (8.1) of Lemma 8.7,

$$\mathbb{P}_D[h(x) \neq h^*(D)(x)] \leq \frac{\epsilon_k}{2} = \epsilon_{k+1}$$

The lemma follows. □

Proof of Lemma 8.6. Assuming E_a happens, we prove the lemma by induction.

Base Case: For $k = 1$, clearly $\mathrm{err}_D(h) - \mathrm{err}_D(h^*(D)) \leq 1 \leq \epsilon_1 = \epsilon 2^{k_0}, \forall h \in V_1 = \mathcal{H}$.

Inductive Case: Note that $\forall h, h' \in V_{k+1} \subseteq V_k$, by Equation (8.4) of Lemma 8.8,

$$\begin{aligned} & \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq y)(1-\gamma_k(x))] - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h'(D)(x) \neq y)(1-\gamma_k(x))] \\ \leq \quad & \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq h'(D)(x))(1-\gamma_k(x))] \leq \frac{\epsilon_k}{8} \end{aligned}$$

From Lemma 8.2, $h^*(D) \in V_k$ whenever the event E_a happens. Thus $\forall h \in V_{k+1}$,

$$\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq y)(1-\gamma_k(x))] - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)(1-\gamma_k(x))] \leq \frac{\epsilon_k}{8} \tag{8.20}$$

On the other hand, if Algorithm 8.2 succeeds with target excess error $\frac{\epsilon_k}{8\phi_k}$, by item(2.2) of Lemma 8.4, for any $h \in V_{k+1}$,

$$\mathbb{P}_{\tilde{\Gamma}_k}[h(x) \neq y] - \min_{h \in V_k} \mathbb{P}_{\tilde{\Gamma}_k}[h(x) \neq y] \leq \frac{\epsilon_k}{8\phi_k}$$

Moreover, as $h^*(D) \in V_k$ from Lemma 8.2,

$$\mathbb{P}_{\tilde{\Gamma}_k}[h(x) \neq y] - \mathbb{P}_{\tilde{\Gamma}_k}[h^*(D)(x) \neq y] \leq \frac{\epsilon_k}{8\phi_k}$$

In other words,

$$\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq y)\gamma_k(x)] - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)\gamma_k(x)] \leq \frac{\epsilon_k}{8}$$

Combining this with Equation (8.20), we get that for all $h \in V_{k+1}$,

$$\mathbb{P}_{\tilde{U}_k}[h(x) \neq y] - \mathbb{P}_{\tilde{U}_k}[h^*(D)(x) \neq y] \leq \frac{\epsilon_k}{4}$$

Finally, combining this with Equation (8.2) of Lemma 8.7, we have that:

$$\mathbb{P}_D[h(x) \neq y] - \mathbb{P}_D[h^*(D)(x) \neq y] \leq \frac{\epsilon_k}{2} = \epsilon_{k+1}$$

The lemma follows. □

8.7 Proofs from Section 8.3

Proof of Theorem 8.2. (1) In the realizable case, suppose that event E_r happens. Then from Equation (8.5) of Lemma 8.8, while running Algorithm 6.1, we have that:

$$\phi_k \leq \mathbf{\Phi}_D(V_k, \frac{\epsilon_k}{128}) + \frac{\epsilon_k}{256} \leq \mathbf{\Phi}_D\left(\mathrm{B}_D(h^*, \epsilon_k), \frac{\epsilon_k}{128}\right) + \frac{\epsilon_k}{256} \leq \mathbf{\Phi}_D\left(\mathrm{B}_D(h^*, \epsilon_k), \frac{\epsilon_k}{256}\right) = \phi(\epsilon_k, \frac{\epsilon_k}{256})$$

where the second inequality follows from the fact that $V_k \subseteq \mathrm{B}_D(h^*(D), \epsilon_k)$, and third inequality follows from Lemma 8.15 and denseness assuption.
Thus, there exists $c_3 > 0$ such that, in round k,

$$m_k = \left(d \ln \frac{768\phi_k}{\epsilon_k} + \ln \frac{48}{\delta_k}\right) \frac{768\phi_k}{\epsilon_k} \leq c_3 \left(d \ln \frac{\phi(\epsilon_k, \epsilon_k/256)}{\epsilon_k} + \ln(\frac{k_0 - k + 1}{\delta})\right) \frac{\phi(\epsilon_k, \epsilon_k/256)}{\epsilon_k}$$

Hence the total number of labels queried by Algorithm 8.1 can be bounded as follows:

$$\begin{aligned} & \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} m_k \\ \leq \quad & c_3 \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} \left(d \ln \frac{\phi(\epsilon_k, \epsilon_k/256)}{\epsilon_k} + \ln(\frac{k_0 - k + 1}{\delta})\right) \frac{\phi(\epsilon_k, \epsilon_k/256)}{\epsilon_k} \end{aligned}$$

(2) In the agnostic case, suppose the event E_a happens.

First, given E_a, from Equation (8.5) of Lemma 8.8 when running Algorithm 6.1,

$$\phi_k \leq \mathbf{\Phi}_D(V_k, \frac{\epsilon_k}{128}) + \frac{\epsilon_k}{256} \leq \mathbf{\Phi}_D\left(\mathrm{B}_D(h^*, 2\nu^*(D) + \epsilon_k), \frac{\epsilon_k}{256}\right) = \phi(2\nu^*(D) + \epsilon_k, \frac{\epsilon_k}{256}) \qquad (8.21)$$

where the second inequality follows from the fact that $V_k \subseteq \mathrm{B}_D(h^*(D), 2\nu^*(D) + \epsilon_k)$ and the third inequality follows from Lemma 8.15 and denseness assumption.

Second, recall that $\tilde{h}_k = \arg\min_{h \in V_k} \mathrm{err}_{\tilde{\Gamma}_k}(h)$,

$$\begin{aligned}
\mathrm{err}_{\tilde{\Gamma}_k}(\tilde{h}_k) &= \min_{h \in V_k} \mathrm{err}_{\tilde{\Gamma}_k}(h) \\
&\leq \mathrm{err}_{\tilde{\Gamma}_k}(h^*(D)) \\
&= \frac{\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)\gamma_k(x)]}{\phi_k} \\
&\leq \frac{\mathbb{P}_{\tilde{U}_k}[h^*(D)(x) \neq y]}{\phi_k} \\
&\leq \frac{\nu^*(D) + \epsilon_k/64}{\phi_k}
\end{aligned}$$

Here the first inequality follows from the suboptimality of $h^*(D)$ under distribution $\tilde{\Gamma}_k$, the second inequality follows from $\gamma_k(x) \leq 1$, and the third inequality follows from Equation (8.1). Thus, conditioned on E_a, in iteration k, Algorithm 8.2 succeeds by Lemma 8.5, and there exists a constant $c_4 > 0$ such that the number of labels queried is

$$\begin{aligned}
& m_k \leq c_1 \frac{\frac{\epsilon_k}{8\phi_k} + \mathrm{err}_{\tilde{\Gamma}_k}(\tilde{h}_k)}{(\frac{\epsilon_k}{8\phi_k})^2} \left(d \ln \frac{1}{\frac{\epsilon_k}{8\phi_k}} + \ln \frac{2}{\delta_k} \right) \\
\leq \quad & c_4 \left(d \ln \frac{\phi(2\nu^*(D) + \epsilon_k, \epsilon_k/256)}{\epsilon_k} + \ln(\frac{k_0 - k + 1}{\delta}) \right) \frac{\phi(2\nu^*(D) + \epsilon_k, \epsilon_k/256)}{\epsilon_k} (1 + \frac{\nu^*(D)}{\epsilon_k})
\end{aligned}$$

Here the last line follows from Equation (8.21). Hence the total number of examples queried can be bounded as follows:

$$\begin{aligned}
& \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} m_k \\
\leq \quad & c_4 \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} \left(d \ln \frac{\phi(2\nu^*(D) + \epsilon_k, \epsilon_k/256)}{\epsilon_k} + \ln(\frac{k_0 - k + 1}{\delta}) \right) \frac{\phi(2\nu^*(D) + \epsilon_k, \epsilon_k/256)}{\epsilon_k} (1 + \frac{\nu^*(D)}{\epsilon_k})
\end{aligned}$$

□

Proof of Theorem 8.3. Assume E_a happens.

First, from Equation (8.5) of Lemma 8.8 when running Algorithm 6.1,

$$\begin{aligned}\phi_k &\leq \mathbf{\Phi}_D(V_k, \frac{\epsilon_k}{128}) + \frac{\epsilon_k}{256} \leq \mathbf{\Phi}_D\left(\mathrm{B}_D(h^*, C_0\epsilon_k^{\frac{1}{\kappa}}), \frac{\epsilon_k}{128}\right) + \frac{\epsilon_k}{256} \\ &\leq \mathbf{\Phi}_D\left(\mathrm{B}_D(h^*, C_0\epsilon_k^{\frac{1}{\kappa}}), \frac{\epsilon_k}{256}\right) = \phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256}) \end{aligned} \tag{8.22}$$

where the second inequality follows from the fact that $V_k \subseteq \mathrm{B}_D(h^*(D), C_0\epsilon_k^{\frac{1}{\kappa}})$, and the third inequality follows from Lemma 8.15 and denseness assumption.

Second, for all $h \in V_k$,

$$\begin{aligned}
& \phi_k \rho_{\tilde{\Gamma}_k}(h, h^*(D)) \\
= \;& \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq h^*(D)(x))\gamma_k(x)] \\
\leq \;& \rho_{\tilde{U}_k}(h, h^*(D)) \\
\leq \;& \rho_D(h, h^*(D)) + \epsilon_k/32 \\
\leq \;& C_0(\mathrm{err}_D(h) - \mathrm{err}_D(h^*(D)))^{\frac{1}{\kappa}} + \epsilon_k/32 \\
\leq \;& C_0(\mathrm{err}_{\tilde{U}_k}(h) - \mathrm{err}_{\tilde{U}_k}(h^*(D)) + \epsilon_k/64)^{\frac{1}{\kappa}} + \epsilon_k/32 \\
= \;& C_0(\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq y)\gamma_k(x)] - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)\gamma_k(x)] \\
& + \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq y)(1-\gamma_k(x))] - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)(1-\gamma_k(x))] + \epsilon_k/16)^{\frac{1}{\kappa}} + \epsilon_k/32
\end{aligned}$$

Here the first inequality follows from $\gamma_k(x) \leq 1$, the second inequality follows from Equation (8.3) of Lemma 8.7, the third inequality follows from Definition 8.1 and the fourth inequality follows from Equation (8.2) of Lemma 8.7. The above can be upper bounded by:

$$\begin{aligned}
\leq \;& C_0(\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq y)\gamma_k(x)] - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)\gamma_k(x)] + \epsilon_k/16)^{\frac{1}{\kappa}} + \epsilon_k/32 \\
\leq \;& 2C_0(\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq y)\gamma_k(x)] - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)\gamma_k(x)])^{\frac{1}{\kappa}} + 2C_0(\epsilon_k/16)^{\frac{1}{\kappa}} + \epsilon_k/32 \\
\leq \;& \max(8C_0, 4)\max\left((\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq y)\gamma_k(x)] - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)\gamma_k(x)]), \frac{\epsilon_k}{16}\right)^{\frac{1}{\kappa}} \\
= \;& \max(8C_0, 4)(\phi_k)^{\frac{1}{\kappa}}\max\left(\mathbb{P}_{\tilde{\Gamma}_k}[h(x) \neq y] - \mathbb{P}_{\tilde{\Gamma}_k}[h^*(D)(x) \neq y], \frac{\epsilon_k}{8\phi_k}\right)^{\frac{1}{\kappa}}
\end{aligned}$$

Here the first inequality follows from Equation (8.4) of Lemma 8.8 and triangle inequality $\mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq y)\gamma_k(x)] - \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h^*(D)(x) \neq y)\gamma_k(x)] \leq \mathbb{E}_{\tilde{U}_k}[\mathbb{1}(h(x) \neq h^*(D)(x))\gamma_k(x)] \leq \epsilon_k/32$, and the last two inequalities follow from simple algebra.

Dividing both sides by ϕ_k, we get:

$$\rho_{\tilde{\Gamma}_k}(h, h^*(D)) \leq C_1(\phi_k)^{\frac{1}{\kappa}-1} \max\left(\mathrm{err}_{\tilde{\Gamma}_k}(h) - \mathrm{err}_{\tilde{\Gamma}_k}(h^*(D)), \frac{\epsilon_k}{8\phi_k}\right)^{\frac{1}{\kappa}}$$

where $C_1 = \max(8C_0, 4)$. Thus in iteration k, Condition (8.9) in Lemma 8.9 holds with $C := C_1(\phi_k)^{\frac{1}{\kappa}-1}$ and $\tilde{h} := h^*(D)$. Thus, from Lemma 8.9, Algorithm 8.2 succeeds, and there exists a constant $c_5 > 0$, such that the number of labels queried is

$$\begin{aligned}
m_k &\leq c_2 \max\Bigg((d\ln(C_1(\phi_k)^{\frac{1}{\kappa}-1}(\frac{\epsilon_k}{8\phi_k})^{\frac{1}{\kappa}-2}) + \ln\frac{2}{\delta_k})(C_1(\phi_k)^{\frac{1}{\kappa}-1}(\frac{\epsilon_k}{8\phi_k})^{\frac{1}{\kappa}-2}), \\
&\quad (d\ln(\frac{\epsilon_k}{8\phi_k})^{-1} + \ln\frac{2}{\delta_k})(\frac{\epsilon_k}{8\phi_k})^{-1}\Bigg) \\
&\leq c_5\left(d\ln(\phi_k\epsilon_k^{\frac{1}{\kappa}-2}) + \ln(\frac{k_0-k+1}{\delta})\right)\phi_k\epsilon_k^{\frac{1}{\kappa}-2} \\
&\leq c_5\left(d\ln(\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})\epsilon_k^{\frac{1}{\kappa}-2}) + \ln(\frac{k_0-k+1}{\delta})\right)\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})\epsilon_k^{\frac{1}{\kappa}-2}
\end{aligned}$$

Where the last line follows from Equation (8.21). Hence the total number of examples queried is at most

$$\sum_{k=1}^{\lceil \log\frac{1}{\epsilon}\rceil} m_k \leq c_5 \sum_{k=1}^{\lceil \log\frac{1}{\epsilon}\rceil}\left(d\ln(\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})\epsilon_k^{\frac{1}{\kappa}-2}) + \ln(\frac{k_0-k+1}{\delta})\right)\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})\epsilon_k^{\frac{1}{\kappa}-2}$$

□

The following lemma is an immediate corollary of Theorem 21, item (a) of Lemma 2 and Lemma 3 of [BL13]:

Lemma 8.12. *Suppose D is isotropic and log-concave on $\mathbb{R}^d$, and $\mathcal{H}$ is the set of homogeneous linear classifiers on $\mathbb{R}^d$, then there exist absolute constants $c_6, c_7 > 0$ such that $\phi(r, \eta) \leq c_6 r \ln\frac{c_7 r}{\eta}$.*

Proof of Lemma 8.12. Denote w_h as the unit vector w such that $h(x) = \mathrm{sign}(w \cdot x)$, and $\theta(w, w')$ to be the angle between vectors w and w'. If $h \in \mathrm{B}_D(h^*, r)$, then by Lemma 3 of [BL13], there exists some constant $c_{11} > 0$ such that $\theta(w_h, w_{h^*}) \leq \frac{r}{c_{11}}$. Also, by Lemma 21 of [BL13], there

exists some constants $c_{12}, c_{13} > 0$, such that, if $\theta(w, w') = \alpha$ then

$$\mathbb{P}_D(\text{sign}(w \cdot x) \neq \text{sign}(w' \cdot x), |w \cdot x| \geq b) \leq c_{12}\alpha \exp(-c_{13}\frac{b}{\alpha})$$

We define a special solution (α, β, γ) as follows:

$$\alpha(x) := \mathbb{1}(w_{h^*} \cdot x \geq \frac{r}{c_{11}c_{13}} \ln \frac{c_{12}r}{c_{11}\eta})$$

$$\beta(x) := \mathbb{1}(w_{h^*} \cdot x \leq -\frac{r}{c_{11}c_{13}} \ln \frac{c_{12}r}{c_{11}\eta})$$

$$\gamma(x) := \mathbb{1}(|w_{h^*} \cdot x| \leq \frac{r}{c_{11}c_{13}} \ln \frac{c_{12}r}{c_{11}\eta})$$

Then it can be checked that for all $h \in \mathrm{B}_D(h^*, r)$,

$$\begin{aligned} & \mathbb{E}[\mathbb{1}(h(x) = +1)\beta(x) + \mathbb{1}(h(x) = -1)\alpha(x)] \\ = \quad & \mathbb{P}_D[\text{sign}(w_{h^*} \cdot x) \neq \text{sign}(w_h \cdot x), |w_{h^*} \cdot x| \geq \frac{r}{c_{11}c_{13}} \ln \frac{c_{12}r}{c_{11}\eta}] \leq \eta \end{aligned}$$

And by item (a) of Lemma 2 of [BL13], we have

$$\mathbb{E}\gamma(x) = \mathbb{P}_D(|w_{h^*} \cdot x| \leq \frac{r}{c_{11}c_{13}} \ln \frac{c_{12}r}{c_{11}\eta}) \leq \frac{r}{c_{11}c_{13}} \ln \frac{c_{12}r}{c_{11}\eta}$$

Hence,

$$\phi(r, \eta) \leq \frac{r}{c_{11}c_{13}} \ln \frac{c_{12}r}{c_{11}\eta}$$

□

Proof of Corollary 8.1. This is an immediate consequence of Lemma 8.12 and Theorems 4.2 and 8.3 and algebra. □

8.8 Proofs of Concentration Lemmas

Proof of Lemma 8.7. We begin by observing that:

$$\text{err}_{\tilde{U}_k}(h) = \frac{1}{n_k} \sum_{i=1}^{n_k} [\mathbb{P}_D[Y = +1 | X = x_i]\mathbb{1}(h(x_i) = -1) + \mathbb{P}_D[Y = -1 | X = x_i]\mathbb{1}(h(x_i) = +1)]$$

Moreover, $\max(\mathcal{S}(\{\mathbb{1}(h(x)=1, h\in\mathcal{H})\}, n), \mathcal{S}(\{\mathbb{1}(h(x)=-1, h\in\mathcal{H})\}, n)) \leq (\frac{en}{d})^d$. Combining this fact with Lemma 8.13, the following equations hold simultaneously with probability $1-\delta_k/6$:

$$\Big|\frac{1}{n_k}\sum_{i=1}^{n_k}\mathbb{P}_D[Y=+1|X=x_i]\mathbb{1}(h(x_i)=-1) - \mathbb{P}_D[h(x)=-1, y=+1]\Big| \leq \sqrt{\frac{8(d\ln\frac{en_k}{d}+\ln\frac{24}{\delta_k})}{n_k}} \leq \frac{\epsilon_k}{128}$$

$$\Big|\frac{1}{n_k}\sum_{i=1}^{n_k}\mathbb{P}_D[Y=-1|X=x_i]\mathbb{1}(h(x_i)=+1) - \mathbb{P}_D[h(x)=+1, y=-1]\Big| \leq \sqrt{\frac{8(d\ln\frac{en_k}{d}+\ln\frac{24}{\delta_k})}{n_k}} \leq \frac{\epsilon_k}{128}$$

Thus Equation (8.1) holds with probability $1-\delta_k/6$. Moreover, we observe that Equation (8.1) implies Equation (8.2). To show Equation (8.3), we observe that by Lemma A.6, with probability $1-\delta_k/12$,

$$|\rho_D(h,h') - \rho_{\tilde{U}_k}(h,h')| = |\rho_D(h,h') - \rho_{S_k}(h,h')| \leq 2\sqrt{\sigma(n_k, \delta_k/12)} \leq \frac{\epsilon_k}{64}$$

Thus, Equation (8.3) holds with probability $\geq 1-\delta_k/12$. By union bound, with probability $1-\delta_k/4$, Equations (8.1), (8.2), and (8.3) hold simultaneously. □

Proof of Lemma 8.8. (1) Given a confidence-rated predictor with inputs hypothesis set V_k, unlabeled data U_k, and error bound $\epsilon_k/64$, the outputs $\{(\alpha_{k,i}, \beta_{k,i}, \gamma_{k,i})\}_{i=1}^{n_k}$ must satisfy that for all $h, h' \in V_k$,

$$\frac{1}{n_k}\sum_{i=1}^{n_k}[\mathbb{1}(h(x_{k,i})=-1)\alpha_{k,i} + \mathbb{1}(h(x_{k,i})=+1)\beta_{k,i}] \leq \frac{\epsilon_k}{64}$$

$$\frac{1}{n_k}\sum_{i=1}^{n_k}[\mathbb{1}(h'(x_{k,i})=-1)\alpha_{k,i} + \mathbb{1}(h'(x_{k,i})=+1)\beta_{k,i}] \leq \frac{\epsilon_k}{64}$$

Since $\mathbb{1}(h(x)\neq h'(x)) \leq \min(\mathbb{1}(h(x)=-1)+\mathbb{1}(h'(x)=-1), \mathbb{1}(h(x)=+1)+\mathbb{1}(h'(x)=+1))$, adding up the two inequalities above, we get

$$\frac{1}{n_k}\sum_{i=1}^{n_k}[\mathbb{1}(h(x_{k,i})\neq h'(x_{k,i}))(\alpha_{k,i}+\beta_{k,i})] \leq \frac{\epsilon_k}{32}$$

That is,

$$\frac{1}{n_k}\sum_{i=1}^{n_k}[\mathbb{1}(h(x_{k,i})\neq h'(x_{k,i}))(1-\gamma_{k,i})] \leq \frac{\epsilon_k}{32}$$

(2) By definition of $\mathbf{\Phi}_D(V,\eta)$, there exist nonnegative functions α, β, γ such that $\alpha(x)+$

$\beta(x)+\gamma(x)\equiv 1$, $\mathbb{E}_D[\gamma(x)] = \mathbf{\Phi}_D(V_k,\epsilon_k/128)$ and for all $h\in V_k$,

$$\mathbb{E}_D[\alpha(x)\mathbb{1}(h(x)=-1)+\beta(x)\mathbb{1}(h(x)=+1)] \le \frac{\epsilon_k}{128}$$

Consider the linear program in Algorithm 6.1 with inputs hypothesis set V_k, unlabeled data U_k, and error bound $\epsilon_k/64$. We consider the following special (but possibly non-optimal) solution for this LP: $\alpha_{k,i}=\alpha(z_{k,i}),\beta_{k,i}=\beta(z_{k,i}),\gamma_{k,i}=\gamma(z_{k,i})$. We will now show that this solution is feasible and has abstention $\mathbf{\Phi}_D(V_k,\epsilon_k/128)$ plus $O(\epsilon_k)$ with high probability.
Observe that $\max(\mathcal{S}(\{\mathbb{1}(h(x)=1,h\in\mathcal{H})\},n),\mathcal{S}(\{\mathbb{1}(h(x)=-1,h\in\mathcal{H})\},n)) \le (\frac{en}{d})^d$. Therefore, from Lemma 8.13 and the union bound, with probability $1-\delta_k/4$, the following hold simultaneously for all $h\in\mathcal{H}$:

$$\Big|\frac{1}{n_k}\sum_{i=1}^{n_k}\gamma(z_{k,i})-\mathbb{E}_D\gamma(x)\Big| \le \sqrt{\frac{\ln\frac{2}{\delta_k}}{2n_k}} \le \frac{\epsilon_k}{256} \tag{8.23}$$

$$\Big|\frac{1}{n_k}\sum_{i=1}^{n_k}\alpha(z_{k,i})\mathbb{1}(h(z_{k,i})=-1)-\mathbb{E}_D[\alpha(x)\mathbb{1}(h(x)=-1)]\Big| \le \sqrt{\frac{8(d\ln\frac{en_k}{d}+\ln\frac{24}{\delta_k})}{n_k}} \le \frac{\epsilon_k}{256} \tag{8.24}$$

$$\Big|\frac{1}{n_k}\sum_{i=1}^{n_k}\beta(z_{k,i})\mathbb{1}(h(z_{k,i})=+1)-\mathbb{E}_D[\beta(x)\mathbb{1}(h(x)=+1)]\Big| \le \sqrt{\frac{8(d\ln\frac{en_k}{d}+\ln\frac{24}{\delta_k})}{n_k}} \le \frac{\epsilon_k}{256} \tag{8.25}$$

Adding up Equations (8.24) and (8.25), we get that

$$\Big|\frac{1}{n_k}\sum_{i=1}^{n_k}[\beta(x_i)\mathbb{1}(h(x_i)=+1)+\alpha(x_i)\mathbb{1}(h(x_i)=-1)]-\mathbb{E}_D[\alpha(x)\mathbb{1}(h(x)=-1)+\beta(x)\mathbb{1}(h(x)=+1))]\Big|$$

is at most $\frac{\epsilon_k}{128}$. Thus $\{(\alpha(z_{k,i}),\beta(z_{k,i})\}_{i=1}^{n_k}$ is a feasible solution of the linear program of Algorithm 6.1. Also, by Equation (8.23), $\frac{1}{n_k}\sum_{i=1}^{n_k}\gamma(z_{k,i}) \le \mathbf{\Phi}_D(V_k,\frac{\epsilon_k}{128})+\frac{\epsilon_k}{64}$. Thus, by Theorem 6.1, the outputs $\{(\alpha_{k,i},\beta_{k,i},\gamma_{k,i})\}_{i=1}^{n_k}$ of the linear program in Algorithm 6.1 satisfy

$$\phi_k = \frac{1}{n_k}\sum_{i=1}^{n_k}\gamma_{k,i} \le \frac{1}{n_k}\sum_{i=1}^{n_k}\gamma(z_{k,i}) \le \mathbf{\Phi}_D(V_k,\frac{\epsilon_k}{128})+\frac{\epsilon_k}{256}$$

due to their optimality. □

Lemma 8.13. *Pick any $n\ge 1$, $\delta\in(0,1)$, a family $\mathcal{F}$ of functions $f:\mathcal{Z}\to\{0,1\}$, a fixed weighting function $w:\mathcal{Z}\to[0,1]$. Let S_n be a set of n iid copies of Z. The following holds with probability*

at least $1-\delta$*:*

$$\Big|\frac{1}{n}\sum_{i=1}^{n} w(z_i)f(z_i) - \mathbb{E}[w(z)f(z)]\Big| \leq \sqrt{\frac{8(\ln \mathcal{S}(\mathcal{F},n) + \ln\frac{2}{\delta})}{n}}$$

where $\mathcal{S}(\mathcal{F},n) = \max_{z_1,\ldots,z_n\in\mathcal{Z}} |\{(f(z_1),\ldots,f(z_n)) : f\in\mathcal{F}\}|$ *is the growth function of* $\mathcal{F}$*.*

Proof. The proof is fairly standard, and follows immediately from the proof of additive VC bounds. With probability $1-\delta$,

$$\begin{aligned}
& \sup_{f\in\mathcal{F}} \Big|\frac{1}{n}\sum_{i=1}^{n} w(z_i)f(z_i) - \mathbb{E}w(z)f(z)\Big| \\
\leq\ & \mathbb{E}_{S\sim D^n} \sup_{f\in\mathcal{F}} \Big|\frac{1}{n}\sum_{i=1}^{n} w(z_i)f(z_i) - \mathbb{E}w(z)f(z)\Big| + \sqrt{\frac{2\ln\frac{1}{\delta}}{n}} \\
\leq\ & \mathbb{E}_{S\sim D^n, S'\sim D^n} \sup_{f\in\mathcal{F}} \Big|\frac{1}{n}\sum_{i=1}^{n} (w(z_i)f(z_i) - w(z_i')f(z_i'))\Big| + \sqrt{\frac{2\ln\frac{1}{\delta}}{n}} \\
\leq\ & \mathbb{E}_{S\sim D^n, S'\sim D^n, \sigma\sim U(\{-1,+1\}^n)} \sup_{f\in\mathcal{F}} \Big|\frac{1}{n}\sum_{i=1}^{n} \sigma_i(w(z_i)f(z_i) - w(z_i')f(z_i'))\Big| + \sqrt{\frac{2\ln\frac{1}{\delta}}{n}} \\
\leq\ & 2\mathbb{E}_{S\sim D^n, \sigma\sim U(\{-1,+1\}^n)} \sup_{f\in\mathcal{F}} \Big|\frac{1}{n}\sum_{i=1}^{n} \sigma_i w(z_i)f(z_i)\Big| + \sqrt{\frac{2\ln\frac{1}{\delta}}{n}} \\
\leq\ & 2\sqrt{\frac{2\ln(2\mathcal{S}(\mathcal{F},n))}{n}} + \sqrt{\frac{2\ln\frac{1}{\delta}}{n}} \leq \sqrt{\frac{8(\ln\mathcal{S}(\mathcal{F},n) + \ln\frac{2}{\delta})}{n}}
\end{aligned}$$

Where the first inequality is by McDiarmid's Lemma; the second inequality follows from Jensen's Inequality; the third inequality follows from symmetry; the fourth inequality follows from $|A+B| \leq |A| + |B|$; the fifth inequality follows from Massart's Finite Lemma. □

Lemma 8.14. *Let* $0 < 2\eta \leq r \leq 1$*. Given a hypothesis set* V *and data distribution* D *over* $\mathcal{X}\times\mathcal{Y}$*, if there exist* $h_1, h_2 \in V$ *such that* $\rho_D(h_1,h_2) \geq r$*, then* $\mathbf{\Phi}_D(V,\eta) \geq r - 2\eta$*.*

Proof. Let (α,β,γ) be a triple of functions from $\mathcal{X}$ to $\mathbb{R}^3$ satisfying the following conditions: $\alpha,\beta,\gamma \geq 0$, $\alpha+\beta+\gamma \equiv 1$, and for all $h\in V$,

$$\mathbb{E}_D[\alpha(x)\mathbb{1}(h(x) = -1) + \beta(x)\mathbb{1}(h(x) = +1)] \leq \eta$$

Then, in particular, we have:

$$\mathbb{E}_D[\alpha(x)\mathbb{1}(h_1(x)=-1)+\beta(x)\mathbb{1}(h_1(x)=+1)]\leq\eta$$

$$\mathbb{E}_D[\alpha(x)\mathbb{1}(h_2(x)=-1)+\beta(x)\mathbb{1}(h_2(x)=+1)]\leq\eta$$

Thus, by $\mathbb{1}(h_1(x)\neq h_2(x))\leq\min(\mathbb{1}(h_1(x)=-1)+\mathbb{1}(h_2(x)=-1),\mathbb{1}(h_1(x)=+1)+\mathbb{1}(h_2(x)=+1))$, adding the two inequalities up,

$$\mathbb{E}_D[(\alpha(x)+\beta(x))\mathbb{1}(h_1(x)\neq h_2(x))]\leq 2\eta$$

Since

$$\rho_D(h_1,h_2)=\mathbb{E}_D\mathbb{1}(h_1(x)\neq h_2(x))\geq r$$

We have

$$\mathbb{E}_D[\gamma(x)\mathbb{1}(h_1(x)\neq h_2(x))]=\mathbb{E}_D[(1-\alpha(x)-\beta(x))\mathbb{1}(h_1(x)\neq h_2(x))]\geq r-2\eta$$

Thus,

$$\mathbb{E}_D[\gamma(x)]\geq\mathbb{E}_D[\gamma(x)\mathbb{1}(h_1(x)\neq h_2(x))]\geq r-2\eta$$

Hence $\mathbf{\Phi}_D(V,\eta)\geq r-2\eta$. □

Lemma 8.15. *Given hypothesis set V and data distribution D over $\mathcal{X}\times\mathcal{Y}$, $0<\lambda<\eta<1$, if there exist $h_1,h_2\in V$ such that $\rho_D(h_1,h_2)\geq 2\eta-\lambda$, then $\mathbf{\Phi}_D(V,\eta)+\lambda\leq\mathbf{\Phi}_D(V,\eta-\lambda)$.*

Proof. Suppose $(\alpha_1,\beta_1,\gamma_1)$ are nonnegative functions satisfying $\alpha_1+\beta_1+\gamma_1\equiv 1$, and for all $h\in V$, $\mathbb{E}_D[\beta_1(x)\mathbb{1}(h(x)=+1)+\alpha_1(x)\mathbb{1}(h(x)=-1)]\leq\eta-\lambda$, and $\mathbb{E}_D\gamma_1(x)=\mathbf{\Phi}_D(V,\eta-\lambda)$. Notice by Lemma 8.14,$\mathbf{\Phi}_D(V,\eta-\lambda)\geq 2\eta-\lambda-2(\eta-\lambda)=\lambda$.

Then we pick nonnegative functions $(\alpha_2,\beta_2,\gamma_2)$ as follows. Let $\alpha_2=\alpha_1$, $\gamma_2=(1-\frac{\lambda}{\mathbf{\Phi}_D(V,\eta-\lambda)})\gamma_1$, and $\beta_2=1-\alpha_2-\gamma_2$. It is immediate that $(\alpha_2,\beta_2,\gamma_2)$ is a valid confidence rated predictor and $\beta_2\geq\beta_1$, $\gamma_2\leq\gamma_1$, $\mathbb{E}_D\gamma_2(x)=\mathbf{\Phi}_D(V,\eta-\lambda)-\lambda$. It can be readily checked that the

confidence rated predictor $(\alpha_2, \beta_2, \gamma_2)$ has error guarantee η, specifically:

$$\begin{aligned}
& \mathbb{E}_D[\beta_2(x)\mathbb{1}(h(x)=+1)+\alpha_2(x)\mathbb{1}(h(x)=-1)] \\
\leq\ & \mathbb{E}_D[(\beta_2(x)-\beta_1(x))\mathbb{1}(h(x)=+1)+(\alpha_2(x)-\alpha_1(x))\mathbb{1}(h(x)=-1)]+\eta-\lambda \\
\leq\ & \mathbb{E}_D[(\beta_2(x)-\beta_1(x))+(\alpha_2(x)-\alpha_1(x))]+\eta-\lambda \\
\leq\ & \lambda+\eta-\lambda=\eta
\end{aligned}$$

Thus, $\mathbf{\Phi}_D(V,\eta)$, which is the minimum abstention probability of a confidence-rated predictor with error guarantee η with respect to hypothesis set V and data distribution D, is at most $\mathbf{\Phi}_D(V,\eta-\lambda)-\lambda$. □

8.9 Detailed Derivation of Label Complexity Bounds

8.9.1 Agnostic with Fixed $\nu^*(D)$

Proposition 8.1. *In agnostic case, the label complexity of Algorithm 8.1 is at most*

$$\tilde{O}\left(\sup_{k\leq\lceil\log(1/\epsilon)\rceil}\frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{2\nu^*(D)+\epsilon_k}(d\frac{\nu^*(D)^2}{\epsilon^2}\ln\frac{1}{\epsilon}+d\ln^2\frac{1}{\epsilon})\right),$$

where the $\tilde{O}$ notation hides factors logarithmic in $1/\delta$.

Proof. Applying Theorem 8.3, the total number of labels queried is at most:

$$c_4\sum_{k=1}^{\lceil\log\frac{1}{\epsilon}\rceil}(d\ln\frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{\epsilon_k}+\ln(\frac{\lceil\log(1/\epsilon)\rceil-k+1}{\delta}))\frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{\epsilon_k}(1+\frac{\nu^*(D)}{\epsilon_k})$$

Using the fact that $\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)\leq 1$, this is

$$
\begin{aligned}
& c_4 \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} (d \ln \frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{\epsilon_k} + \ln(\frac{\lceil \log(1/\epsilon) \rceil - k + 1}{\delta})) \frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{\epsilon_k} \cdot \\
& (1+\frac{\nu^*(D)}{\epsilon_k}) \\
= \quad & \tilde{O}\left(\sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} (d \ln \frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{\epsilon_k} + \ln\log(1/\epsilon)) \frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{2\nu+\epsilon_k} (1+\frac{\nu^*(D)^2}{\epsilon_k^2}) \right) \\
\leq \quad & \tilde{O}\left(\sup_{k \leq \lceil \log(1/\epsilon) \rceil} \frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{2\nu^*(D)+\epsilon_k} \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} (1+\frac{\nu^*(D)^2}{\epsilon_k^2}) (d \ln \frac{1}{\epsilon} + \ln\ln \frac{1}{\epsilon}) \right) \\
\leq \quad & \tilde{O}\left(\sup_{k \leq \lceil \log(1/\epsilon) \rceil} \frac{\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256)}{2\nu^*(D)+\epsilon_k} (d \frac{\nu^*(D)^2}{\epsilon^2} \ln \frac{1}{\epsilon} + d \ln^2 \frac{1}{\epsilon}) \right),
\end{aligned}
$$

where the last line follows as ϵ_k is geometrically decreasing. □

Specifically, if $\mathcal{H}$ is the class of homogeneous linear classifiers in $\mathbb{R}^d$, $D_{\mathcal{X}}$ is isotropic log-concave in $\mathbb{R}^d$, then, our label complexity bound can be written as:

$$
O\left(\ln \frac{\epsilon+\nu^*(D)}{\epsilon} (\ln \frac{1}{\epsilon} + \frac{\nu^*(D)^2}{\epsilon^2}) (d \ln \frac{\epsilon+\nu^*(D)}{\epsilon} + \ln \frac{1}{\delta}) + \ln \frac{1}{\epsilon} \ln \frac{\epsilon+\nu^*(D)}{\epsilon} \ln\ln \frac{1}{\epsilon} \right)
$$

Indeed, recall by Lemma 8.12, we have $\phi(2\nu^*(D)+\epsilon_k,\epsilon_k/256) \leq C(\nu^*(D)+\epsilon_k) \ln \frac{\nu^*(D)+\epsilon_k}{\epsilon_k}$ for some constant $C>0$. Applying Theorem 8.2, the label complexity is

$$
O\left(\sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} (d \ln(\frac{2\nu^*(D)+\epsilon_k}{\epsilon_k} \ln \frac{2\nu^*(D)+\epsilon_k}{\epsilon_k}) + \ln(\frac{\log(1/\epsilon)-k+1}{\delta})) \ln \frac{2\nu^*(D)+\epsilon_k}{\epsilon_k} (1+\frac{\nu^*(D)^2}{\epsilon_k^2}) \right)
$$

This can be simplified to: (dealing with 1 and $\frac{\nu^*(D)^2}{\epsilon_k^2}$ separately)

$$
\begin{aligned}
& O\left(\sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} \ln \frac{\nu^*(D)+\epsilon_k}{\epsilon_k}(d \ln \frac{\nu^*(D)+\epsilon_k}{\epsilon_k} + \ln \frac{k_0-k+1}{\delta})\right. \\
& \left. + \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} \frac{\nu^*(D)^2}{\epsilon_k^2} \ln \frac{\nu^*(D)+\epsilon_k}{\epsilon_k}(d \ln \frac{\nu^*(D)+\epsilon_k}{\epsilon_k} + \ln \frac{k_0-k+1}{\delta})\right) \\
\leq \; & O\left(\ln \frac{1}{\epsilon} \ln \frac{\epsilon+\nu^*(D)}{\epsilon}(d \ln \frac{\epsilon+\nu^*(D)}{\epsilon} + \ln \ln \frac{1}{\epsilon} + \ln \frac{1}{\delta})\right. \\
& \left. + \frac{\nu^*(D)^2}{\epsilon^2} \ln \frac{\epsilon+\nu^*(D)}{\epsilon}(d \ln \frac{\epsilon+\nu^*(D)}{\epsilon} + \ln \frac{1}{\delta})\right) \\
\leq \; & O\left(\ln \frac{\epsilon+\nu^*(D)}{\epsilon}(\ln \frac{1}{\epsilon} + \frac{\nu^*(D)^2}{\epsilon^2})(d \ln \frac{\epsilon+\nu^*(D)}{\epsilon} + \ln \frac{1}{\delta}) + \ln \frac{1}{\epsilon} \ln \frac{\epsilon+\nu^*(D)}{\epsilon} \ln \ln \frac{1}{\epsilon}\right).
\end{aligned}
$$

8.9.2 Tsybakov Noise Condition with $\kappa > 1$

Proposition 8.2. *Suppose the hypothesis class $\mathcal{H}$ and the data distribution D satisfies (C_0, κ)-Tsybakov Noise Condition with $\kappa > 1$. Then the label complexity of Algorithm 8.1 is at most*

$$
\tilde{O}\left(\sup_{k \leq \lceil \log(1/\epsilon) \rceil} \frac{\phi(C_0 \epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})}{\epsilon_k^{\frac{1}{\kappa}}} \epsilon^{\frac{2}{\kappa}-2} d \ln \frac{1}{\epsilon}\right),
$$

where the $\tilde{O}$ notation hides factors logarithmic in $1/\delta$.

Proof. Applying Theorem 8.3, the total number of labels queried is at most:

$$
c_5 \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} (d \ln(\phi(C_0 \epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256}) \epsilon_k^{\frac{1}{\kappa}-2}) + \ln(\frac{k_0-k+1}{\delta})) \phi(C_0 \epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256}) \epsilon_k^{\frac{1}{\kappa}-2}
$$

Using the fact that $\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256}) \leq 1$, we get

$$\begin{aligned} & c_5 \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} (d\ln(\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})\epsilon_k^{\frac{1}{\kappa}-2}) + \ln(\frac{k_0-k+1}{\delta}))\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})\epsilon_k^{\frac{1}{\kappa}-2} \\ \leq & \tilde{O}\left(\sup_{k \leq \lceil \log(1/\epsilon) \rceil} \frac{\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})}{\epsilon_k^{\frac{1}{\kappa}}} \sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} \epsilon_k^{\frac{2}{\kappa}-2} d\ln \frac{1}{\epsilon} \right) \\ \leq & \tilde{O}\left(\sup_{k \leq \lceil \log(1/\epsilon) \rceil} \frac{\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})}{\epsilon_k^{\frac{1}{\kappa}}} \epsilon^{\frac{2}{\kappa}-2} d\ln \frac{1}{\epsilon} \right) \end{aligned}$$

□

Specifically, if $\mathcal{H}$ is the class of homogeneous linear classifiers in $\mathbb{R}^d$, and $D_{\mathcal{X}}$ is isotropic log-concave in $\mathbb{R}^d$, our label complexity bound is at most

$$O\left(\epsilon^{\frac{2}{\kappa}-2} \ln \frac{1}{\epsilon} (d\ln \frac{1}{\epsilon} + \ln \frac{1}{\delta}) \right)$$

Indeed, recall by Lemma 8.12, we have$\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256}) \leq C\epsilon_k^{\frac{1}{\kappa}} \ln \frac{1}{\epsilon_k}$ for some constant $C > 0$. Applying Theorem 8.3, the label complexity is:

$$O\left(\sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} (d\ln(\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})\epsilon_k^{\frac{1}{\kappa}-2}) + \ln(\frac{k_0-k+1}{\delta}))\phi(C_0\epsilon_k^{\frac{1}{\kappa}}, \frac{\epsilon_k}{256})\epsilon_k^{\frac{1}{\kappa}-2} \right)$$

This can be simplified to :

$$\begin{aligned} & O\left(\sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} (d\ln(\epsilon_k^{\frac{2}{\kappa}-2} \ln \frac{1}{\epsilon_k}) + \ln(\frac{k_0-k+1}{\delta}))\epsilon_k^{\frac{2}{\kappa}-2} \ln \frac{1}{\epsilon_k} \right) \\ \leq & O\left((\sum_{k=1}^{\lceil \log \frac{1}{\epsilon} \rceil} \epsilon_k^{\frac{2}{\kappa}-2}) \ln \frac{1}{\epsilon} (d\ln \frac{1}{\epsilon} + \ln \frac{1}{\delta}) \right) \\ \leq & O\left(\epsilon^{\frac{2}{\kappa}-2} \ln \frac{1}{\epsilon} (d\ln \frac{1}{\epsilon} + \ln \frac{1}{\delta}) \right). \end{aligned}$$

8.10 Acknowledgements

This chapter is based on the material as it appears in Advances in Neural Information Processing Systems 2014 (Chicheng Zhang and Kamalika Chaudhuri, "Beyond Disagreement-based Agnostic Active Learning"). The dissertation author is the primary investigator and author of this material.

Appendix A

Concentration Inequalities

We collect a few concentration inequalities that will be used throughout this thesis.

Lemma A.1 (Bernstein's Inequality)**.** *Let $X_1, \ldots, X_n$ be independent zero-mean random variables. Suppose that $|X_i| \leq M$ almost surely. Then for all positive t,*

$$\Pr\left[\sum_{i=1}^{n} X_i > t\right] \;\leq\; \exp\left(-\frac{t^2/2}{\sum_{j=1}^{n} \mathbb{E}[X_j^2] + Mt/3}\right).$$

Lemma A.2. *Let $Z_1, \ldots, Z_n$ be independent Bernoulli random variables with mean p. Let $\bar{Z} = \frac{1}{n}\sum_{i=1}^{n} Z_i$. Then with probability $1-\delta$,*

$$\bar{Z} \;\leq\; p + \sqrt{\frac{2p\ln(1/\delta)}{n}} + \frac{2\ln(1/\delta)}{3n}.$$

Proof. Let $X_i = Z_i - p$ for all i, note that $|X_i| \leq 1$. The lemma follows from Bernstein's Inequality and algebra. □

Lemma A.3 (Freedman's Inequality)**.** *Let $X_1, \ldots, X_n$ be a martingale difference sequence, and $|X_i| \leq M$ almost surely. Let V be the sum of the conditional variances, i.e.*

$$V = \sum_{i=1}^{n} \mathbb{E}[X_i^2 | X_1, \ldots . X_{i-1}]$$

Then, for every $t, v > 0$,

$$\Pr\left[\sum_{i=1}^{n} X_i > t \text{ and } V \le v\right] \le \exp\left(-\frac{t^2/2}{v + Mt/3}\right).$$

Lemma A.4. *Let $Z_1, \ldots, Z_n$ be a sequence of Bernoulli random variables, where $\mathbb{E}[Z_i | Z_1, \ldots, Z_{i-1}] = p_i$. Then, for every $\delta > 0$, with probability $1 - \delta$:*

$$\sum_{i=1}^{n} Z_i \le 2v_n + \sqrt{4v_n \ln \frac{\log 4n}{\delta}} + \frac{2}{3} \ln \frac{\log 4n}{\delta}.$$

where $v_n = \max(\sum_{i=1}^{n} p_i, 1)$.

Proof. Let $X_i = Z_i - p_i$ for all i, note that $\{X_i\}$ is a martingale difference sequence and $|X_i| \le 1$. From Freedman's Inequality and algebra, for any v,

$$\Pr\left[\frac{1}{n}\sum_{i=1}^{n} Z_i > v + \sqrt{\frac{2v \ln \frac{\log 4n}{\delta}}{n}} + \frac{2 \ln \frac{\log 4n}{\delta}}{3n} \text{ and } \sum_{i=1}^{n} p_i \le v\right] \le \frac{\delta}{\log n + 2}.$$

The proof follows by taking union bound over $v = 2^i, i = 0, 1, \ldots, \lceil \log n \rceil$. □

Define:

$$\sigma(d, n, \delta) = \frac{8}{n}(2d \ln \frac{2en}{d} + \ln \frac{24}{\delta}), \tag{A.1}$$

where d is the VC dimension of the hypothesis class $\mathcal{H}$. when it is clear from context, we sometimes abbreviate the above as $\sigma(n, \delta)$.

Theorem A.1 ([VC71])**.** *Let $\mathcal{F}$ be a family of functions $f \colon \mathcal{Z} \to \{0, 1\}$ on a domain $\mathcal{Z}$ with VC dimension at most d, and let P be a distribution on $\mathcal{Z}$. Let P_n denote the empirical measure from an iid sample of size n from P. For any $\delta \in (0, 1)$, with probability at least $1 - \delta$, for all $f \in \mathcal{F}$,*

$$-\min\left\{\varepsilon + \sqrt{Pf\varepsilon}, \sqrt{P_n f \varepsilon}\right\} \;\le\; Pf - P_n f \;\le\; \min\left\{\varepsilon + \sqrt{P_n f \varepsilon}, \sqrt{Pf\varepsilon}\right\}$$

where $\varepsilon := \sigma(d, n, \delta)$.

We have the following simple fact about $\sigma(\cdot, \cdot, \cdot)$.

Fact A.1.

$$\sigma\left(d, m, \frac{\delta}{2\log m(\log m+1)}\right) \geq \epsilon \implies m \leq \frac{64}{\epsilon}\left(d\log\frac{512}{\epsilon}+\log\frac{24}{\delta}\right).$$

The following lemma is a corollary of Theorem A.1; we use the version of [Hsu10].

Lemma A.5. *Pick any $n \geq 1$, $\delta \in (0,1)$. Let S_n be a set of n iid copies of (X,Y) drawn from a distribution D over labeled examples. Then, the following hold with probability at least $1-\delta$ over the choice of S_n:*
(1) For all $h \in \mathcal{H}$,

$$|\operatorname{err}_D(h) - \operatorname{err}_{S_n}(h)| \leq \min\left(\sigma(n,\delta)+\sqrt{\sigma(n,\delta)\operatorname{err}_D(h)}, \sigma(n,\delta)+\sqrt{\sigma(n,\delta)\operatorname{err}_{S_n}(h)}\right) \tag{A.2}$$

In particular, all classifiers h in $\mathcal{H}$ consistent with S_n satisfies

$$\operatorname{err}_D(h) \leq \sigma(n,\delta) \tag{A.3}$$

(2) For all h, h' in $\mathcal{H}$,

$$(\operatorname{err}_D(h)-\operatorname{err}_D(h'))-(\operatorname{err}_{S_n}(h)-\operatorname{err}_{S_n}(h')) \leq \sigma(n,\delta)+\min\left(\sqrt{\sigma(n,\delta)\rho_D(h,h')}, \sqrt{\sigma(n,\delta)\rho_{S_n}(h,h')}\right) \tag{A.4}$$

$$|\rho_D(h,h') - \rho_{S_n}(h,h')| \leq \sigma(n,\delta)+\min\left(\sqrt{\sigma(n,\delta)\rho_D(h,h')}, \sqrt{\sigma(n,\delta)\rho_{S_n}(h,h')}\right) \tag{A.5}$$

Where $\sigma(n,\delta)$ is defined in Equation (A.1)*.*

We occasionally use the following (weaker) version of Lemma A.5.

Lemma A.6. *Pick any $n \geq 1$, $\delta \in (0,1)$. Let S_n be a set of n iid copies of (X,Y). The following holds with probability at least $1-\delta$: (1) For all $h \in \mathcal{H}$,*

$$|\operatorname{err}_D(h) - \operatorname{err}_{S_n}(h)| \leq \sqrt{4\sigma(n,\delta)} \tag{A.6}$$

(2) For all h, h' in $\mathcal{H}$,

$$(\operatorname{err}_D(h)-\operatorname{err}_D(h'))-(\operatorname{err}_{S_n}(h)-\operatorname{err}_{S_n}(h')) \leq \sqrt{4\sigma(n,\delta)} \tag{A.7}$$

$$|\rho_D(h,h') - \rho_{S_n}(h,h')| \leq \sqrt{4\sigma(n,\delta)} \tag{A.8}$$

Where $\sigma(n,\delta)$ *is defined in Equation* (A.1).

Bibliography

[ABDL13] Alekh Agarwal, Léon Bottou, Miroslav Dudík, and John Langford. Para-active learning. *CoRR*, abs/1310.8243, 2013.

[ABHU15] Pranjal Awasthi, Maria-Florina Balcan, Nika Haghtalab, and Ruth Urner. Efficient learning of linear separators under bounded noise. In *Proceedings of The 28th Conference on Learning Theory, COLT 2015, Paris, France, July 3-6, 2015*, pages 167–190, 2015.

[ABHZ16] Pranjal Awasthi, Maria-Florina Balcan, Nika Haghtalab, and Hongyang Zhang. Learning and 1-bit compressed sensing under asymmetric noise. In *Proceedings of The 28th Conference on Learning Theory, COLT 2016*, 2016.

[ABL14] Pranjal Awasthi, Maria Florina Balcan, and Philip M Long. The power of localization for efficiently learning linear separators with noise. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 449–458. ACM, 2014.

[Ale87] Kenneth S Alexander. Rates of growth and sample moduli for weighted empirical processes indexed by sets. *Probability Theory and Related Fields*, 75(3):379–423, 1987.

[ALW06] Jacob Abernethy, John Langford, and Manfred K. Warmuth. Continuous experts and the binning algorithm. In *19th Annual Conference on Learning Theory, COLT 2006*, pages 544–558, 2006.

[Ang87] Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1987.

[Ang04] Dana Angluin. Queries revisited. *Theor. Comput. Sci.*, 313(2):175–194, 2004.

[AP10] Josh Attenberg and Foster Provost. Why label when you can search?: alternatives to active learning for applying human resources to build classification models under extreme class imbalance. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 423–432. ACM, 2010.

[Bal16] Akshay Balsubramani. Learning to abstain from binary prediction. *arXiv preprint arXiv:1602.08151*, 2016.

[BB05] Nader H. Bshouty and Lynn Burroughs. Maximizing agreements with one-sided error with applications to heuristic learning. *Machine Learning*, 59(1-2):99–123, 2005.

[BBL09] M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. *J. Comput. Syst. Sci.*, 75(1):78–89, 2009.

[BBZ07] M.-F. Balcan, A. Z. Broder, and T. Zhang. Margin based active learning. In *COLT*, 2007.

[BDL09] A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. In *ICML*, 2009.

[BH12] Maria Florina Balcan and Steve Hanneke. Robust interactive learning. In *Conference on Learning Theory*, pages 20–1, 2012.

[BHK+11] Alina Beygelzimer, Daniel Hsu, Nikos Karampatziakis, John Langford, and Tong Zhang. Efficient active learning. In *ICML 2011 Workshop on On-line Trading of Exploration and Exploitation*, 2011.

[BHLZ10] A. Beygelzimer, D. Hsu, J. Langford, and T. Zhang. Agnostic active learning without constraints. In *NIPS*, 2010.

[BHV10] Maria-Florina Balcan, Steve Hanneke, and Jennifer Wortman Vaughan. The true sample complexity of active learning. *Machine learning*, 80(2-3):111–139, 2010.

[BL13] M.-F. Balcan and P. M. Long. Active and passive learning of linear separators under log-concave distributions. In *COLT*, 2013.

[Boa04] Michael Boardman. The egg drop number. *Mathematics Magazine*, 77(5):368–372, 2004.

[BPS09] Shai Ben-David, Dávid Pál, and Shai Shalev-Shwartz. Agnostic online learning. In *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009*, 2009.

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[BW08] P. L. Bartlett and M. H. Wegkamp. Classification with a reject option using a hinge loss. *JMLR*, 9, 2008.

[CAL94] D. A. Cohn, L. E. Atlas, and R. E. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2), 1994.

[CBL06] Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

[CDM16a] Corinna Cortes, Giulia DeSalvo, and Mehryar Mohri. Boosting with abstention. In *Advances in Neural Information Processing Systems*, pages 1660–1668, 2016.

[CDM16b] Corinna Cortes, Giulia DeSalvo, and Mehryar Mohri. Learning with rejection. In *International Conference on Algorithmic Learning Theory*, pages 67–82. Springer, 2016.

[CDV10] K. Chandrasekaran, D. Dadush, and S. Vempala. Thin partitions: Isoperimetric inequalities and a sampling algorithm for star shaped bodies. In M. Charikar, editor, *SODA*. SIAM, 2010.

[CFHW96] Nicolò Cesa-Bianchi, Yoav Freund, David P. Helmbold, and Manfred K. Warmuth. On-line prediction and conversion strategies. *Machine Learning*, 25(1):71–110, 1996.

[CHK17] Lin Chen, Seyed Hamed Hassani, and Amin Karbasi. Near-optimal active learning of halfspaces via query synthesis in the noisy setting. In *AAAI*, pages 1798–1804, 2017.

[Cho70] C.K. Chow. On optimum error and reject trade-off. *IEEE Trans. on Inform. Theory*, 1970.

[CKW05] K. Crammer, M. J. Kearns, and J. Wortman. Learning from data of variable quality. In *NIPS*, 2005.

[CL11] C-C. Chang and C-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2, 2011.

[CN07] Rui Castro and Robert D. Nowak. Minimax bounds for active learning. In *COLT*, pages 5–19, 2007.

[CTGC05] Edward Y Chang, Simon Tong, Kingshy Goh, and Cheng-Wei Chang. Support vector machine concept-dependent active learning for image retrieval. 2005.

[Das04] S. Dasgupta. Analysis of a greedy active learning strategy. In *NIPS*, 2004.

[Das05] S. Dasgupta. Coarse sample complexity bounds for active learning. In *NIPS*, 2005.

[Das11] S. Dasgupta. Two faces of active learning. *Theor. Comput. Sci.*, 412(19), 2011.

[DC08] P. Donmez and J. Carbonell. Proactive learning: Cost-sensitive active learning with multiple imperfect oracles. In *CIKM*, 2008.

[DGL96] Luc Devroye, László Györfi, and Gabor Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer Verlag, 1996.

[DGS12] O. Dekel, C. Gentile, and K. Sridharan. Selective sampling and active learning from single and multiple teachers. *JMLR*, 13:2655–2697, 2012.

[DH08] S. Dasgupta and D. Hsu. Hierarchical sampling for active learning. In *ICML*, 2008.

[DHM07] S. Dasgupta, D. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. In *NIPS*, 2007.

[DZ13] Erik D. Demaine and Morteza Zadimoghaddam. Learning disjunctions: Near-optimal trade-off between mistakes and "i don't know's". In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013*, pages 1369–1379, 2013.

[EYW10] R. El-Yaniv and Y. Wiener. On the foundations of noise-free selective classification. *JMLR*, 2010.

[EYW11] R. El-Yaniv and Y. Wiener. Agnostic selective classification. In *NIPS*, 2011.

[EYW12] R. El-Yaniv and Y. Wiener. Active learning via perfect selective classification. *JMLR*, 2012.

[FMS04] Y. Freund, Y. Mansour, and R. E. Schapire. Generalization bounds for averaged classifiers. *The Ann. of Stat.*, 32, 2004.

[Fri09] Eric Friedman. Active learning for smooth problems. In *COLT*, 2009.

[FSST97] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168, 1997.

[FZL+12] Meng Fang, Xingquan Zhu, Bin Li, Wei Ding, and Xindong Wu. Self-taught active learning from crowds. In *ICDM*, pages 858–863. IEEE, 2012.

[GBNT05] R. Gilad-Bachrach, A. Navot, and N. Tishby. Query by committee made real. In *NIPS*, 2005.

[GEY17] Roei Gelbhart and Ran El-Yaniv. The relationship between agnostic selective classification active learning and the disagreement coefficient. *arXiv preprint arXiv:1703.06536*, 2017.

[GF08] William Gasarch and Stuart Fletcher. The egg game. 2008.

[GK11] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

[GWBV02] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3), 2002.

[HAH+15] Tzu-Kuo Huang, Alekh Agarwal, Daniel Hsu, John Langford, and Robert E. Schapire. Efficient and parsimonious agnostic active learning. In *Advances in Neural Information Processing Systems 28*, 2015.

[Han06] Steve Hanneke. The cost complexity of interactive learning. 2006.

[Han07] S. Hanneke. A bound on the label complexity of agnostic active learning. In *ICML*, 2007.

[Han09] S. Hanneke. Adaptive rates of convergence in active learning. In *COLT*, 2009.

[Han14] Steve Hanneke. Theory of disagreement-based active learning. *Foundations and Trends® in Machine Learning*, 7(2-3):131–309, 2014.

[HLV+16] Tzu-Kuo Huang, Lihong Li, Ara Vartanian, Saleema Amershi, and Xiaojin Zhu. Active learning with oracle epiphany. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2820–2828. Curran Associates, Inc., 2016.

[Hsu10] D. Hsu. *Algorithms for Active Learning*. PhD thesis, UC San Diego, 2010.

[HW06] R. Herbei and M. H. Wegkamp. Classfication with reject option. *Canadian J. of Stat.*, 4, 2006.

[HY12] S. Hanneke and L. Yang. Surrogate losses in passive and active learning. *CoRR*, abs/1207.3772, 2012.

[IPSW14] Panagiotis G Ipeirotis, Foster Provost, Victor S Sheng, and Jing Wang. Repeated labeling using multiple noisy labelers. *Data Mining and Knowledge Discovery*, 28(2):402–441, 2014.

[Kää06] M. Kääriäinen. Active learning in the non-realizable case. In *ALT*, 2006.

[KKM12] Adam Tauman Kalai, Varun Kanade, and Yishay Mansour. Reliable agnostic learning. *J. Comput. Syst. Sci.*, 78(5):1481–1495, 2012.

[KLMZ17] Daniel M Kane, Shachar Lovett, Shay Moran, and Jiapeng Zhang. Active classification with comparison queries. *arXiv preprint arXiv:1704.03564*, 2017.

[Kol10] V. Koltchinskii. Rademacher complexities and bounding the excess risk in active learning. *JMLR*, 2010.

[KSS94] M. J. Kearns, R. E. Schapire, and L. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994.

[KT14] Varun Kanade and Justin Thaler. Distribution-independent reliable learning. In *COLT*, 2014.

[KUBD15] Samory Kpotufe, Ruth Urner, and Shai Ben-David. Hierarchical label queries with data-dependent partitions. In *Conference on Learning Theory*, pages 1176–1189, 2015.

[LC05] Y. Lecun and C. Cortes. The mnist database of handwritten digits. 2005.

[LG94] David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12. Springer-Verlag New York, Inc., 1994.

[Lit87] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4), 1987.

[LLWS11] Lihong Li, Michael L. Littman, Thomas J. Walsh, and Alexander L. Strehl. Knows what it knows: a framework for self-aware learning. *Machine Learning*, 82(3):399–443, 2011.

[LMW14] C. H. Lin, Mausam, and D. S. Weld. To re(label), or not to re(label). In *HCOMP*, 2014.

[LMW15] C.H. Lin, Mausam, and D.S. Weld. Reactive learning: Actively trading off larger noisier training sets against smaller cleaner ones. In *ICML Workshop on Crowdsourcing and Machine Learning and ICML Active Learning Workshop*, 2015.

[LS06] Mingkun Li and Ishwar K Sethi. Confidence-based active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1251–1261, 2006.

[LV06] L. Lovász and S. Vempala. Hit-and-run from a corner. *SIAM J. Comput.*, 35(4), 2006.

[MCR14] L. Malago, N. Cesa-Bianchi, and J. Renders. Online active learning with strong and weak annotators. In *NIPS Workshop on Learning from the Wisdom of Crowds*, 2014.

[Mit82] Tom M Mitchell. Generalization as search. *Artificial intelligence*, 18(2):203–226, 1982.

[MP17] Mason McGill and Pietro Perona. Deciding how to decide: Dynamic routing in artificial neural networks. *arXiv preprint arXiv:1703.06217*, 2017.

[MT89] Wolfgang Maass and György Turán. On the complexity of learning from counterexamples. In *Foundations of Computer Science, 1989., 30th Annual Symposium on*, pages 262–267. IEEE, 1989.

[MT04] Enno Mammen and Alexandre B. Tsybakov. Smooth discrimination analysis. *Ann. Statist.*, 32(5):2340–2341, 10 2004.

[Muk03] S. Mukherjee. Chapter 9. classifying microarray data using support vector machines. In *of scient. from the Univ. Penn. Sch. of Med. and the Sch. of EAS*. Kluwer Academic Publishers, 2003.

[NJC13] M. Naghshvar, T. Javidi, and K. Chaudhuri. Noisy bayesian active learning. In *Allerton*, 2013.

[Now11] R. D. Nowak. The geometry of generalized binary search. *IEEE Transactions on Information Theory*, 57(12):7893–7906, 2011.

[PD17] Stefanos Poulis and Sanjoy Dasgupta. Learning with feature feedback: from theory to practice. In *Artificial Intelligence and Statistics*, pages 1104–1113, 2017.

[RMJ06] Hema Raghavan, Omid Madani, and Rosie Jones. Active learning with feedback on features and instances. *Journal of Machine Learning Research*, 7(Aug):1655–1686, 2006.

[RR11] Maxim Raginsky and Alexander Rakhlin. Lower bounds for passive and active learning. In *Advances in Neural Information Processing Systems*, pages 1026–1034, 2011.

[RSS12] Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Relax and randomize : From value to algorithms. In *Advances in Neural Information Processing Systems 25*, pages 2150–2158, 2012.

[RST10] Alexander Rakhlin, Karthik Sridharan, and Ambuj Tewari. Online learning: Random averages, combinatorial parameters, and learnability. In *Advances in Neural Information Processing Systems*, pages 1984–1992, 2010.

[RST15a] Alexander Rakhlin, Karthik Sridharan, and Ambuj Tewari. Online learning via sequential complexities. *The Journal of Machine Learning Research*, 16(1):155–186, 2015.

[RST15b] Alexander Rakhlin, Karthik Sridharan, and Ambuj Tewari. Sequential complexities and uniform martingale laws of large numbers. *Probability Theory and Related Fields*, 161(1-2):111–153, 2015.

[SCL+14] Patrice Y. Simard, David Maxwell Chickering, Aparna Lakshmiratan, Denis Xavier Charles, Léon Bottou, Carlos Garcia Jurado Suarez, David Grangier, Saleema Amershi, Johan Verwey, and Jina Suh. ICE: enabling non-experts to build models interactively for large-scale lopsided problems. *CoRR*, abs/1409.4814, 2014.

[SCS15] S. Song, K. Chaudhuri, and A. D. Sarwate. Learning from data with heterogeneous noise using sgd. In *AISTATS*, 2015.

[Set10] B. Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison, 2010.

[Sha12] Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

[Sim14] Hans Ulrich Simon. Pac-learning in the presence of one-sided classification noise. *Ann. Math. Artif. Intell.*, 71(4):283–300, 2014.

[SS11] István Szita and Csaba Szepesvári. Agnostic kwik learning and efficient approximate reinforcement learning. In *COLT*, pages 739–772, 2011.

[SSBD14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

[SV08] G. Shafer and V. Vovk. A tutorial on conformal prediction. *JMLR*, 2008.

[SZB10] A. Sayedi, M. Zadimoghaddam, and A. Blum. Trading off mistakes and don't-know predictions. In *NIPS*, 2010.

[TD17] Christopher Tosh and Sanjoy Dasgupta. Diameter-based active learning. *arXiv preprint arXiv:1702.08553*, 2017.

[TK01] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. pages 45–66, 2001.

[TS13] Kirill Trapeznikov and Venkatesh Saligrama. Supervised sequential classification under budget constraints. In *Artificial Intelligence and Statistics*, pages 581–589, 2013.

[Tsy04] A. B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Annals of Statistics*, 32:135–166, 2004.

[UBS12] R. Urner, S. Ben-David, and O. Shamir. Learning from weak teachers. In *AISTATS*, pages 1252–1260, 2012.

[UWBD13] R. Urner, S. Wulff, and S. Ben-David. Plal: Cluster-based active learning. In *COLT*, 2013.

[Val84] L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984.

[Vap82] Vladimir N. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, 1982.

[VC71] Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971.

[Wan11] Liwei Wang. Smoothness, disagreement coefficient, and the label complexity of agnostic active learning. *Journal of Machine Learning Research*, 12(Jul):2269–2292, 2011.

[WHEY15] Yair Wiener, Steve Hanneke, and Ran El-Yaniv. A compression technique for analyzing disagreement-based active learning. *Journal of Machine Learning Research*, 16(4):713–745, 2015.

[XZM+17] Yichong Xu, Hongyang Zhang, Kyle Miller, Aarti Singh, and Artur Dubrawski. Noise-tolerant interactive learning from pairwise comparisons with near-minimal label complexity. *arXiv preprint arXiv:1704.05820*, 2017.

[YCJ16] Songbai Yan, Kamalika Chaudhuri, and Tara Javidi. Active learning from imperfect labelers. In *Advances in Neural Information Processing Systems*, pages 2128–2136, 2016.

[YRF+12] Y. Yan, R. Rosales, G. Fung, F. Farooq, B. Rao, and J. G. Dy. Active learning from multiple knowledge sources. In *AISTATS*, pages 1350–1357, 2012.

[YRFD11] Y. Yan, R. Rosales, G. Fung, and J. G. Dy. Active learning from crowds. In *ICML*, pages 1161–1168, 2011.

[YW10] M. Yuan and M. H. Wegkamp. Classification methods with reject option based on convex risk minimization. *JMLR*, 11, 2010.