EFFECTIVE CLASSIFICATION LEARNING

by

Dale Eric Schuurmans

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto

Effective Classification Learning

Doctor of Philosophy, 1996
Dale Eric Schuurmans
Graduate Department of Computer Science
University of Toronto

Abstract

This thesis addresses the problem of learning a classification rule from random examples. We first consider the problem of learning a target concept with guaranteed accuracy and reliability given that the target belongs to some known class C; a task commonly referred to as *probably approximately correct (pac) learning*. Previous work on this problem assumes a fixed-sample-size approach to data collection that fails to achieve practical data-efficiency in most applications. In this thesis we consider an alternative "sequential" approach where the learner observes training examples one-at-a-time and decides on-line when to stop training. We prove that sequential learning strategies can pac-learn with fewer training examples than previous fixed-sample-size approaches, even while incurring minimal computational overhead. Moreover, these new strategies use many times fewer training examples in practical case studies.

Next, we study the average error of a learner's hypotheses as a function of training sample size—its so-called *learning curve*. Specifically, we investigate the best learning curve that can be achieved in the worst case over a class of concepts C. Previous work has shown that rational convergence to zero error can always be obtained in this model, but it is impossible to do better in the worst case. However, recent empirical studies have shown that exponential convergence can be achieved in many experimental settings. We explain this discrepancy by noting that the previous analysis is non-uniform in training sample size, and prove that a uniform analysis predicts the exact same dichotomy as observed in the experimental studies. Overall, this thesis shows how the worst case theory of classification learning can be brought closer to practice.

Acknowledgements

First, I would like to thank my two supervisors, Russell Greiner and Hector Levesque, for their helpful guidance and free donation of their time. My research was greatly improved by their suggestions and by the high standards with which they conduct their own work. Throughout, Russ has been an enthusiastic and stimulating collaborator as well as friend. After Russ' departure from Toronto, Hector took up my local supervisory duties with poise and professionalism, and I'd like to thank him for that. I was also extremely fortunate to have an exceptional supervisory committee consisting of Stephen Cook, Geoffrey Hinton, Allan Borodin, and Allan Jepson. Not only was their feedback quick and incisive, but they were remarkably kind. I also benefitted from the feedback of my two external examiners, Rob Tibshirani and Nader Bshouty.

My years in Toronto have left me with many friends and colleagues whom I'd like to thank. From start to finish, Michael Grüninger and Sheila McIlraith were constant friends: always there to bounce ideas off of and to share the highs and lows. Thanks to Steven Shapiro for always being willing to engage in last minute problem solving sessions and crazy debates. I'd also like to thank Richard Mann, Evan Steeg, and John Funge for their interest in my work and the productive discussions we shared—their feedback was a tremendous help. Other friends I've made along the way are Mike Godfrey, Javier Pinto, John de Haan, and Kathy Yen. Last but not least, I'd like to thank my friend Marc Ouellette for sharing in the pursuit of our many sporting diversions.

For much of our time in Toronto, we were fortunate to have the company of my sister Carol and her husband Peter. I am grateful for the coincidence that brought them to Toronto and for the time we spent together. I'd also like to thank both of my families for their love and support: especially my parents (and parents-in-law!) and my sister Carmen.

Finally, my deepest thanks go to my wife, Sharon, for her enduring love, support, and patience. This thesis is as much a product of her resolve and optimism as it is of my efforts. I could not have succeeded without her constant encouragement, unwavering commitment, and grace. Thank you Sharon.

Publication notes

Some of the work reported in this thesis has appeared in the following publications: Much of the material from Chapter 2 appears in [Schuurmans and Greiner, 1995a] and [Schuurmans and Greiner, 1995b]. Some preliminary results from Chapter 3 were reported in [Schuurmans, 1996b]. The key results from Chapter 4 appear in [Schuurmans, 1995], and a full paper is currently under review [Schuurmans, 1996a]. I would like to thank my co-author, Russ Greiner, and the copyright holders for their permission to include this material here.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Classification learning

Machine Learning studies computational systems that improve their performance at a task with experience. This thesis considers a specific form of learning task—classification learning—which is by far the most studied in machine learning research.

Classification learning is the problem of producing an accurate classification function for some domain, given the correct classifications of a few domain objects. Abstractly, we have a domain of objects X and a fixed classification scheme $c : X \to Y$ that assigns each domain object x to one class $c(x)$ from a mutually exclusive set of classes Y. Given a sequence of training examples $\langle\langle x_1, c(x_1)\rangle, \langle x_2, c(x_2)\rangle, ..., \langle x_t, c(x_t)\rangle\rangle$ the goal is to produce a classification function $f : X \to Y$ that agrees with the correct classification scheme c over as much of the domain X as possible.

For example, one might be interested in classifying emergency room patients into various disease categories based on their overt symptoms, *e.g.*, determining whether a patient has *meningitis* given observations of *rash* and *flu* symptoms. In the event the correct classification scheme is unknown *a priori*, the idea is to exploit the existence of a few training examples and extrapolate their classifications to an accurate scheme over the entire domain. For example, upon observing symptoms

⟨*flu, rash* ⟩	classified as *meningitis*,
⟨*no flu, rash* ⟩	classified as *not meningitis*,
⟨*no flu, no rash* ⟩	classified as *not meningitis*,

we might postulate a general classification rule

$$\textit{meningitis} \;\Leftrightarrow\; \textit{flu} \text{ and } \textit{rash}$$

over the entire domain of patients (hence, classifying the unseen example ⟨*flu, no rash*⟩ as *not meningitis*). This general pattern of reasoning is known as *induction* in the philosophical literature, or *supervised learning* in the machine learning literature. In practice, domain objects might be described in a number of ways; *e.g.*, vectors of Boolean attributes, real-valued attributes, or structured descriptions like strings, graphs, term structures, *etc.* Regardless of the specific representation however, the central question is always how best to extrapolate the classifications of a few domain objects to obtain a general scheme that accurately covers as much of the domain as possible.

Motivation

Classification learning has been widely investigated by numerous (largely disjoint) research communities, in a wide variety of application areas. The immense interest is due to the fact that *classification* itself is an important subtask in many applications (in fact, comprising the central function of most expert systems [Clancey, 1985]). *Learning* turns out to be a useful technique for synthesizing effective classification systems in many cases. This is because in many domains where adequate classification schemes are not known *a priori*, it is nevertheless possible to obtain a number of correctly classified training examples; for example, by directly observing historical data (*e.g.*, weather and stock market patterns), intrusive examination procedures (*e.g.*, exploratory surgery), consultation with domain experts, or "mining" existing databases [Piatetsky-Shapiro and Frawley, 1991]. These training examples can be exploited to produce effective global classification schemes. In fact, there are many successful examples where learning systems have produced classifiers that outperform the best available hand-coded systems. For example,

[Buchanan and Mitchell, 1978] discusses a system that learned to predict the structure of organic molecules from their mass spectrograms;

[Qian and Sejnowski, 1988] presents a neural network that learned to predict the secondary structure of protein molecules from their amino acid sequences;

[le Cun et al., 1989] develops a system that learned to recognize hand-printed digits from digitized images; and

[Weiss and Kulikowski, 1991] describe a number of systems that have learned to diagnose patient diseases from laboratory test results.

The success of these efforts demonstrates that it can be advantageous to exploit a collection of correctly classified training examples rather than directly engineering a classifier on the basis of inadequate domain knowledge. Even in domains where adequate knowledge might reside with certain domain experts, it has been suggested that learning be used to overcome the "knowledge acquisition bottleneck" [Michalski, 1983].

Goals

As mentioned, the central task of classification learning is to produce a globally accurate classification scheme given the classifications of a few domain objects. Of course, we are most interested in developing strategies that do this effectively; *i.e.*, produce an accurate classification scheme (covering most if not all of the domain), using few training examples, within reasonable computational limits. So there are two distinct aspects to extrapolation effectiveness here: data-efficiency (the number of training examples needed to produce an accurate classification scheme), and computational-efficiency (the computational resources required to produce the final classifier). Data-efficiency is particularly important since in practice correctly classified training examples often come at non-negligible cost and are in limited supply. Of course, it is also essential that a learning system produce a hypothesis using reasonable computational resources for it to be practically useful.

There is a wide variety of ideas on how best to extrapolate the classifications of a few domain objects to obtain globally accurate classification schemes. Many extrapolation strategies have been proposed in the literature and their generalization performance empirically investigated in many domains. The specific nature of each proposal tends to vary with different choices of domain object representation. However, despite these differences, the general efficacy of these strategies is invariably demonstrated via empirical case studies on one or two domains. These investigations have led to some impressive results in specific applications, as noted above, but no doubt these successes are vastly outnumbered by numerous unreported failures.

Although most reported research examines the particular performance properties of specific extrapolation strategies on specific domains, the underlying goal of machine learning research is to uncover whatever *general* principles might underly effective extrapolation. The most common research strategy seems to be abstracting these general principles from empirically successful case studies. However, one does not have to think about the problem long before realizing that perhaps there are no general principles to be found: given the classifications of domain objects $x_1, x_2, ..., x_t$, there is nothing really that can be inferred about the classification of an unobserved object x in general. The classification of x need have nothing whatsoever to do with the classifications of $x_1, x_2, ..., x_t$—unless x's classification was somehow constrained by the classifications of $x_1, x_2, ..., x_t$. The singular aspect of an extrapolation strategy yielding success in a particular application is best described as *fortuitous predisposition*: the strategy just happens to guess right on unseen domain objects, whether by prior knowledge or plain luck. This elementary observation has been made many times in the past [Mitchell, 1980; Wantanabe, 1987], and recently asserted yet again [Schaffer, 1994].

Theory

A recent trend in machine learning research is the theoretical analysis of classification learning. This represents a fundamental shift in emphasis away from uncovering "general purpose" or "universal" extrapolation strategies, towards explicitly acknowledging the essential role played by prior knowledge and constraints in yielding successful extrapolation.

A theoretical analysis of classification learning must adopt a mathematical model of the learning situation (characterizing the source of training and test examples), and make *explicit* assumptions about whatever prior constraints and knowledge are available about the target classification. The underlying goal of theoretical analysis is to determine how best to exploit available prior knowledge and constraints to achieve successful extrapolation in practice. Given explicit assumptions it becomes possible to ask: What levels of generalization and computational performance are possible to achieve in principle? What is impossible? Which extrapolation strategies are most effective under specific circumstances? What are the optimum levels of generalization and computational performance that can be achieved for given prior constraints?

The two best known theoretical analyses of classification learning are due to Gold [1967] and Valiant [1984]. Gold's analysis models the learning situation by assuming a countable domain X and considering arbitrary sequences of examples that enumerate the domain. Prior knowledge is modelled by assuming the target classification scheme $c : X \to Y$ belongs to some known class C. Gold then asks whether, for a given class C, there exists a computable guessing strategy that makes at most a finite number of mistakes on any enumeration of examples from some classification scheme c in C.

Valiant models the learning situation by instead assuming examples are randomly generated according to a fixed distribution P_X on the domain X. Prior knowledge is also modelled by assuming the target scheme c belongs to some known class C. Valiant then asks whether, for a given class C, there exists an efficient learning algorithm that reliably produces an accurate approximation to any target classification c from C, given a reasonable number of training examples generated by an arbitrary domain distribution P_X.

Regardless of the specific mathematical model one adopts, the role of theoretical analysis is not to *prescribe* whatever prior knowledge or constraints one ought to have about a classification task *a priori*, but rather to provide the best course of action one might take *given* whatever is known (or presumed) about the task beforehand. Theoretical analysis can provide better prescriptions for practice and offer deeper insights into the sources of learning efficacy than empirical study—provided the mathematical model adequately captures the situation encountered in practice.

Thesis

This thesis focuses on the theoretical analysis of classification learning, primarily considering the random example model popularized by Valiant [1984]. Since Valiant's work, this model has seen a recent explosion in interest under the rubric of "pac-learning theory." In spite of this increased theoretical activity however, pac-learning theory has arguably had little direct impact on practice: its prescriptions are rarely if ever followed, and the current theory does not capture all empirically observed phenomena. The goal of this thesis is to address some of these shortcomings by re-assessing many basic assumptions of the existing theory, and identifying more natural alternatives that lead to theoretical prescriptions and insights that are more relevant to practice.

1.2 Model: learning from random examples

Before outlining the specific topics addressed by this thesis, I first survey relevant results from the current theory of learning a classification rule from random examples.

I.i.d. random example model

The *independent identically distributed (i.i.d.) random example model* is the most common mathematical model adopted in theoretical analyses of classification learning. This model has been brought into recent prominence by the work of Valiant, but has also been well studied long before that, *cf.* [Vapnik and Chervonenkis, 1971; Duda and Hart, 1973]. In this model we assume there is a natural distribution P_X on the domain X that randomly and independently generates domain objects which are then classified according to a fixed scheme $c : X \to Y$. The learner receives a set of random training examples from which it must produce a classification function $f : X \to Y$ that will then be tested on random examples drawn from the same random source. The *accuracy* of a hypothesis f is just the probability it correctly classifies a random domain object according to c. The learner's goal is to produce an accurate classifier using as few training examples (and computational resources) as possible. This is a natural model of many learning situations where there is no significant relationship between successive training and testing examples and the example generation process remains stable over time.

A significant assumption made throughout much of the literature is that the example generation process is noise-free. That is, the language used to represent domain objects is adequate to capture the distinctions encoded by the target classification scheme, and furthermore, the true classifications of all training objects are correctly reported. Most theoretical research also focuses on an important special case of learning a two-category classification scheme (*i.e.*, $Y = \{0, 1\}$); commonly known as *concept learning.* A *concept* $c : X \to \{0, 1\}$ specifies a subset of the domain, $c^{-1}(1) \subset X$, with those domain objects falling within this subset are known as positive examples, and those falling outside as negative examples. Generally speaking, most theoretical results concerning concept learning can be scaled up to general classification learning in a natural way. Concept learning represents the minimal core problem.

A number of different analyses can be considered within this model. The two most significant analyses consider the difficulty of producing an accurate classifier with guaranteed reliability ("pac-learning"), and the expected error of a learner's hypotheses as a function of training sample size ("learning curves").

Reliably accurate learning

The problem of producing an accurate classification scheme with guaranteed reliability has received a lot of attention since the work of Valiant [1984]. The specific problem is, given specified accuracy and reliability levels $1 - \epsilon$ and $1 - \delta$ respectively, to observe random examples of some unknown target concept c and produce a classifier with accuracy at least $1 - \epsilon$, with an overall probability of at least $1 - \delta$ over possible training sequences. That is, the learner must return a sufficiently accurate hypothesis most of the time, but is allowed to fail sometimes and return an inaccurate hypothesis, but only with some small probability δ. This has since become known as the "pac-criterion" (for "$\underline{\text{p}}$robably $\underline{\text{a}}$pproximately $\underline{\text{c}}$orrect" [Angluin, 1988]).

The idea is that these two parameters provide the learning system designer with sufficient flexibility to appropriately tailor the training requirements to the task at hand, keeping in mind that more stringent levels require more training resources. The accuracy parameter ϵ specifies the desired quality of the final classification system, and the reliability parameter δ specifies the quality of the training process (*i.e.*, the reliability with which the target accuracy is achieved). So, for example, if an application only requires a moderately accurate classifier (some moderately small value of ϵ), but it is crucial that this minimal accuracy level be attained, then δ could be set sufficiently small so as to provide a reasonable guarantee. On the other hand, if all one really desires is a reasonable chance of producing an extremely accurate hypothesis, then ϵ could be set much smaller than δ, *etc.* The specific choice is left up to the application designer and not fixed by the theory.

For given ϵ and δ, the learner's goal is to meet the pac-criterion using as few training examples as possible within reasonable computational limits. Of course this can be easy or hard depending on what is known about the target concept *a priori.* For example, if we already knew the exact identity of the target concept (or that it was one of very few possibilities) the pac-criterion would be trivial to achieve. If, on the other hand, the target concept were part of a completely arbitrary set of concepts on a large domain, then achieving the pac-criterion could be impossible. In order to analyze the difficulty of achieving the pac-criterion we need to explicitly model whatever knowledge and prior constraints we might have about the target concept before training begins.

Valiant considered a specific form of prior knowledge: namely, that the target concept c belongs to some known class of concepts C, but nothing is known about the domain distribution P_X, which could be arbitrary. Within this model of prior knowledge it is natural to demand for a concept class C that the learner meet the pac-criterion for an arbitrary target concept in C, regardless of the underlying domain distribution. That is, a learner successfully pac-learns a concept class C if it can guarantee an accurate hypothesis with sufficient reliability in the worst case over all possible target concepts c in C and all possible domain distributions P_X on X.

For example, say we need an accurate classification scheme for screening emergency room patients for *meningitis*, but we are unable to specify an appropriate rule. Then we can consider learning an accurate classification scheme by observing examples of correctly classified emergency room patients. Assume for the sake of argument that we know the correct classification rule is given by some conjunctive definition of the *rash* and *flu* conditions of the patient, but that nothing is known about the distribution of patients. Say we require a classifier that has an error of no more than 1%, but can tolerate a probability of at most, say, 5% of not achieving this target. Given these specifications, Valiant asks: Is there a learning procedure that can meet these goals? How many training examples are required? Can an acceptable classifier be produced with reasonable computational resources?

The *data-efficiency* of a pac-learning procedure (with respect to ϵ and δ) is given by the maximum number of training examples it uses in the worst case. The data-*complexity* of a concept class C (with respect to ϵ and δ) is given by the smallest number of training examples required that any learning procedure requires to successfully pac-learn C. Intuitively, the data-complexity of a concept class should be related to its overall "representational complexity." That is, the greater the variety of possible target concepts, the more training examples it should take to disambiguate an adequate concept from the multitude of inaccurate concepts. For example, a class that contains only two mutually exclusive concepts can be trivially pac-learned by observing a single training example, since one example is always sufficient to identify the target. On the other hand, it is obviously impossible to pac-learn arbitrarily complex concept classes, like the class of all subsets of the real line; *e.g.*, given a target concept $\varnothing$, no finite set of training examples can ever rule out every bad concept like $\mathbb{R} - \{\text{finite set}\}$.

The general intuition that it should be harder to pac-learn complex concept classes than simple ones has been precisely captured by a deep theory of Vapnik and Chervonenkis [1971]. They identify a measure of representational complexity known as the "Vapnik-Chervonenkis dimension" (or just VCdimension) that precisely characterizes the data-complexity of pac-learning concept classes. Their results have been adapted to show that the data-complexity of pac-learning a class C depends *linearly* on its VCdimension. In particular, Ehrenfeucht *et al.* [1989] have shown that, for fixed ϵ and δ, the minimum number of training examples required to pac-learn C grows as a linear function of $\mathrm{VC}(C)$. Moreover, Blumer *et al.* [1989] demonstrate a simple pac-learning procedure that obtains a data-efficiency that is within constant factors of this lower bound (and a log factor of $1/\epsilon$).

Another aspect of learning performance, orthogonal to data-efficiency, is computational-efficiency; *i.e.*, the computational resources a learner requires to produce its hypotheses. Computational-efficiency is one of the main issues addressed by Valiant in his seminal paper. Specifically, Valiant considered the problem of pac-learning parameterized *families* of concept classes $\{C_n\}$ defined on parameterized families of domains $\{X_n\}$ with a single learning algorithm L. Following standard practice in complexity-theory, Valiant asks whether, for a family $\{C_n\}$, there exists a learning algorithm L that successfully pac-learns each class C_n with a training sample size and computation time bounded by some polynomial in n, $1/\epsilon$, and $1/\delta$.

Overall, this type of analysis has important implications for practice, as it provides explicit prescriptions for obtaining accurate classification schemes with guaranteed reliability, under very general learning circumstances. This type of analysis could be important in domains where it is critical to guarantee the accuracy of any proposed classification scheme.

Learning curves

Of course, under the i.i.d. random examples model, it is possible to analyze other aspects of learning performance beyond pac-learning. For example, another important form of analysis considers the expected error of a learner's hypotheses as a function of training sample size. That is, rather than considering the probability

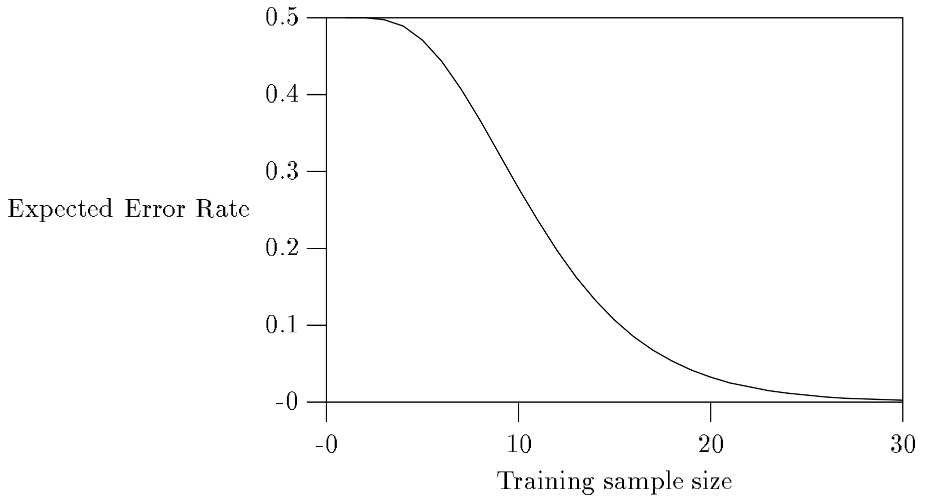

Figure 1.1: A learning curve

that a learner produces a hypothesis with error below some threshold, we instead consider the *average* error obtained by the learner's hypotheses. The expected error achieved by a hypothesis guessing strategy as a function of its training sample size describes its overall *learning curve.* The learning curve is exactly what one estimates by repeatedly training a learning system at various training sample sizes and plotting the average error of its hypotheses. For example, Figure 1.1 shows the exact learning curve obtained by a particular hypothesis guessing strategy for learning conjunctive concept definitions on a domain defined by 50 boolean attributes, under a uniform domain distribution [Pazzani and Sarrett, 1990]. Clearly, in this example the average error of the learner's hypotheses decreases rapidly for larger training sample sizes.

As before, we expect the shape of the learning curve to depend on how much the learning system knows about the target concept beforehand. Obviously, if we have a lot of prior information (*e.g.*, that the target concept belongs to a small set of possibilities) we expect to obtain a learning curve that rapidly converges to zero error, since the target concept is likely to be disambiguated after a few training examples. On the other hand, we would expect to obtain poor convergence rates if very little is known about the target classification *a priori.*

An analysis of learning curves in much the same spirit as Valiant's analysis of reliably accurate learning has been performed by Haussler, Littlestone and Warmuth [1988]. In particular, they adopt the same model of prior knowledge (that the target concept belongs to a known class C, but nothing is known about the domain distribution) but ask a different question: For a fixed training sample size t, what is the smallest expected error any hypothesis guessing strategy can achieve in the worst case over all possible target concepts and all possible domain distributions? For different sample sizes t this characterizes the best "worst case" (*i.e.*, minimax) learning curve that can be achieved for a concept class C. Analogous to the previous pac-learning results, Haussler, Littlestone and Warmuth [1988] show that, for a fixed t, the minimax expected error of any (reasonable) concept class C is a *linear* function of its VCdimension. Moreover, their results show that *all* minimax learning curves exhibit "rational" convergence to zero error. That is, the best achievable worst case learning curve always converges as a function $\Theta(t^{-1})$ in terms of training sample size t. (This is also known as "inverse power law" convergence [Haussler et al., 1994] since for example, cutting the minimax expected error in half requires the training sample size to be doubled.)

This form of learning curve analysis provides an important characterization of learning performance. In some sense, the average hypothesis error as a function of training sample size gives the true measure of an extrapolation strategy's overall generalization efficacy. The fact that minimax learning curves always exhibit rational convergence appears to be a deep insight. It says that we can never hope to obtain better than power law convergence in generalization error for any non-trivial concept class.

Assessment

Overall, these theoretical results have greatly increased our knowledge and understanding of classification learning problems. For instance, they clearly demonstrate the effects of prior knowledge on learning effectiveness, and moreover, do so in a precise quantitative fashion. Furthermore, these results clearly circumscribe the limits of learning: characterizing what can and cannot be achieved, and helping us to see what is in fact possible. These results are not only just descriptive in nature, but also *prescriptive*, in the sense that they provide constructive procedures for obtaining specified learning performance levels. Finally, the abstract nature of the theory, in ignoring superfluous representational and algorithmic details, makes it quite powerful. The general form of prior constraints considered by the theory makes its results applicable to a wide range of learning situations commonly encountered in practice (*e.g.*, learning function classes defined by neural network architectures).

However, despite these strengths, the current theory has not had a significant impact on practice. We outline some of the reasons for this below and suggest ways the current theory can be improved.

1.3 Thesis

The analysis of classification learning under the i.i.d. random example model has been quite influential on the theory and practice of machine learning. The descriptive aspects of this theory have influenced the general practice of machine learning, by pointing out the essential role played by prior knowledge and constraints in determining learning effectiveness. However, the prescriptive aspects of this theory have generally been ignored by practitioners, who point to a number of impracticalities in the specific requirements, and the fact that the theory fails to explain certain empirical phenomena.

This thesis addresses these shortcomings by: first identifying those assumptions in the existing theory that are responsible for these perceived weaknesses, proposing more natural assumptions in their place, and finally determining whether this brings the theory any closer to practicality. Specific progress is reported in three areas:

1. Improving the empirical data-efficiency of pac-learning procedures to practical levels.

2. Incorporating distributional assumptions to obtain further efficiency improvements.

3. Extending the theory of minimax learning curves to account for empirically observed *exponential* convergence behavior.

Efficient pac-learning (Chapter 2)

Pac-learning theory promises useful prescriptions for practice by providing procedures that learn with guaranteed accuracy and reliability. However, the current prescriptions of this theory are rarely (if ever) followed in real applications. The main reason for this is simple: the particular training sample sizes currently demanded by pac-learning theory are far too large to be practical. That is, the minimum sample sizes that have been proved sufficient for obtaining guaranteed accuracy and reliability are far larger than deemed practically reasonable by practitioners. This is especially problematic since training examples are often the critical resource in practical learning applications, and therefore, minimizing the number of training examples required is crucial.

The apparent impracticality of pac-learning has lead many researchers to speculate about the sources of inefficiency. The common wisdom is that this impracticality is due to the worst case nature of the pac-learning guarantees. However, this prevailing view may not be entirely accurate, since:

1. The current sample size bounds [Blumer et al., 1989; Shawe-Taylor, Anthony and Biggs, 1993] incorporate approximations that go well beyond just taking the worst case situation into account. (For example, the sample size bounds that have been proved sufficient and necessary differ by a factor of more than 64.)

2. Pac-analyses typically consider a simplistic learning strategy (fix a sample size, collect this many training examples, and then choose a consistent hypothesis) that may not be making the most efficient use of the available training data.

In Chapter 2 of this thesis we focus on the second of these alternatives, by asking whether more sophisticated learning strategies might require fewer training examples than the simple fixed-sample-size approach in practice. In particular, we consider *sequential* pac-learning strategies that autonomously decide when to stop their own training based on the specific sequence of training examples they observe. This chapter proposes a number of sequential pac-learning strategies, and investigates their data (and computational) efficiency, both analytically and experimentally. We ask whether these techniques are substantially more efficient than current pac-learning strategies, if so, how much of an improvement is possible, and whether any new concept classes become effectively learnable. A number of theoretical results are presented that show these sequential learning strategies are provably more efficient than current pac-learning approaches (for certain accuracy and reliability levels) while achieving the exact same worst case pac-guarantees. Furthermore, additional experimental results show these new strategies are *many times* more efficient in practice.

Efficient distribution-specific pac-learning (Chapter 3)

Although pac-learning can be more efficiently achieved in practice than previously thought, the practicality of the preceding results is still equivocal. That is, even though there is potential for additional improvement, it might well be that worst case pac-learning really *is* impractical. Many researchers adopt the line of reasoning that there must be "pathological" domain distributions that force impractical training sample sizes—arguing moreover that these pathological distributions do not arise in "typical" applications. This points to the need for making additional distributional assumptions to achieve practical results [Aha, Kibler and Albert, 1991; Albert and Aha, 1991]. In fact, many researchers have investigated an extreme model that considers the difficulty of pac-learning from a *known* domain distribution. This is known as *distribution-specific* pac-learning. The problem is to learn an accurate approximation to an unknown target concept c from a known class C, given that the distribution of domain objects, P_X, is actually known *a priori* [Benedek and Itai, 1991; Kulkarni, 1991].

In Chapter 3 of this thesis we observe that, even given these distributional assumptions, it is still possible to consider a sequential approach to learning. In this chapter we investigate the data-efficiency of sequential strategies for distribution-specific pac-learning. Surprisingly, it turns out that even stronger results can be obtained here than in the previous distribution-free setting. Specifically, we propose a number of new sequential learning strategies, analyze their data-efficiency, and show that the data-efficiency of these new procedures uniformly dominates the previous fixed-sample-size approaches—requiring average training sample sizes that are many times smaller.

In this chapter we also investigate a stronger form of learning where we demand that the learner return an accurate hypothesis with *certainty* (*i.e.*, with probability 1), not just high probability—a learning criterion we refer to as cac-learning (for c̲ertainly a̲pproximately c̲orrect). It is shown that cac-learning is impossible in the distribution-free setting, but can be achieved in the distribution-specific setting via sequential learning. A simple sequential learning procedure is proposed for this case, and shown to learn with *optimal* expected data-efficiency.

Exponential versus rational learning curves (Chapter 4)

Next, we turn our attention to the theory of minimax learning curves. Here we find that the current theory also has some weaknesses. In particular, the theory predicts that all minimax learning curves exhibit the same rational convergence to zero error [Haussler, Littlestone and Warmuth, 1988; 1994], whereas other forms of convergence are often observed in practice. For example, in a recent series of experiments Cohn and Tesauro [1990; 1992] observe *exponential* learning curves, in addition to the rational curves predicted by the previous theory. This phenomenon of exponential convergence is posited as a clear weakness of the current theory of minimax learning curves: the theory completely misses the significant dichotomy between rational and exponential convergence. This raises the obvious question of reconciling the theory with the empirical observations.

Chapter 4 of this thesis addresses this issue. Although it is possible to speculate that this oversight is simply due to the worst case nature of the analysis, we ask whether the dichotomy between exponential and rational convergence can still be predicted from a worst case analysis of learning curve behavior. By noting a simple weakness in the original theory, we show that, in fact, this dichotomy can be revealed in a worst case analysis. The basic observation is that the previous analysis of Haussler, Littlestone and Warmuth [1988] is *non-uniform* in training sample size. That is, the lower bounds are established by choosing *different* target concepts and domain distributions for each training sample size to force bad behavior. By undertaking a uniform analysis, we show that the dichotomy between exponential and rational minimax learning curves can be predicted from the finiteness or continuity of the prior concept class. These results show that the experimental results of Cohn and Tesauro are no accident: For finite concept classes any consistent learner achieves an exponential worst case learning curve, whereas continuous concept classes have rational minimax learning curves. Further analysis shows that the exact boundary between these two modes of convergence is determined by the presence of any dense subchains in the prior concept class (*i.e.*, whether the class contains any chain of concepts such that between any two concepts in the chain there is a third).

Contributions

Overall, the results of this research show how the theoretical analysis of classification learning can be made more relevant to practice. In particular it is shown how pac-learning might be more efficiently achieved in practice than previously thought, and how making additional assumptions only serves to make these improvements more significant. It is also shown how the current theory can be extended to account for empirical phenomena regularly encountered in practice. Throughout, the general aim is to pursue abstract theoretical results that are as general (and, hence, as widely applicable) as possible.

1.4 Overview

Each chapter of this thesis is largely self contained; supplying a survey of the relevant literature, presentation of the main results, and discussion of relevant research directions.

Chapter 2 first investigates the use of sequential stopping rules to improve the data-efficiency of distribution-free pac-learning, presenting both theoretical and empirical results to support this claim.

Chapter 3 then extends these results to the distribution-specific setting (where one presumes the domain distribution is known *a priori*), and finds that even stronger efficiency improvements can be obtained. The chapter also investigates a stronger learning criterion that demands the learner return a sufficiently accurate hypothesis with certainty, not just high probability. It is shown that sequential learning is *necessary* to achieve this criterion, and we derive an optimal learning technique for this case.

Chapter 4 then shifts attention to the theory of learning curves. This chapter investigates the empirical dichotomy between rational and exponential convergence that is left unaccounted for by the current theory of worst case learning curves. A uniform analysis that keeps the domain distribution and target concept fixed throughout the training process shows that exponential versus rational learning curves can be predicted from the structure of the concept class. A precise boundary is drawn between these two convergence modes based on the existence of any dense chains in the prior concept class.

Chapter 5 then concludes the thesis with a discussion of the implications of these results for practical classification learning. Here we note that many other generalizations of the current theory are also possible, and these are contemplated as directions for future research.

Chapter 2

Distribution-free *sequential* pac-learning

2.1 Introduction

By far the most influential analysis of classification learning has been the theory of "probably approximately correct" (pac) learning introduced by Valiant [1984]. Rather than speculate about the various strategies that might underly effective "general purpose" classification learning, Valiant's idea was to characterize those situations where successful learning could be provably achieved and where it was demonstrably impossible.

Pac-learning

Pac-learning theory addresses the problem of learning an accurate concept definition from examples: given a sequence of training examples describing domain objects and their membership in an unknown target concept, the task is to infer a concept definition that agrees with the target concept over as much of the domain as possible. This theory considers a particular mathematical model of the learning environment, the "i.i.d. random examples model," which assumes training and test examples are randomly and independently generated according to a fixed domain distribution and classified according to a fixed target concept. Given this model, we demand that the learner meet the *pac-criterion*; *i.e.*, return a hypothesis of minimum specified accuracy with some minimum specified probability. Of course, achieving the pac-criterion can be easy or hard depending on what is known about the target concept beforehand. Following the seminal work of Valiant, pac-learning theory adopts a model of prior knowledge that assumes the target concept belongs to some known class C, but nothing is known about the distribution of domain objects *a priori*. Given these assumptions it is natural to demand that the learner meet the pac-criterion for any target concept from C and any domain distribution. Thus, we say a learner pac-learns a class C if it meets the specified pac-criterion in the worst case over all possible domain distributions and all possible target concepts in C.

Efficiency and complexity

Since the original formulation of this task many researchers have investigated the difficulty of pac-learning different concept classes defined on a variety of domains. The main issues are the existence and efficiency of these solutions. That is, we wish to solve pac-learning problems, while using a minimum of data and computational resources. Research has focused on developing provably correct pac-learning procedures, analyzing their data and computational efficiency, and determining the minimum training resources needed to solve pac-learning problems.

Some of the most important technical results of this theory concern the data resources required to pac-learn. Intuitively, it takes more training examples to pac-learn a complex concept class than a simple class, since it is harder to disambiguate possible targets from a complex class. The question is: how can the complexity of a concept class be measured in a way that precisely determines the number of training examples needed to pac-learn? It turns out that the *Vapnik-Chervonenkis dimension* (VCdimension) provides just such a measure [Vapnik and Chervonenkis, 1971]. With this notion, researchers have been able to show that the minimum number of training examples required to pac-learn any concept class C grows as a linear function of its VCdimension [Ehrenfeucht et al., 1989]. Moreover, there is a simple fixed-sample-size learning procedure that pac-learns concept classes with a training sample size that is within constant factors of this lower bound (logarithmic in $1/\epsilon$). This procedure simply collects a large training sample that is sufficient to ensure that, with high probability, every concept in C with a large error misclassifies at least one training example; and then returns any concept in C that correctly classifies every training example [Blumer et al., 1989].

Issue

Overall, these results precisely characterize how the number of training examples needed to pac-learn scales up in terms of the desired accuracy and reliability levels and the VCdimension of the prior concept class C. This theory is constructive in the sense that it provides a generic procedure for pac-learning with near-optimal data-efficiency. On the strength of these and other results, pac-learning has become a predominant form of theoretical analysis of classification learning.

However, despite this success, pac-learning theory has had little direct impact on the practice of machine learning. That is, practitioners rarely heed the results of this theory in designing learning systems for real applications. Why? Perhaps, the most common criticism of pac-learning theory is that the training sample sizes it demands are far too large to be practical: Even though the data-efficiency of the simple fixed-sample-size learning procedure scales-up near-optimally, in practice the specific constants of scaling are excessive and make the theory inapplicable to realistic situations. This apparent inefficiency has lead to much speculation about the sources of difficulty. The predominant folk wisdom is that these impractical training sample sizes inevitably follow from the *worst case* nature of the pac-analysis. That is, guaranteeing the pac-criterion for all possible target concepts and domain distributions necessitates an excessive number of training examples, and therefore (distribution-free) pac-learning, though desirable, must not be a practical goal. However, this view may not be entirely accurate, since:

1. The current sample size bounds proved sufficient for pac-learning incorporate many approximations that go beyond just taking the worst case target concept and domain distribution into account (*i.e.*, the constants in the current bounds are not tight).

2. The simplistic fixed-sample-size learning strategy tacitly assumed by most pac-learning research may not make the most efficient use of available training data.

Approach

The research reported in this chapter begins with the second observation: Perhaps alternative learning strategies might pac-learn more efficiently than the simple fixed-sample-size approach? Specifically, we consider a *sequential* approach where the learner observes training examples one at a time and decides on-line whether to stop and produce a hypothesis, or continue training. The idea is that, by observing the specific sequence of training examples, we might be able to detect situations when an accurate hypothesis can be reliably returned and stop well before the fixed-sample-size bounds are reached. The goal is to reduce the number of training examples observed while still providing the exact same worst case pac-guarantees as before; namely, that a hypothesis with error at most ϵ be returned with probability at least $1-\delta$, regardless of the domain distribution and target concept from the prior class.

This sequential approach raises some new issues that do not arise under the fixed-sample-size approach. One issue is that the number of training examples a sequential learner observes depends on the specific training sequence, and hence is a random variable rather than just a fixed number. Therefore, to compare the data-efficiency of a sequential learner to a fixed-sample-size learner we must compare a distribution of training sample sizes to a fixed number. Although there are many ways one could do this, we will focus on what is arguably the most natural measure: we compare the average (*i.e.*, expected) training sample size of a sequential learner with the fixed sample size used by a fixed-sample-size learner to solve the exact same pac-learning problem.

This chapter asks whether sequential learning is able to pac-learn with fewer training examples than fixed-sample-size learning; if so, how much of an improvement is possible in principle; and whether any such improvement makes pac-learning practically achievable.

Results

In this chapter we propose new sequential learning strategies and prove them to be correct pac-learners. One of these procedures is shown to have a data-efficiency that scales the same as fixed-sample-size approaches (up to constant factors)—with a slight improvement over previous bounds for certain accuracy and reliability levels. Although this theoretical advantage is not overwhelming, a series of experiments show that this sequential learning procedure actually uses many times fewer training examples in practice than existing fixed-sample-size approaches, while attaining the exact same worst case pac-learning guarantees.

An analysis of the intrinsic data-complexity of sequential pac-learning reveals that the inherent data-requirements of sequential learning must scale the same as fixed-sample-size learning up to constant factors. Thus, for fixed accuracy and reliability levels, sequential learning can at best offer a constant improvement in expected training sample size over fixed-sample-size procedures. But, even these constant improvements can matter a great deal in practice.

Stronger results are obtained for the special case of finite concept classes, where both theoretical and experimental results show how sequential learning substantially improves the data-efficiency of fixed-sample-size learning when converting a "mistake bounded" hypothesizer to a pac-learner.

Finally, it is revealed that sequential learning is, in fact, applicable to a wider range of learning problems than fixed-sample-size learning. Specifically, sequential learning procedures are able to pac-learn many concept classes that have infinite VCdimension, which is impossible for any fixed-sample-size learner.

Overview

Before investigating the sequential approach to pac-learning, we first review the relevant aspects of pac-learning theory. Section 2.2 provides basic definitions of pac-learning theory and surveys current results concerning the efficiency of fixed-sample-size learning procedures and the intrinsic complexity of pac-learning problems. Section 2.3 then discusses the issue of practical data-efficiency—demonstrating the impracticality of existing fixed-sample-size learning procedures, and noting that a comparison of fixed versus sequential learners entails comparing fixed versus expected training sample sizes.

Section 2.4 then begins the investigation of sequential pac-learning by proposing a few sequential learning procedures, proving them correct, and investigating their data-efficiency. This section then analyzes the intrinsic data-complexity of sequential pac-learning and shows that the proposed procedure attains near-optimal scaling in data-efficiency. The primary strength of these procedures is their *empirical* data-efficiency, which Section 2.5 shows to be many times better than existing fixed-sample-size approaches. Section 2.6 then considers the special case of pac-learning finite concept classes and investigates the data-efficiency of converting mistake-bounded hypothesizers to pac-learners.

Aside from improving the data-efficiency of pac-learning, Section 2.7 notes how sequential learning procedures are applicable to a wider range of pac-learning problems than fixed-sample-size learning, specifically pac-learning many concept classes that have infinite VCdimension.

Section 2.8 concludes this chapter by discussing the implications of these results and suggesting directions for future research. Overall, these results indicate that pac-learning can be achieved more efficiently in practice than previously thought, and that pac-learning may yet be a practically achievable criterion in many applications.[1]

2.2 Background: distribution-free pac-learning theory

Before investigating the effectiveness of sequential learning procedures, we first review the definitions of pac-learning theory and survey the existing results concerning correct pac-learning procedures, their efficiency,

[1]Much of the material from this chapter appears in [Schuurmans and Greiner, 1995a] and [Schuurmans and Greiner, 1995b]. Permission has been obtained from IJCAI and ACM respectively for inclusion of this material here.

and the inherent complexity of pac-learning problems.

2.2.1 Problem

Pac-learning theory addresses the problem of learning a concept definition from examples. Formally, we have a domain of objects X and a target concept c defined on X. Here a *concept* c is just a subset of the domain, which we represent by its indicator function $c : X \to \{0,1\}$, where $c(x) = 1$ indicates x is a member of the concept, and $c(x) = 0$ indicates that x is not a member. An *example* is a pair $\langle x, c(x)\rangle$ that describes a domain object x and indicates its membership in the target concept c. Here, we consider a *batch* training protocol where the learner is given a sequence of training examples $\langle\langle x_1, c(x_1)\rangle, \langle x_2, c(x_2)\rangle, ...\rangle$ from which it must produce a hypothesis $h : X \to \{0,1\}$ that is then tested *ad infinitum* on subsequent test examples. A hypothesis h makes an *error* on any test example $\langle x, c(x)\rangle$ for which $h(x) \neq c(x)$.

Pac-learning theory adopts the i.i.d. random example model of the learning environment, which assumes there is a fixed, natural distribution P_X defined on the domain X. Training objects are drawn randomly and independently according to P_X and classified by a fixed target concept c before being presented to the learner. Any hypothesis the learner produces will then be tested on the *same* distribution of examples. Therefore, the *error* of a hypothesis h is just the probability it misclassifies a random domain object according to c, $\mathrm{P}_X\{h(x) \neq c(x)\}$.[2] We say that a hypothesis h is *ϵ-bad* if $\mathrm{P}_X\{h(x) \neq c(x)\} > \epsilon$, and otherwise *$\epsilon$-good* (or *$\epsilon$-accurate*).

Within this model, pac-learning theory considers the difficulty of producing an accurate hypothesis with guaranteed reliability. Specifically, for given accuracy and reliability parameters ϵ and δ, we demand that the learner meet the *pac(ϵ,δ)-criterion*; *i.e.*, produce a hypothesis of error at most ϵ, with probability at least $1-\delta$.

Of course, the difficulty of meeting the pac-criterion depends on what is known about the target concept and domain distribution beforehand. Following Valiant, most pac-learning research adopts a model of prior knowledge where we assume the target concept c belongs to some known class C but nothing is known about the domain distribution P_X, which could be arbitrary. Given these assumptions it is natural to demand that, for specified C, ϵ, and δ, the learner meet the pac-criterion in the worst case over all potential target concepts in C and all possible domain distributions P_X. Therefore, we specify an instance of a pac-learning problem by a concept class C, accuracy parameter ϵ, and reliability parameter δ.

Definition 2.1 (Pac-learning problem) *A learner L solves the pac-learning problem (C, ϵ, δ) (or "pac(ϵ,δ)-learns C") if, given random training objects generated by any distribution P_X and labelled according to any target concept $c \in C$, L produces a hypothesis h such that $\mathrm{P}_X\{h(x) \neq c(x)\} \leq \epsilon$ with probability at least $1-\delta$.*

Solving a pac-learning problem involves two main sub-tasks: *(i)* deciding when a sufficient number of training examples have been observed to ensure the pac-guarantees, and *(ii)* finding an appropriate hypothesis to return given these training examples. Therefore, we formalize a learner L as consisting of:

1. a *sample size function* $T_L(C,\epsilon,\delta)$ that determines a suitable training sample size for given C, ϵ, and δ; and

2. a *hypothesizer* $H_L(C,\epsilon,\delta) : (X \times \{0,1\})^* \to \{0,1\}^X$ that maps finite sequences of training examples to hypotheses (which in general could be a stochastic or even non-deterministic mapping).

Note that these definitions ignore representational and computational details specific to particular domain object or classification function representations. In practice, we will have to represent domain objects x in some concrete way, and H_L will need to algorithmically produce some effective representation of its hypotheses $h : X \to \{0,1\}$ that can be used to classify domain objects. So any computational questions cannot be divorced from the specific domain and hypothesis representations. However, we adopt this abstract

[2] This definition assumes that all concept labels are correctly reported to the learner; *i.e.*, that classification is noise free. This is a common assumption made throughout much of pac-learning theory. Some researchers have considered more general settings that permit noise, *cf.* [Angluin and Laird, 1988; Kearns and Li, 1988; Haussler, 1992; Kearns, 1993], but we will focus on the noise free case here.

Procedure `F` $(C, \epsilon, \delta;\ H)$

INPUT: target concept class C,
accuracy parameter ϵ,
reliability parameter δ;
a hypothesizer H that returns consistent concepts from C.

RETURN: a hypothesis h with accuracy at least $1 - \epsilon$, with probability at least $1 - \delta$.

PROCEDURE:

- Determine a training sample size $T_{\mathtt{F}}(C, \epsilon, \delta)$ that is sufficient to eliminate every ϵ-bad concept in C with probability at least $1 - \delta$ regardless of which target concept c from C is used to label the examples.
- Collect $T_{\mathtt{F}}(C, \epsilon, \delta)$ random training examples labelled by some unknown target concept $c \in C$.
- Call H to obtain an arbitrary concept $h \in C$ that correctly classifies every training example; return h.

Figure 2.1: Procedure `F`

mathematical view here because any statistical (*i.e.*, pac) properties of a learner L will always be independent of such computational details. Therefore, we will only need to consider such abstract specifications of learning procedures when addressing the *data* requirements of pac-learning.

2.2.2 Procedures

Successful pac-learning procedures have been developed for a variety of concept classes defined on many different domains. Interestingly, most of these procedures follow the same basic fixed-sample-size strategy first proposed by Valiant [1984]: Simply collect a large training sample sufficient to ensure that, with probability at least $1 - \delta$, any concept in C with error greater than ϵ misclassifies at least one training example; then return any concept from C that is consistent with every training example. Such a hypothesis will automatically be guaranteed to satisfy the pac(ϵ, δ)-criterion. We refer to this simple fixed-sample-size approach as Procedure `F`; see Figure 2.1.

Correctness

In general, we assume `F` has access to a hypothesizer H that always returns hypotheses h from C that correctly classify every observed training example. We call such a hypothesizer H *consistent for* C. Ensuring the correctness of Procedure `F` then is simply a matter of finding an appropriate sample size function $T_{\mathtt{F}}$ that can be proved sufficient to eliminate all ϵ-bad hypotheses with probability at least $1 - \delta$. This is normally accomplished by using well-known results on the uniform convergence of families of frequency-estimates to their true probabilities. For example, it is easy to determine a sufficient sample size for finite concept classes.

Proposition 2.2 *For any* $\epsilon > 0$ *and* $\delta > 0$*, and any* finite *concept class* C*:*

$$T_{\mathit{finite}}(C, \epsilon, \delta) \;=\; \frac{1}{\epsilon} \ln \frac{|C|}{\delta}$$

random examples are sufficient to ensure that, with probability at least $1 - \delta$*, every concept in* C *with error greater than* ϵ *misclassifies at least one training example.*

This is easy to prove: The probability that one ϵ-bad concept correctly classifies t random training examples is at most $(1 - \epsilon)^t$. Since there can be at most $|C|$ ϵ-bad hypotheses in total, the probability that any one of them survives t training examples is at most $|C|(1 - \epsilon)^t$. Therefore, since $(1 - \epsilon)^t \le e^{-\epsilon t}$, we need simply choose a sample size t that is large enough to ensure $|C|e^{-\epsilon t} \le \delta$.

So, using T_{finite}, Procedure F correctly pac-learns any finite concept class. Notice, however, that this bound degenerates for *infinite* concept classes. This is an important limitation in practice, since many applications involve infinite domains which naturally give rise to infinite concept classes (conceptually at least). For example, learning concepts defined on $\mathbb{R}^n$ generally involves concept classes like halfspaces or multilayer-perceptrons which are clearly infinite. It turns out that determining sufficient training sample sizes for infinite concept classes is much more difficult than for the finite case. In fact, the question of whether pac-learning is even possible at all arises in this case.

A central tool of pac-learning theory is a result, due to Vapnik and Chervonenkis, that provides sufficient training sample sizes for many infinite concept classes. Specifically, Vapnik and Chervonenkis [1971] characterize the rate at which the empirical frequency-estimates for arbitrary families of events converge *uniformly* to their true probabilities. Their main contribution is a bound on this rate of uniform convergence that can be expressed in terms of a specific measure of the representational complexity of the family; namely, the Vapnik-Chervonenkis dimension (VCdimension).

Definition 2.3 (Vapnik-Chervonenkis dimension) *The Vapnik-Chervonenkis dimension of a concept class C is defined as follows:*

- *For any two nested subsets S and F of X,* i.e., *such that $S \subset F \subset X$, we say that a concept $c : X \to \{0, 1\}$ "picks S out from F" if $c(x) = 1$ for all x in S and $c(x) = 0$ for all x in $F - S$.*
- *A concept class C is said to* "shatter" *a finite set F if for each of the $2^{|F|}$ subsets S of F, there is a $c \in C$ that picks S out from F.*
- *The* Vapnik-Chervonenkis dimension *of a concept class C, written* $\mathrm{VC}(C)$, *is the size of the largest set shattered by C (defined to be ∞ if no largest such set exists).*

Thus, the VCdimension measures the representational complexity of a concept class C by the maximum number of points that can be independently labelled by choosing concepts from C. To illustrate this, notice that the VCdimension of a *finite* class C can be at most $\log_2 |C|$, since independently labelling a set of d objects requires at least 2^d distinct concepts. The importance of the VCdimension is that it is an abstract measure that applies to arbitrary concept classes defined on arbitrary domains. Moreover, this measure generally gives an intuitive characterization of the effective dimensionality of a concept class C, often corresponding to the number of free parameters used to define the class under its most natural encoding. For example, the class of halfspace concepts defined on $\mathbb{R}^n$ has a VCdimension of $n + 1$ [Pollard, 1984], and is normally defined by a *PERCEPTRON* encoding that also involves $n + 1$ free parameters [Nilsson, 1965].[3] Baum and Haussler [1989] also observe that the VCdimension of the class of multilayer-perceptron concepts roughly corresponds to the number of free parameters used to define the *PERCEPTRON* networks, up to constants (and log factors in some parameters).

Specializing the earlier results of Vapnik and Chervonenkis, Blumer *et al.* [1989] obtain a sample size function T_{BEHW} that can be proved sufficient to ensure F correctly pac-learns a wide range of infinite concept classes.

Theorem 2.4 [Blumer et al., 1989] *For any $\epsilon > 0$, $\delta > 0$, and any (well behaved[4]) concept class C with* $\mathrm{VC}(C) < \infty$*:*

$$T_{BEHW}(C, \epsilon, \delta) = \max\left\{ \frac{8\,\mathrm{VC}(C)}{\epsilon} \log_2 \frac{13}{\epsilon},\ \frac{4}{\epsilon} \log_2 \frac{2}{\delta} \right\}$$

random examples are sufficient to ensure that, with probability at least $1 - \delta$, every concept in C with error greater than ϵ misclassifies at least one training example.

[3] To illustrate this for $\mathbb{R}^2$, notice that any (non-co-linear) set of 3 points on the plane can be shattered by halfspaces, but no set of 4 points can be.

[4] These uniform convergence results assume the concept class C satisfies certain measurability restrictions. This is a benign technical condition that is normally satisfied by all concept classes encountered in practice (however, it is possible to construct classes that violate these assumptions; see [Blumer et al., 1989] for details). We will not be concerned with these issues here. All concept classes we consider will be assumed to be suitably well behaved in this manner.

Thus, using T_{BEHW}, Procedure F can correctly pac-learn any class C, provided only that C has finite VCdimension. This is a general and powerful result as most concept classes normally encountered in practice have finite VCdimension and therefore can be pac-learned by F (for any $\epsilon > 0$, $\delta > 0$). However, not every class of concepts has finite VCdimension; *e.g.*, the class $\mathcal{B}$ of all Borel subsets of $\mathbb{R}$ clearly has infinite VCdimension (since it can shatter infinite sets) and hence Procedure F cannot use T_{BEHW} to pac-learn $\mathcal{B}$. This leaves open the question of whether classes with infinite VCdimension can be pac-learned by *any* fixed-sample-size learning procedure. We address this question in Section 2.2.4 below.

2.2.3 Efficiency

Of course the key issue, aside from designing *correct* pac-learning procedures, is designing *efficient* procedures. We are particularly interested in pac-learning procedures that are both data-efficient and computationally-efficient. That is, procedures that observe as few training examples as possible and produce hypotheses within reasonable computational limits.

Data-efficiency

The data-efficiency of a learning procedure is determined by the number of training examples it observes, regardless of the computation time it uses in producing its hypotheses. Clearly, the data-efficiency of a fixed-sample-size learning strategy is directly determined by the sufficient sample size function it uses. Since any such sample size function must be proved sufficient to eliminate all bad hypotheses with high probability, the best data-efficiency that can be achieved by a fixed-sample-size learner is directly determined by the smallest bound we can *prove* does the job. Since the work of Blumer *et al.* [1989] some effort has been expended towards improving their bound, and recent progress has been made in this direction.

Theorem 2.5 [Shawe-Taylor, Anthony and Biggs, 1993] *For any $\epsilon > 0$, $\delta > 0$, and any well-behaved concept class C with $\mathrm{VC}(C) \geq 2$:*

$$T_{STAB}(C, \epsilon, \delta) \;=\; \frac{1}{\epsilon(1-\sqrt{\epsilon})}\left(2\,\mathrm{VC}(C)\ln\frac{6}{\epsilon} + \ln\frac{2}{\delta}\right)$$

random examples are sufficient to ensure that, with probability at least $1-\delta$, every concept in C with error greater than ϵ misclassifies at least one training example.

This sample size function improves the previous bound T_{BEHW} by roughly a factor of $4/\ln 2$. However, notice that both of these sample size bounds scale as

$$T_{\mathtt{F}}(C, \epsilon, \delta) \;=\; \Theta\left(\frac{1}{\epsilon}\left(\mathrm{VC}(C)\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right) \qquad (2.1)$$

in terms of ϵ, δ and $\mathrm{VC}(C)$, which, interestingly, is linear in $\mathrm{VC}(C)$. Although it remains an open question whether T_{STAB} can be improved further, it is known that its scaling behavior cannot be improved: There are concept classes and (poor) hypothesizers that can force Procedure F to require a number of training examples that meets this scaling bound to within constant factors [Ehrenfeucht et al., 1989; Haussler, Littlestone and Warmuth, 1988]. Therefore, the worst case scaling performance indicated by (2.1) can only be improved upon by considering special hypothesis guessing strategies. (However, in Section 2.2.4 below, we will see there are strong limits even to this.) For the special case of finite concept classes, we note that the sufficient sample size bound T_{finite} often gives better results than T_{STAB}.

Computational-efficiency

Aside from data-efficiency, the other key aspect of a learning system's performance is the computational resources it requires to produce its hypotheses. For Procedure F, the only computational task is to find a concept from C that correctly classifies every training example; *i.e.*, calling H. Efficient computational procedures (*i.e.*, hypothesizers) have been developed for finding consistent concepts from many different families of concept classes, including:

- $k$dnf concepts defined on $\{0,1\}^n$ [Valiant, 1984],
- decision-list concepts defined on $\{0,1\}^n$ [Rivest, 1987], and
- halfspace concepts defined on $\mathbb{R}^n$ [Blumer et al., 1989].

These procedures all run in time polynomial in the input dimension n, and the number of training examples (which, in turn, is polynomial in n, $1/\epsilon$, and $1/\delta$).[5] However, it is not always easy to find efficient procedures for such a task. In fact, there are many concept class families where the problem of finding a consistent concept is known to be *NP*-hard.[6] For example, for the class of k-term-cnf concepts defined on $\{0,1\}^n$, the problem of finding a k-term-cnf concept consistent with a polynomial number of training examples is *NP*-hard [Pitt and Valiant, 1988].

Sometimes in these cases a neat trick due to Pitt and Valiant [1988] can be used to circumvent this difficulty: Rather than find a consistent concept from C, we instead find a consistent concept from some superclass $C^H \supset C$, where C^H makes the computational task easy but does not require too many additional training examples to meet the pac-criterion. For example, even though finding a consistent k-term-cnf concept is *NP*-hard, we can notice that $k$dnf $\supset$ k-term-cnf and use the efficient procedure for finding consistent $k$dnf concepts to pac-learn k-term-cnf in polynomial time [Pitt and Valiant, 1988]. Although this comes at the expense of a slight increase in the number of training examples used (since $k$dnf is a strictly larger class), we substantially reduce the computational costs of pac-learning. Thus, the idea is to trade-off a slight loss in data-efficiency for a significant gain in computational-efficiency. In general, this trick will work whenever we can find a suitable superset of the concept class $C^H \supset C$. However, we will see below that this is not likely to be possible in every situation.

2.2.4 Complexity

Aside from determining the correctness and efficiency of specific learning procedures, it is also important to consider the intrinsic difficulty of solving pac-learning *problems*; *i.e.*, determining the minimum data and computational resources required by *any* learning procedure to meet the pac-criterion in the worst case. As before, the complexity of a pac-learning problem can be measured along two orthogonal dimensions: data-complexity and computational-complexity.

Data-complexity

A number of theoretical results have been obtained that characterize the inherent data-complexity of pac-learning, ignoring computational costs. These analyses consider the minimum number of training examples required by any learning procedure to meet the pac-guarantees in the worst case.

Definition 2.6 ((Fixed) data-complexity) *The (fixed)* data-complexity *of a pac-learning problem (C, ϵ, δ) is the smallest number of training examples required by any (fixed-sample-size) learning procedure to meet the pac(ϵ, δ)-criterion for every target concept c in C and every domain distribution* P_X.

This type of analysis is important because it allows us to compare the data-efficiency of particular learning procedures with the optimum possible data-efficiency, thus determining whether these procedures can be improved, and if so, by how much. Just such an analysis has been carried out for pac-learning problems by Ehrenfeucht *et al.* [1989].

Theorem 2.7 [Ehrenfeucht et al., 1989] *For any $0 < \epsilon \le 1/8$, $0 < \delta \le 1/100$, and any concept class C with $\mathrm{VC}(C) \ge 2$: any (fixed-sample-size) learning procedure that observes fewer than*

$$t_{\scriptscriptstyle BHKV}(C, \epsilon, \delta) \quad = \quad \max\left\{ \frac{\mathrm{VC}(C) - 1}{32\epsilon}, \; \frac{1-\epsilon}{\epsilon} \ln \frac{1}{\delta} \right\}$$

[5] Note that an analysis of computational-efficiency generally considers how the cost of solving a parameterized *family* of problems scales-up in terms of some size parameter n. Here we are addressing how the cost of solving a family of pac-learning problems $\{(C_n, \epsilon, \delta)\}_{n=1}^{\infty}$ defined on domains $\{X_n\}_{n=1}^{\infty}$ scales-up in terms of n, $1/\epsilon$, and $1/\delta$.

[6] It widely believed there are no polynomial time algorithms for such problems.

random training examples will fail to meet the pac(ϵ, δ)-criterion for some *target concept $c \in C$ and* some *domain distribution* P_X.

This shows that the VCdimension not only provides an upper bound on the number of training examples F needs to pac-learn a concept class C, it also provides a nontrivial lower bound on the number of training examples required by any fixed-sample-size learning procedure to pac-learn C. From this result we can see that finite VCdimension is not only sufficient, but also necessary for any fixed-sample-size learning procedure to correctly pac(ϵ, δ)-learn C. (Thus, proving that the previous example of the class $\mathcal{B}$ of all Borel set (indicators) on $\mathbb{R}$ cannot be pac(ϵ, δ)-learned for any $\epsilon > 0$, $\delta > 0$, by any fixed-sample-size learning procedure.) Consequently F can be seen as a "universal" learning procedure in the sense that it pac-learns any concept class C for which this is possible by any fixed-sample-size learning procedure.

Notice that this lower bound $t_{\scriptscriptstyle BHKV}$ scales as

$$t_{\scriptscriptstyle BHKV}(C, \epsilon.\delta) \;=\; \Theta\left(\frac{1}{\epsilon}\left(\mathrm{VC}(C) + \ln\frac{1}{\delta}\right)\right)$$

in terms of $\mathrm{VC}(C)$, ϵ and δ, which matches the scaling behavior of $T_{\scriptscriptstyle BEHW}$ and $T_{\scriptscriptstyle STAB}$ up to constant factors and a $\ln(1/\epsilon)$ term. Therefore, not only is F a universal learning procedure, it also has near-optimal scaling behavior, in that no other fixed-sample-size learning procedure can improve on its data-efficiency by more than a constant (and $\ln(1/\epsilon)$) factor.

Overall, these results show that VCdimension is a powerful measure of concept class complexity, as it tightly determines the data-efficiency with which a concept class C can be pac-learned by any fixed-sample-size learning procedure.

Computational-complexity

Aside from the minimum number of training examples needed for pac-learning, it is also important to consider the minimum computational resources required to produce an acceptable hypothesis. As noted above, Procedure F can be efficiently implemented for many concept classes (like $k$dnf) rather easily, but special tricks are required to implement F efficiently for other concept classes (like k-term-cnf). This raises the question of whether there are families of concept classes that have no efficient learning procedures.

Definition 2.8 (Feasible learnability) *A family $\{C_n\}$ of concept classes is* feasibly pac-learnable *if there is a learning algorithm L, taking n, ϵ, and δ as input, that solves the pac-learning problems (C_n, ϵ, δ) with training sample size and running time bounded by a polynomial in n, $1/\epsilon$, and $1/\delta$.*

Proving that a family *is* feasibly pac-learnable is simply a matter of showing $\{C_n\}$ has polynomial VCdimension and demonstrating a polynomial-time hypothesizer H that finds consistent hypotheses from a suitable class $C_n^H \supset C_n$; *e.g.*, as for $k$dnf $\supset$ k-term-cnf. However, proving that a family of concept classes *cannot* be feasibly pac-learned (no matter how many tricks one tries) is a significant challenge. In fact, this has yet to be accomplished for any family of concept classes with polynomial VCdimension. However, Kearns and Valiant [1989] have been able to show that, given standard assumptions about the hardness of certain cryptographic tasks, there are families of concept classes (*e.g.*, boolean-formulae) that cannot be feasibly pac-learned, unless these assumptions turn out to be false.

2.2.5 Assessment

The theory of pac-learning has greatly increased our understanding of the difficulty of classification learning. Not only do these theoretical results describe the limits of what can be achieved in terms of data and computational-efficiency, they also prescribe how to learn with guaranteed accuracy, reliability, and efficiency, whenever this is possible. Particularly strong are the results that show these procedures achieve near-optimal data-efficiencies in terms of how they scale-up in the desired accuracy, reliability, and complexity of the prior concept class.

The most important aspect of this theory is that it treats prior knowledge explicitly. The theory attempts to characterize optimal learning performance given whatever prior knowledge is available, and quantifies the strength of this prior knowledge by the effect it has on learning performance. It turns out that a suitable measure of the prior knowledge embodied by a concept class C is given by its VCdimension.

For $(X = \mathbb{R}^{10}, C = \textsf{halfspaces}, \epsilon = 0.01, \delta = 0.05)$:		
T_{BEHW}	$=$	$91,030$
T_{STAB}	$=$	$15,981$
T_{thumb}	$\approx$	$1,100$
t_{EHKV}	$=$	32

Table 2.1: Comparing training sample sizes

2.3 Issue

However, despite the apparent strength of these results, pac-learning theory has arguably had little impact on the practice of machine learning. Although machine learning practitioners are generally aware of the (main) results, they rarely consider the specific prescriptions when tackling real applications. Why?

It is certainly possible to criticize many of the specific modelling assumptions made by pac-learning theory (*e.g.*, bivalent classification, example independence, noise free classification, *etc.*), arguing that these do not address the real situations encountered in practice. However, pac-learning theory has addressed many of these objections at one time or another (*cf.* [Ben-David, Cesa-Bianchi and Long, 1992; Haussler, 1992; Angluin and Laird, 1988; Kearns and Li, 1988]) and yet still is not adopted in practice.

The key reason that pac-learning theory is ignored in practice over and above these concerns is *data-efficiency.* That is, in spite of their near-optimal scaling behavior, the actual training sample sizes demanded by current sample size bounds are far too large to be practical. This is easily demonstrated by a simple example.

Example

Consider a typical example of learning a halfspace concept defined on $\mathbb{R}^n$. Here, each domain object is described by a vector of n real-valued attributes, and the target concept is a half-space of $\mathbb{R}^n$ defined by some unknown hyperplane and direction. As previously noted, this concept class is naturally defined by $n+1$ free parameters and has VCdimension $n+1$. A typical (small) machine learning application might involve object descriptions consisting of about 10 object attributes, and we might be interested in achieving an error of, say, 1% with reliability at least 95%; giving a pac-learning problem $(X = \mathbb{R}^{10}, C = \textsf{halfspaces}, \epsilon = 0.01, \delta = 0.05)$.

Given these modest requirements it is easy to determine a sufficient sample size for Procedure F to solve this problem: Simply note that the class of halfspace concepts on $\mathbb{R}^{10}$ has VCdimension 11, and then plug the specified parameters into T_{BEHW} (or T_{STAB}). But here we find that T_{BEHW} demands $91,030$ training examples! Even the improved bound T_{STAB} demands $15,981$ training examples in this case. These seem like outrageous training sample sizes given the apparently reasonable parameter settings. Moreover, these results compare quite poorly to the empirical "rule of thumb" that, for a concept class defined by w free parameters, roughly $T_{thumb} = w/\epsilon$ training examples should be required to achieve an error of ϵ [Hinton, 1989; Baum and Haussler, 1989]. Applied here, T_{thumb} demands only $1,100$ training examples—an order of magnitude fewer than T_{STAB}. Of course, this rule of thumb comes with no guarantees, but it does give a general indication of how many training examples practitioners would deem reasonable for this problem.

Further evidence of the weakness of T_{BEHW} and T_{STAB} is obtained by considering the necessary sample size bound t_{EHKV} in this case, which specifies the minimum number of training examples that can be proved absolutely necessary to meet the pac-criterion. Here, t_{EHKV} turns out to require only 32 training examples! So, although the theoretical upper and lower bounds share nearly the same scaling properties, they give results that are orders of magnitude apart in practice. Moreover, these bounds are an order of magnitude away from the empirically supported rule of thumb. Table 2.1 gives a direct comparison.

Impact

The weakness of these sample size bounds has drastic consequences for the practical applicability of the theory since, in practice, training data, not computation time is usually the critical resource. (This is because correctly labelled training examples usually come at non-negligible cost and it is critical that they be conserved.) So multiplying the training sample size merely to account for theoretical slop in the previous bounds leads to results that are grossly unacceptable in practice.

This fact demands a different emphasis than the one adopted by most pac-learning research. To the theoretical learning community, with its emphasis on determining what can be done in polynomial time, the fact that the upper and lower bounds are within constant and logarithmic factors of each other provides little incentive to find improvements. However, the specific constants in these bounds matter a great deal in practice. For example, cutting required training sample sizes in half would be a significant improvement, even if this came at the expense of a slight increase in overall computation time. (Note that this is the opposite trade-off to that considered by Pitt and Valiant [1988].)

Explanations

There are a number of reasons why current fixed-sample-size bounds are impractical.

1. *The existing analysis is fundamentally weak and the current bounds can be drastically improved.*

Strong evidence for this is given by the substantial difference between T_{STAB} and t_{EHKV}. However, the prevailing view among machine learning practitioners is that impractical training sample sizes are inevitable consequences of the worst case pac-learning guarantees.

2. *Achieving the pac-criterion in the worst case over all possible target concepts and domain distributions is just too hard.*

In fact, this is the most commonly held view: the worst case bounds are inherently unreasonable because they must take into account "pathological" domain distributions that force large sample sizes—moreover, the argument continues, these pathological distributions do not arise in "typical" applications. However, this line of reasoning is really quite weak: First of all, no-one has yet been able to prove that these pathological distributions really exist (for this would be tantamount to improving the lower bound result t_{EHKV}). Moreover, even if they did exist, knowing for certain that pathological distributions do not arise in typical applications constitutes a strong form of prior (meta) knowledge, which is probably best dealt with explicitly; *e.g.*, as in Chapter 3 below.

Approach

The work reported in this chapter investigates an alternative view:

3. *Perhaps the simplistic fixed-sample-size learning strategy* **F** *(collect a sufficient training sample; then find a consistent hypothesis) does not make the most efficient use of the available training data.*

This view raises the question of whether alternative learning strategies might be more data-efficient than **F** in practice. This constitutes the starting point of the research reported here. Specifically, we consider learning strategies that are based on a *sequential* decision making approach, where the learner is allowed to observe the training examples one at a time and autonomously decides after each whether to stop and produce a hypothesis, or to continue training. The hope is that this extra leverage can be exploited to reduce the number of training examples observed, while maintaining the same worst case pac-guarantees.

One assumption behind this approach is that we are willing to incur a slight computational cost in order to obtain a significant improvement in data-efficiency. Again, this trade-off is motivated by the fact that, in most practical learning applications correctly labelled training data is the critical resource.

2.4 *Sequential* pac-learning

We now consider the idea of pac-learning a concept class C by using on-line stopping-rules that decide when to stop training by observing the effects of the training sequence on C. The challenge is to find an appropriate stopping-rule that maintains the pac-criterion while observing as few training examples as possible. In this section, we develop a few simple sequential learning procedures that (*i*) are provably correct pac-learners (*ii*) are provably data-efficient (improving on the existing fixed-sample-size bounds in some cases), and (*iii*) use many times fewer training examples in practice—often achieving what appears to be near-practical data-efficiency in empirical case studies. We also assess the inherent data-complexity of sequential pac-learning.

2.4.1 Problem

This section addresses the same pac-learning problem introduced in Section 2.2: given a concept class C and specified accuracy and reliability parameters ϵ and δ, we demand that the learner produce an ϵ-accurate hypothesis with probability at least $1-\delta$ for any target concept $c \in C$ and domain distribution P_X. Here, however, we permit the learner to choose the size of its own training sample based on the specific sequence of examples it receives. Thus, we re-formalize a learner L as consisting of:

1. A *stopping rule* $T_L(C,\epsilon,\delta) : (X \times \{0,1\})^\infty \to I\!N$ that maps training sequences to stopping times (where the event $\{T_L = t\}$ depends only on the first t training examples).
2. A *hypothesizer* $H_L(C,\epsilon,\delta) : (X \times \{0,1\})^* \to \{0,1\}^X$ that maps finite training sequences to hypotheses.

Since the number of training examples a sequential learner observes depends on the specific training sequence, its training sample size is a *random variable* rather than a fixed number. So a sequential learner's data-efficiency is fully described by a distribution of training sample sizes. Although there are several ways to characterize a learner's efficiency by the shape of its sample size distribution, we will focus on arguably the most natural measure: the average (*i.e.*, expected) training sample size. To compare the relative data-efficiency of a sequential to a fixed-sample-size learner (which entails comparing a distribution to a fixed number) we will simply compare the expected and fixed training sample sizes directly. Our goal is to develop sequential pac-learning procedures that observe a small average number of training examples in the worst case over all possible target concepts in C and domain distributions P_X. We ask whether sequential learning procedures can use a smaller average number of training examples than existing fixed-sample-size learning procedures while maintaining the same worst case pac-learning guarantees.

2.4.2 Procedures

Why do we expect sequential learning to reduce the number of training examples needed to pac-learn? Perhaps the simplest illustration of this is the notion of "premature convergence": Suppose that while attempting to learn an unknown target concept from a class C we happen to notice, well before the sufficient fixed-sample-size bounds have been reached, that only a single concept c in C correctly classifies every training example. Clearly there is no need to continue training, since by assumption c must actually *be* the target concept—thus immediately returning c automatically satisfies the pac-criterion. So halting as soon as the target concept is identified can sometimes substantially reduce the number of training examples observed. Of course, this naive strategy does not work in general: if the domain distribution does not distinguish between two distinct target concepts, then neither concept will be uniquely identified after any finite number of training examples (with probability 1). However, it turns out that more sophisticated versions of this premature convergence idea can be made to work in general.

The general approach we will take is the following: We assume that we have some hypothesizer H that is consistent for C (*i.e.*, H always returns a hypothesis h from C that correctly classifies every training example, provided the examples are consistent with some $c \in C$). This hypothesizer will be called as a subroutine in our learning procedures. The basic strategy will be to call H to obtain an initial hypothesis h_0, and then sequentially observe training examples $\langle x_1, c(x_1)\rangle$, $\langle x_2, c(x_2)\rangle$, ..., *etc.* Whenever the most recent hypothesis, say h_i, misclassifies the current training example, $\langle x_t, c(x_t)\rangle$, we will call H to obtain a new hypothesis h_{i+1} that is consistent with all observed training examples $\langle x_1, c(x_1)\rangle, ..., \langle x_t, c(x_t)\rangle$. In this way we will generate a sequence of hypotheses $h_0, h_1, ...$ *etc.*, where each subsequent hypothesis is consistent with a greater initial segment of the training sequence. The only question then is, how do we decide when one of these hypotheses has correctly classified enough training examples to safely be returned as the final hypothesis?

Obvious approach

The most obvious approach to sequential pac-learning is based on the idea of "repeated survival testing": Instead of just waiting for premature convergence, we observe the series of hypotheses $h_0, h_1, ...$ produced by

Procedure R $(C, \epsilon, \delta;\ H)$

INPUT: target concept class C,
accuracy parameter $\epsilon > 0$,
reliability parameter $\delta > 0$;
a consistent hypothesizer H for C.

RETURN: a hypothesis h with accuracy at least $1 - \epsilon$, with probability at least $1 - \delta$.

PROCEDURE:

- Fix an arbitrary sequence $\{\delta_i\}_{i=1}^{\infty}$ such that $\delta_i > 0$ and $\sum_{i=1}^{\infty} \delta_i = \delta$.
- Obtain an arbitrary initial hypothesis h_0 from H.
- Sequentially observe training examples $\langle x_t, c(x_t)\rangle$, $t = 1, 2, \ldots$, *etc.*, labelled by some unknown target concept $c \in C$:
    - If the current hypothesis h_i has correctly classified $T_{\mathit{finite}}(\{h_i\}, \epsilon, \delta_i) = \frac{1}{\epsilon} \ln \frac{1}{\delta_i}$ consecutive training examples since its inception: halt and return h_i.
    - If the current hypothesis h_i misclassifies a training example: discard h_i, call H to generate a new consistent hypothesis h_{i+1}, and begin testing h_{i+1}.
- Repeat until some h_i passes the test.

Figure 2.2: Procedure R

H and accept the first hypothesis that correctly classifies a sufficient number of consecutive training examples. Any hypothesis that makes a mistake is discarded in favor of a new consistent hypothesis. Specifically, Procedure R (Figure 2.2) takes as a subroutine an arbitrary hypothesizer H that produces consistent concepts from C. Whenever the current hypothesis h_i misclassifies a training example, R calls H to generate a new consistent hypothesis h_{i+1}. The first hypothesis h_i that correctly classifies a sufficient number of consecutive training examples (as specified by $T_{\mathit{finite}}(\{h_i\}, \epsilon, \delta_i)$) is returned as the final hypothesis; see Figure 2.2.[7]

It is not hard to see that R returns an ϵ-bad hypothesis with probability at most δ, since by construction the probability of accepting hypothesis h_i, if ϵ-bad, is bounded by δ_i. Thus, the total probability of accepting any ϵ-bad hypothesis is bounded by $\sum_{i=1}^{\infty} \delta_i = \delta$. That is, we carry out an unbounded number of tests, but perform each with increasing reliability to ensure that the overall probability of making a mistake is bounded by δ. To show R meets the pac-criterion, then all we need to do is prove that R eventually accepts some hypothesis with probability 1 (wp1), since this will then guarantee that R returns an acceptable hypothesis with probability at least $1-\delta$. It turns out that R can be proved to halt wp1 for *any* concept class C, provided only that the target class C has finite VCdimension and the hypothesizer H produces only consistent concepts from C (see Theorem 2.9 below).

However, even though Procedure R is a reasonable approach to sequential pac-learning, it does not prove to be particularly data-efficient in theory. The problem is that R can spend too much time rejecting "good enough" hypotheses. That is, R rejects hypotheses of error ϵ with high probability (greater than $1 - \delta$) even though such hypotheses are perfectly acceptable. Moreover, R takes a long time to reject these borderline hypotheses (*i.e.*, with expected time close to $1/\epsilon$). Thus, if H produces a series of such hypotheses, R will take an unacceptably long time to terminate (expected time about $1/(\epsilon\delta)$, which is not very good). This prevents us from proving good bounds on R's data-efficiency—unless we incorporate additional assumptions

[7]This basic "repeated survival testing" strategy has been independently proposed by many researchers, *e.g.*, [Kearns et al., 1987b; Angluin, 1988; Benedek and Itai, 1988b; Linial, Mansour and Rivest, 1991; Oblow, 1992]—primarily for achieving "non-uniform" pac-learning. Here we consider the same strategy, but for a different goal: we seek a *uniform* improvement in sample size over all targets in C, whereas non-uniform pac-learning attempts to increase the range of "pac-learnable" concept classes by sacrificing a uniform bound on data-efficiency. These two concerns are orthogonal, as discussed in Section 2.7.2 below.

about H, or somehow argue that H cannot produce an unbounded sequence of consistent hypotheses with ϵ error. However, rather than pursue a complicated analysis we take a different approach: by explicitly considering `R`'s shortcomings, we develop an alternative strategy that circumvents these difficulties. This leads to a novel approach that proves to be more data-efficient than `R`, both in theory and practice.

Better approach

Here we propose a new sequential learning strategy, Procedure `S` (Figure 2.3), that is also based on repeated significance testing, but attempts to avoid the inefficiency of `R`'s survival testing approach. `S` is based on two ideas: First, instead of discarding hypotheses after a single mistake, `S` saves H's hypotheses in a list $\{h_0, h_1, ...\}$, and continues testing each one until some hypothesis proves to have sufficiently small empirical error. Second, `S` identifies an accurate candidate in the list by using an on-line *sequential probability ratio test* (`sprt`) [Wald, 1947] to test each hypothesis, returning the first hypothesis that passes the test (see Figure 2.3). In this way, `S` never rejects a potentially acceptable hypothesis while quickly identifying any accurate candidate on the list.

Procedure `S` is a correct pac-learner in the exact same sense as `F` and `R`: it is guaranteed to return an ϵ-accurate hypothesis with probability at least $1-\delta$ (provided $\mathrm{VC}(C) < \infty$ and H is consistent for C). The key property of `S` is that its call to `sprt` is guaranteed to accept any ϵ/κ-good hypothesis wp1, but only accepts an ϵ-bad hypothesis h_i with probability at most δ_i. In this way `S` returns an ϵ-bad hypothesis with probability at most $\sum_{i=1}^{\infty} \delta_i = \delta$ and yet eventually returns some hypothesis wp1, provided only that H eventually produces an accurate candidate. Thus, proving that `S` meets the pac-criterion is a simple matter of showing that H eventually must produce an ϵ/κ-accurate hypothesis wp1.

Theorem 2.9 (Correctness) *For any $\epsilon > 0$, $\delta > 0$, and any (well behaved) concept class C with $\mathrm{VC}(C) < \infty$: Given i.i.d. examples generated by any distribution P_X and target concept $c \in C$, Procedure `S` (`R`) returns a hypothesis h such that $\mathrm{P}_X\{h(x) \neq c(x)\} \leq \epsilon$ with probability at least $1-\delta$; using any hypothesizer H that is consistent for C.*[8]

[8] Proofs of all (original) results stated in this chapter are given in Appendix A.

Procedure `S` $(C, \epsilon, \delta;\ H, \kappa)$

INPUT: target concept class C,
accuracy parameter $\epsilon > 0$,
reliability parameter $\delta > 0$;
a consistent hypothesizer H for C,
and an arbitrary constant $\kappa > 1$ (see explanation below).

RETURN: a hypothesis h with accuracy at least $1 - \epsilon$, with probability at least $1 - \delta$.

PROCEDURE:

- Fix an arbitrary sequence $\{\delta_i\}_{i=1}^{\infty}$ such that $\delta_i > 0$ and $\sum_{i=1}^{\infty} \delta_i = \delta$.
- Obtain an arbitrary initial hypothesis h_0 from H.
- Sequentially observe training examples $\langle x_t, c(x_t)\rangle$, $t = 1, 2, ...,$ *etc.*, labelled by some unknown target concept $c \in C$:
  - Collect H's hypotheses in a list $\{h_0, h_1, ...\}$.
  - If the most recent hypothesis h_i misclassifies a training example: call H to obtain a new consistent hypothesis h_{i+1}, and add h_{i+1} to the end of the list.
  - Once added, begin testing hypothesis h_i on *subsequent* training examples.
  - In parallel, subject each hypothesis h_i in the list to a statistical test that accepts h_i, if ϵ-bad, with probability at most δ_i.

    (Note that there are many ways to implement this test. The specific way we will do it is to fix an arbitrary constant $\kappa > 1$ and test whether h_i's error is below some threshold ϵ/κ, or above ϵ, with a probability of incorrectly deciding that h_i's error is less than ϵ/κ (when in fact it is greater than ϵ) bounded by δ_i. We will make this decision *on-line* by using a sequential probability ratio test (`sprt`); in particular, we will call `sprt(` $h_i(x) \neq c(x)$, ϵ/κ, ϵ, δ_i, 0 `)` to test each hypothesis h_i; see Figure 2.4.[9] The key property of this call is that if h_i is ϵ-bad, `sprt` accepts it with probability at most δ_i, but if h_i is ϵ/κ-good, then `sprt` accepts it with probability 1.[10])
  - If some h_j in the current list is accepted by the test: halt and return h_j.
- Repeat until some h_j passes the test.

Figure 2.3: Procedure `S`

[9] Another way to implement this test would be to fix a training sample size that is sufficient to give a $(1-\delta_i)$–confident $p_i + \epsilon(1 - 1/\kappa)$ upper error bar on h_i's true error p_i (by using Chernoff bounds [Hagerup and Rüb, 8990]) and detecting whether this error bar is below ϵ. However, `sprt` is more efficient than this.

[10] Note that here κ serves as a tradeoff parameter between *(a)* the time it takes to find an ϵ/κ-accurate hypothesis, and *(b)* the time to accept such a hypothesis (wp1) once found. For example, choosing a small value of κ (near 1) reduces (a) but increases (b). In Section 2.5 below, we choose a value of κ that balances between these two factors in a convenient (but not necessarily optimal) way.

Procedure **sprt** (ϕ, a, r, δ_{acc}, δ_{rej}) [Wald, 1947]

INPUT: boolean random variable $\phi : X \to \{0,1\}$,
acceptance boundary a, $0 \le a < 1$,
rejection boundary r, $a < r \le 1$,
acceptance reliability parameter $\delta_{acc} \ge 0$,
rejection reliability parameter $\delta_{rej} \ge 0$.

DECIDE: $H_{acc} : \mathrm{P}_X\{\phi(x) = 1\} \le a$ versus $H_{rej} : \mathrm{P}_X\{\phi(x) = 1\} \ge r$;
returning "H_{acc}" with probability at most δ_{acc} when H_{rej} true, and
returning "H_{rej}" with probability at most δ_{rej} when H_{acc} true.

PROCEDURE:

- Sequentially observe $\phi_t = \phi(x_t)$ for $t = 1, 2, ...$, and monitor the sum

$$S_t(\boldsymbol{\phi}^t) = \sum_{\phi_i \in \boldsymbol{\phi}^t} \phi_i \ln \frac{a}{r} + (1 - \phi_i) \ln \frac{1-a}{1-r}.$$

  - Return "H_{acc}" if ever $S_t(\boldsymbol{\phi}^t) \ge \ln(1/\delta_{acc})$.
  - Return "H_{rej}" if ever $S_t(\boldsymbol{\phi}^t) \le \ln(\delta_{rej})$.

- Continue to make observations while $\ln(\delta_{rej}) < S_t(\boldsymbol{\phi}^t) < \ln(1/\delta_{acc})$.[11]

Figure 2.4: Procedure sprt

[11]To briefly explain this procedure: Note that we have two alternative probability models for a random sequence of i.i.d. bits: Model A says that $\mathrm{P}(\phi = 1) = a$ and Model R says $\mathrm{P}(\phi = 1) = r$ (for simplicity we are focusing on the boundary cases here). Now, assume that we observe a sequence of t bits $\boldsymbol{\phi}^t = \langle \phi_1, ..., \phi_t \rangle$. Then the probability of observing this sequence under Model A is $\mathrm{P}_a^t\{\boldsymbol{\phi}^t\} = \prod_{i=1}^t a^{\phi_i}(1-a)^{1-\phi_i}$, and the probability of observing this sequence under Model R is $\mathrm{P}_r^t\{\boldsymbol{\phi}^t\} = \prod_{i=1}^t r^{\phi_i}(1-r)^{1-\phi_i}$. (These are called the *likelihoods* of the sequence $\boldsymbol{\phi}^t$ under Models A and R respectively.)

What Procedure sprt does is, in effect, monitor the ratio of these two likelihoods, $R_t(\boldsymbol{\phi}^t) = \mathrm{P}_a^t\{\boldsymbol{\phi}^t\}/\mathrm{P}_r^t\{\boldsymbol{\phi}^t\}$, and decide "$A$" if the probability of $\boldsymbol{\phi}^t$ under Model R is less than δ_{acc} times the probability of $\boldsymbol{\phi}^t$ under Model A (or decide "R" if the probability of $\boldsymbol{\phi}^t$ under Model A is less than δ_{rej} times the probability of $\boldsymbol{\phi}^t$ under Model R); continuing to make observations while neither condition is satisfied. To implement this procedure however, it is convenient to monitor the *log* of the likelihood ratio, $S_t(\boldsymbol{\phi}^t) = \ln R_t(\boldsymbol{\phi}^t)$, and hence turn the product into a sum. A proof that sprt correctly meets the stated reliability criteria, along with a detailed analysis of its stopping time, is given in Section A.1 of Appendix A.

The advantage Procedure `S` holds over Procedure `R` is that we can prove a reasonable upper bound on `S`'s expected training sample size. In addition, `S` proves to be more data-efficient than `R` in empirical case studies, as shown in Section 2.5 below.

2.4.3 Efficiency

We have seen (Theorem 2.7) that no fixed-sample-size learner can improve on the data-efficiency of T_{STAB} by more than a constant factor (logarithmic in $1/\epsilon$)—where data-efficiency is measured by the maximum training sample size used in the worst case. However, merely considering the maximum number of training examples a sequential learner might observe is not an accurate measure of its overall data-efficiency. A sequential learner might only observe large training samples with small probability and so only use a small number of examples on average. A far more natural measure of a sequential learner's data-efficiency is its expected rather than maximum training sample size.

Definition 2.10 (Expected data-efficiency) *The* expected data-efficiency *of a sequential learner L, for solving a pac-learning problem (C, ϵ, δ), is given by the maximum expected training sample size L uses in the worst case over all possible target concepts $c \in C$ and domain distributions* P_X.

Given this more natural measure of data-efficiency, it is possible to ask whether sequential learning can obtain any advantage over fixed-sample-size learning in terms of worst case expected sample size. Here we derive a reasonable upper bound on `S`'s expected training sample size under the weak assumption that C has finite VCdimension and `S` has access to a consistent hypothesizer H for C. As before, this bound can be expressed in terms of ϵ, δ, and $\mathrm{vc}(C)$.

Theorem 2.11 (Data-efficiency) *For any $\delta > 0$, sufficiently small $\epsilon > 0$, and any (well behaved) concept class C with finite VCdimension: Given i.i.d. examples generated by any distribution P_X and target concept $c \in C$, Procedure `S` observes an average training sample size of at most*

$$\mathrm{E}\, T_{\mathtt{S}}(C, \epsilon, \delta) \;\leq\; \left(\frac{\kappa}{\kappa - 1 - \ln \kappa}\right) \frac{1}{\epsilon} \left([2.12\, \kappa\, \mathrm{vc}(C) + 3] \ln \frac{14\kappa}{\epsilon} + \ln \frac{1}{\delta} \right); \tag{2.2}$$

using any hypothesizer H that produces consistent concepts from C, any constant $\kappa > 1$, and the sequence $\{\delta_i = 6\delta/(\pi^2 i^2)\}_{i=1}^{\infty}$ (which gives $\sum_{i=1}^{\infty} \delta_i = \delta$).

Although this bound is somewhat loose, interestingly it still exhibits the same scaling as T_{BEHW} and T_{STAB} in terms of ϵ, δ, and $\mathrm{vc}(C)$:

$$\mathrm{E}\, T_{\mathtt{S}}(C, \epsilon, \delta) \;=\; O\left(\frac{1}{\epsilon}\left(\mathrm{vc}(C) \ln \frac{1}{\epsilon} + \ln \frac{1}{\delta}\right)\right).$$

That is, `S`'s expected data-efficiency scales no worse than `F`'s (fixed) data-efficiency. Of course, the specific constants of scaling are of utmost importance in practical applications. To this end we can directly compare the upper bound (2.2) to the fixed sample size bounds T_{BEHW} and T_{STAB}. Here we note that the bound (2.2) actually beats T_{BEHW} and T_{STAB} for small reliability levels.

Proposition 2.12 (Comparison)
$\mathrm{E}\, T_{\mathtt{S}}(C, \epsilon, \delta) < T_{BEHW}(C, \epsilon, \delta)$ *for $\kappa \geq 3.5$ and sufficiently small $\delta = \epsilon^{\Theta(\mathrm{vc}(C))}$.*
$\mathrm{E}\, T_{\mathtt{S}}(C, \epsilon, \delta) < T_{STAB}(C, \epsilon, \delta)$ *for $\kappa \geq (2/\sqrt{\epsilon}) \ln(2/\sqrt{\epsilon})$ and sufficiently small $\delta = \epsilon^{\Theta(\kappa \cdot \mathrm{vc}(C))}$.*

Unfortunately this advantage is only slight, and only holds for high reliability levels. Overall, the bound (2.2) does not appreciably dominate the efficiency of `F`, nor do the fixed-sample-size bounds T_{BEHW} and T_{STAB} dominate (2.2).[12]

[12] The weakness of this bound (2.2) is not too surprising since the upper bound on the expected number of training examples needed to rule out all ϵ/κ-bad concepts is directly obtained from the existing fixed-sample-size bound T_{STAB}. Clearly, the expected time to eliminate all ϵ/κ-bad concepts is only smaller than the fixed time needed to eliminate all ϵ-bad concepts with probability $1 - \delta$ for very small values of δ.

Although this might not seem like much of an advantage at first, an important property of Procedure `S` is that we expect it to perform much better in practice than any bounds we can prove about its performance *a priori*. This is because the number of training examples `S` observes in any particular application is only determined by the specific case at hand, not the worst case situation (and certainly not by what we can *prove* about the worst case). Therefore, we expect `S` to be far more data-efficient in practice than the loose upper bound (2.2) would indicate. Note that this is a property of sequential learning that the fixed-sample-size approach does not share. The number of training examples observed by Procedure `F` is always determined by the worst case—in fact, by what we can *prove* about the worst case, which is usually much worse. In Section 2.3 we saw that this can lead to training sample sizes that are far larger than intuitively reasonable; *cf.* Table 2.1. So not only do we expect `S` to perform much better than the crude upper bound (2.2) but also much better than the fixed-sample-size bounds T_{BEHW} and T_{STAB}. In fact, a series of empirical case studies in Section 2.5 below show that `S` observes many times fewer training examples than T_{BEHW} or T_{STAB} in practice, even while maintaining the exact same worst case pac-guarantees.

However, before demonstrating `S`'s advantage in empirical tests, we first note that there are inherent limits even to the data-efficiency of sequential learning. These limits are revealed by considering the inherent data-complexity of solving pac-learning problems by sequential learning.

2.4.4 Complexity

Beyond considering the data-efficiency of specific learning procedures (like `S`) it is also important to consider the inherent data-complexity of solving pac-learning *problems*; *i.e.*, the minimum number of training examples required by *any* sequential procedure to meet the worst case pac-guarantees. Here we generalize the original definition of data-complexity from Section 2.2 to consider the minimum *average* number of training examples needed to meet the pac-criterion.

Definition 2.13 (Expected data-complexity) *The* expected data-complexity *of a pac-learning problem (C, ϵ, δ) is given by the smallest* average *number of training examples any learning procedure must observe to meet the pac(ϵ,δ)-criterion for any fixed $c \in C$ and P_X, in the worst case over all possible target concepts $c \in C$ and domain distributions P_X.*

Investigating the inherent data-complexity of pac-learning problems allows us to determine the extent to which sequential learning might ultimately improve on the data-efficiency of fixed-sample-size learning, and how significant this advantage might possibly be. Here, we derive a lower bound on the expected number of training examples any learner must observe to meet the pac-learning guarantees in the worst case over all possible target concepts in C and all possible domain distributions P_X. This bound is expressed in terms of ϵ, δ and the VCdimension of C.

Theorem 2.14 (Data-complexity) *For any $0 < \epsilon \le 1/8$, $0 < \delta \le 1/683$, and any concept class C with $\mathrm{vc}(C) \ge 2$: Any learner that always observes an average training sample size less than*

$$t_{avg}(C,\epsilon,\delta) \;=\; \max\left\{ \frac{\mathrm{vc}(C)-1}{480\epsilon},\ \frac{1-2\delta}{2\epsilon} \right\}$$

for every fixed $c \in C$ and P_X will fail to meet the pac(ϵ,δ)-criterion for some target concept $c' \in C$ and domain distribution P'_X.

This shows that no new concept classes become pac-learnable with bounded data-efficiency merely by considering sequential over fixed-sample-size learners. That is, the target class C must still have finite VCdimension for there to be any sequential learner that can meet the pac-criterion for every target concept c in C and domain distribution P_X with a bound on expected training sample size. Thus, finite VCdimension is not only sufficient, but remains necessary for pac-learning with bounded data-efficiency. This also shows that `S` is a universal learning strategy in the same sense as `F`; *i.e.*, `S` correctly pac-learns any concept class C for which this is possible via any sequential learning procedure.

Interestingly, the lower bound t_{avg} behaves as

$$t_{avg}(C,\epsilon,\delta) \;=\; \Omega\left(\frac{\mathrm{vc}(C)-\delta}{\epsilon}\right),$$

and hence, scales the same as the fixed-sample-size lower bound t_{BEHKV} in terms of ϵ and $\mathrm{VC}(C)$; but with a slightly weaker dependence on δ. This shows that the expected data-efficiency of sequential learning necessarily scales the same as fixed-sample-size learning, in that the best possible improvement can only involve constant factors in $\mathrm{VC}(C)$ (and possibly logarithmic factors in $1/\epsilon$). Also, notice that the scaling behavior of t_{avg} matches the the worst case upper bound on S's expected sample size (2.2) up to constants and a small logarithmic factor. Thus, not only is S a universal pac-learner in the above sense, it also learns with near optimal scaling: no other learning procedure can improve S's data-efficiency by more than constant (and logarithmic) factors.

Overall, these results show how VCdimension continues to be a powerful measure of concept class complexity, as it still characterizes the inherent data-complexity of pac-learning, even when considering sequential over fixed-sample-size learners.

2.5 Empirical efficiency

As mentioned, we expect S to perform much better in practice than any bounds we can prove about its performance *a priori*, and hence, much better than either of the fixed-sample-size bounds T_{BEHW} or T_{STAB}. In fact, Procedure S proves to be many times more data-efficient than T_{BEHW} or T_{STAB} in empirical case studies; demonstrating a clear advantage for the sequential over the fixed-sample-size approach. This empirical advantage enjoyed by S is easily demonstrated by a simple example.

2.5.1 Problem

Consider the pac-learning problem $(X = \mathbb{R}^n, C = \text{halfspaces}, \epsilon = 0.01, \delta = 0.05)$ discussed in Section 2.3. The class of halfspace concepts on $\mathbb{R}^n$ is a canonical example that appears throughout machine learning, statistical pattern recognition [Duda and Hart, 1973], and neural networks research [Minsky and Papert, 1969; Hampson and Volper, 1986; Gallant, 1990], and thus should serve as a good example of a practical learning problem.[13] I tested Procedure S on this problem by fixing a target concept and domain distribution (chosen without much care), and seeing how many training examples S would observe while meeting the specified pac(ϵ, δ)-criterion. Specifically, the following setup was used: Training objects were generated by the uniform distribution on $[-1, 1]^n$ and labelled according to a fixed (diagonal) hyperplane that passed through the origin $\mathbf{0}^n$ with norm directed towards $\mathbf{1}^n$. S's constant κ was set to 3.1461932 (so that $\kappa = \kappa/(\kappa - 1 - \ln \kappa)$), and I supplied S with a hypothesizer H = HALFSPACE that finds consistent halfspace concepts for any linearly separable set of training examples.[14]

2.5.2 Results

A series of experiments was performed by setting $n = 10$ and running Procedure S 100 times—yielding the results shown in Table 2.2. Here we see that S observed an average number of about 3,402 training examples on this problem, which is about 5 times smaller than T_{STAB} and 27 times smaller than T_{BEHW}. Not only does S solve this pac-learning problem using far fewer training examples than demanded by the previous worst case bounds, it appears to be achieving near-practical data-efficiency: S's average training sample size was only about 3 times larger than T_{thumb}, the empirical rule of thumb for how many training examples should be needed to obtain ϵ error in practice. It is important to emphasize that S achieves these results while maintaining the exact same worst case pac-guarantees as before; namely, that an ϵ-accurate hypothesis is always returned with probability at least $1 - \delta$.

Not only were these surprising results obtained for the parameter settings $n = 10$, $\epsilon = 0.01$, and $\delta = 0.05$, they were also obtained over the entire range of parameter settings. For example, changing the dimensionality of the problem (n) does not reduce S's empirical advantage (Figure 2.5). In fact, it has the opposite effect: as n increases, S observes proportionally fewer training examples than F; *i.e.*, the ratio $T_{STAB}/\text{avg } T_{\mathsf{S}}$ increases

[13] The other advantage of this class is that there is also a well-developed body of computational techniques for dealing with this problem.

[14] Specifically, H = HALFSPACE was implemented by using a BFGS secant optimization procedure [Dennis and Schnabel, 1983] with a "relaxation" objective function [Duda and Hart, 1973].

For $(X = \mathbb{R}^{10}, C = \textsf{halfspaces}, \epsilon = 0.01, \delta = 0.05)$:

Sufficient:	T_{BEHW}	$=$	$91,030$
Improved:	T_{STAB}	$=$	$15,981$
Folklore:	T_{thumb}	$\approx$	$1,100$
Necessary:	t_{EHKV}	$=$	32

After 100 trials, Procedure S used:

avg T_{S}	**$=$**	**$3,402$**
$\max T_{\mathsf{S}}$	$=$	$5,155$
$\min T_{\mathsf{S}}$	$=$	$2,267$

Table 2.2: A direct comparison of training sample sizes for the pac-learning problem $(\mathbb{R}^{10}, \textsf{halfspaces}, \epsilon = 0.01, \delta = 0.05)$.

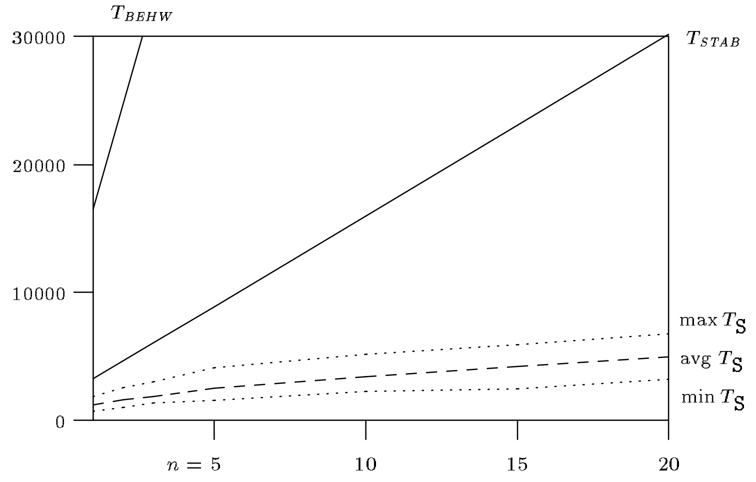

Figure 2.5: Scaling in input dimension n. Number of training examples observed for $(\mathbb{R}^n, \textsf{halfspaces}, \epsilon = 0.01, \delta = 0.05)$ with $n = 1, 2, 3, 5, 10, 15, 20$. Results of 100 runs at each parameter setting.

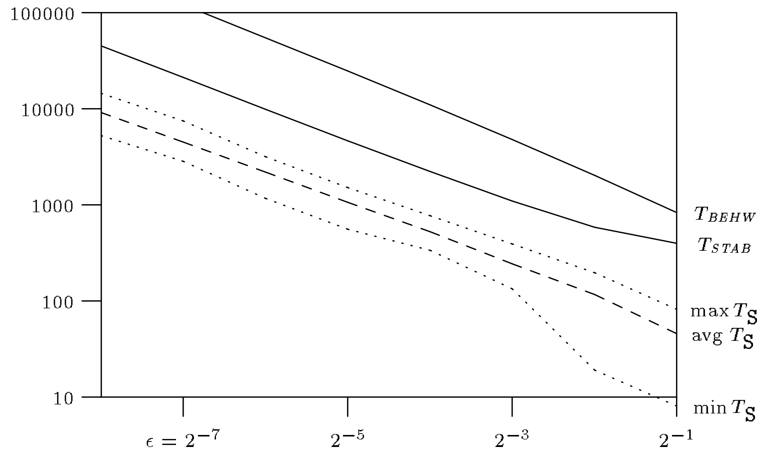

Figure 2.6: Scaling in error level ϵ. Number of training examples observed for $(\mathbb{R}^{10}, \textsf{halfspaces}, \epsilon, \delta = 0.05)$ with $\epsilon = 2^{-1}, 2^{-2}, ..., 2^{-8}$. Results of 100 runs at each parameter setting; log plot.

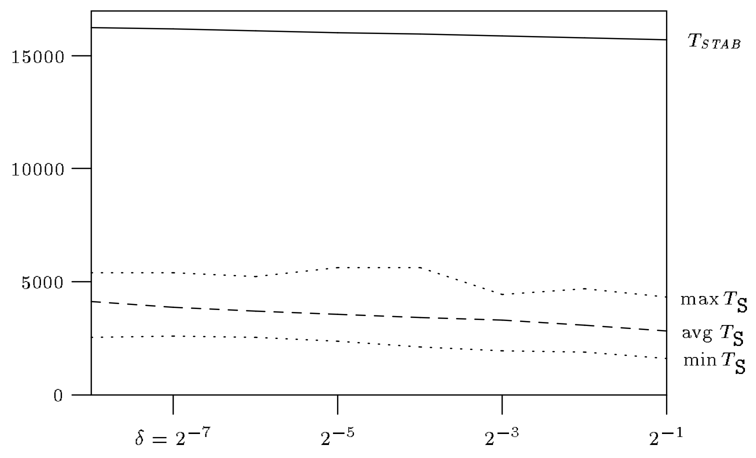

Figure 2.7: Scaling in failure level δ. Number of training examples observed for ($\mathbb{R}^{10}$, halfspaces, $\epsilon = 0.01$, δ) with $\delta = 2^{-1}, 2^{-2}, ..., 2^{-8}$. Results of 100 runs at each parameter setting.

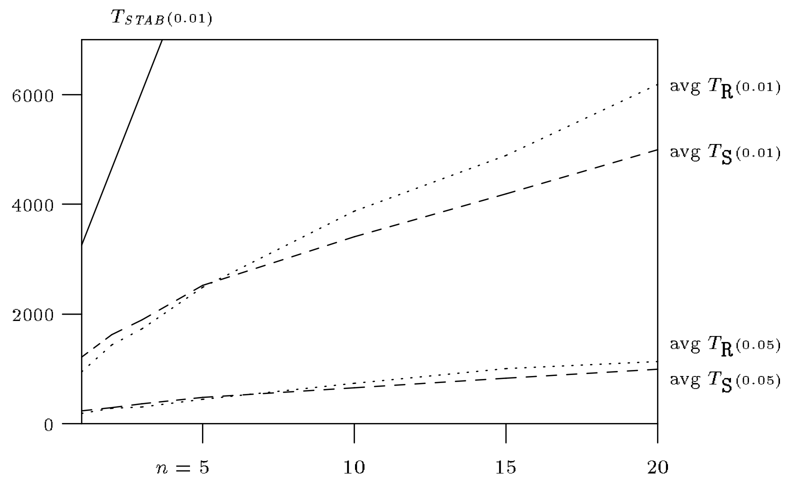

Figure 2.8: Comparing S versus R: Scaling in n. Number of training examples observed for ($\mathbb{R}^n$, halfspaces, ϵ, $\delta = 0.05$) with $n = 1, 2, 3, 5, 10, 15, 20$ and $\epsilon = 0.01, 0.05$. Results of 100 runs at each parameter setting.

for increasing n. This advantage is also maintained over the entire range of accuracy and reliability levels: Figures 2.6 and 2.7 demonstrate that the relative performance ratio $T_{STAB}/\text{avg } T_{\mathtt{S}}$ is unaffected by different choices of ϵ and δ. Overall, S appears to become relatively more efficient than F for harder parameter settings.

Interestingly, Procedure S also outperforms the simplistic sequential learning procedure R on this problem. Figure 2.8 shows that R performs nearly as well as S on problems with low dimension, accuracy, and reliability, but S's advantage grows significantly as these parameters are scaled up. In particular, S's advantage increases for larger problem dimension (Figure 2.8) and is maintained over the entire range of accuracy and reliability levels (not shown). So again, S's superiority increases for harder learning problems. Of course, the primary advantage of S over R is that we can obtain a reasonable upper bound on S's expected sample size, whereas this cannot be easily achieved for R. These results suggest that S might be inherently more data-efficient than R for harder learning problems.

2.5.3 Robustness

These empirical results demonstrate a clear advantage for the sequential learning procedure S over previous fixed-sample-size approaches, showing that S solves the exact same pac-learning problems using far fewer training examples. Of course, these results are anecdotal, and it is tempting dismiss the advantage as an artifact of the particular experimental setup. However, the previous results are extremely robust over changes of target concept, domain distribution, and even concept class. This suggests that S's empirical data-efficiency is in fact representative of what one might expect to achieve in practice, if not the worst case. To counter the claim that S's apparent success is merely anecdotal and has no bearing on its worst case behavior, I consider each of the ways that S's empirical performance might not be representative of its worst case performance in turn.

Easy target concept

One way S might accidentally demonstrate better than worst case performance is if the target concept happens to be a particularly easy one for S to learn. For example, if the hypothesizer H = HALFSPACE were somehow biased to guess hypotheses close to the target. However, this is easily demonstrated not to be the case here: Recall that the previous experiments considered a particular diagonal target halfspace c defined by a decision-hyperplane passing through the origin $\mathbf{0}^{10}$ with a norm directed towards $\mathbf{1}^{10}$. To test S's dependency on the target concept I repeated the previous experiment for a series of 10 different target halfspace-concepts, $c_{10}, c_9, ..., c_1$, each successively more axis-parallel than its predecessors. (In particular, c_i was defined by a halfspace that passed through the origin $\mathbf{0}^{10}$ with norm directed towards $\mathbf{1}^i\mathbf{0}^{10-i}$; meaning that c_i's classification of a domain object $x \in \mathbb{R}^{10}$ depended only on the first i attributes.) This experiment shows that changing the target concept has no effect on S's performance; see Figure 2.9.

Easy domain distribution

Another reason why the previous results might not be representative of S's true worst case performance is if the specific domain distribution, uniform$[-1,1]^n$, happens to make the learning problem particularly easy for S. To counter this claim, I tested S on a series of different domain distributions to determine whether any could have a serious effect on S's learning performance. Specifically, I considered three basic transformations of the uniform$[-1,1]^n$ distribution: *accretive*, *spherical*, and *pyramidal*.

The first intuition I explored was whether any discreteness in the domain distribution could make the learning problem any harder for S. The feeling I had was that it could not: A discrete distribution in effect stacks quanta of probability on top of one another, so that seeing any one member of a stack automatically gives the correct classification for all other members for free. It seems reasonable then that any distribution cannot be made harder by stacking these quanta. If true, this intuition means that we can concentrate our search for a hard distribution on continuous transformations of the uniform$[-1,1]^n$ distribution.

ACCRETIVE: To support this hypothesis, I considered an accretive transformation of the uniform$[-1,1]^n$ distribution: every domain object in $[-1,1]^n$ is moved a fraction τ of the distance towards its counterpart in $\{-1,1\}^n$, $0 \le \tau \le 1$, thus accreting probability towards the discrete points in $\{-1,1\}^n$. This

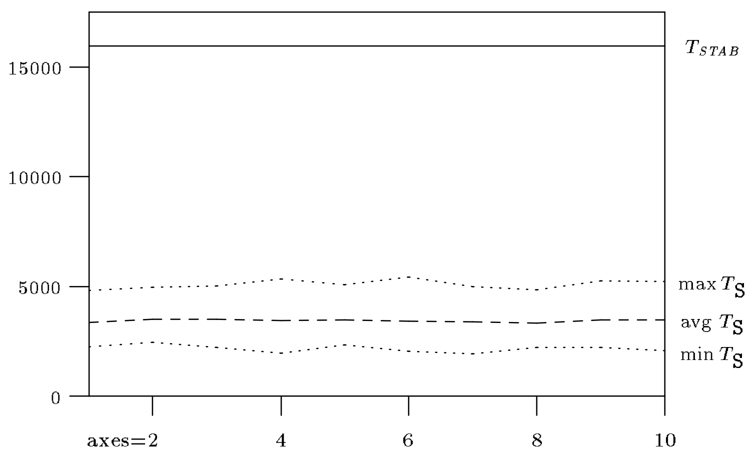

Figure 2.9: Comparing different target concepts. Number of training examples observed for ($\mathbb{R}^{10}$, halfspaces, $\epsilon = 0.01$, $\delta = 0.05$) with diagonal target concepts depending on $r = 1, 2, ..., 10$ relevant axes. Results of 100 runs at each parameter setting.

transformation interpolates a series of distributions between the original uniform$[-1,1]^n$ distribution (at $\tau = 0$) towards the fully discrete distribution uniform$\{-1,1\}^n$ (at $\tau = 1$).

Figure 2.10 shows that none of these transformations had any effect on S's data-efficiency. Therefore, we focus our attention on finding hard continuous distributions.

For a continuous distribution it is not clear what transformations of uniform$[1,-1]^n$ might make the learning problem harder for S. One intuition might be that clustering the domain distribution towards the target hyperplane should make learning harder, since this forces S to find a more precise representation of the target hyperplane. On the other hand, this also means that more training examples will be generated near the decision-hyperplane, and hence all concepts further away from the target will be eliminated sooner. To determine which, if any, of these factors has a serious effect on S's performance, I tested S on two different transformations of the uniform$[-1,1]^n$ distribution.

SPHERICAL: The first type of distribution I considered was spherical transformations of the uniform$[-1,1]^n$ distribution, where domain objects were translated directly towards or away from the origin $\mathbf{0}^n$. Specifically, I translated object descriptions $x \in \mathbb{R}^n$ so that their dot product with the normal to the target hyperplane, $\mathbf{1}^n$, was reduced or increased by some power factor of their original value. That is, the domain objects x were transformed $x \mapsto x'$ such that $x' \cdot \mathbf{1}^n = (x \cdot \mathbf{1}^n)^\alpha$ for some power factor α.[15] Thus, $\alpha = 1$ gives the original uniform$[1,-1]^n$ distribution, $\alpha > 1$ compresses the distribution towards the origin, and $\alpha < 1$ dilates the distribution away from the origin.

Figure 2.11 shows that none of these spherical transformations of the uniform$[-1,1]^{10}$ distribution had any significant effect on S's data-efficiency.

PYRAMIDAL: The other type of distribution I considered was pyramidal transformations of the uniform$[-1,1]^n$ distribution, where, instead of towards the origin, domain objects were transformed directly away from the opposite corners of the unit hypercube ($-\mathbf{1}^n$ and $\mathbf{1}^n$) towards the decision-hyperplane. That is, positive examples x were transformed along the vector $\mathbf{1}-x$ and negative examples x along $-\mathbf{1}-x$. The effect of this transformation is to spread domain objects uniformly about both sides of the decision-hyperplane, rather than cluster them around the origin. Again, object descriptions $x \in \mathbb{R}^n$ were transformed so that their dot product with the normal vector of the target hyperplane, $\mathbf{1}^n$, was reduced or increased to some power factor of their original value.

[15]Note that straightforward multiplicative scaling just yields an isomorphic learning problem.

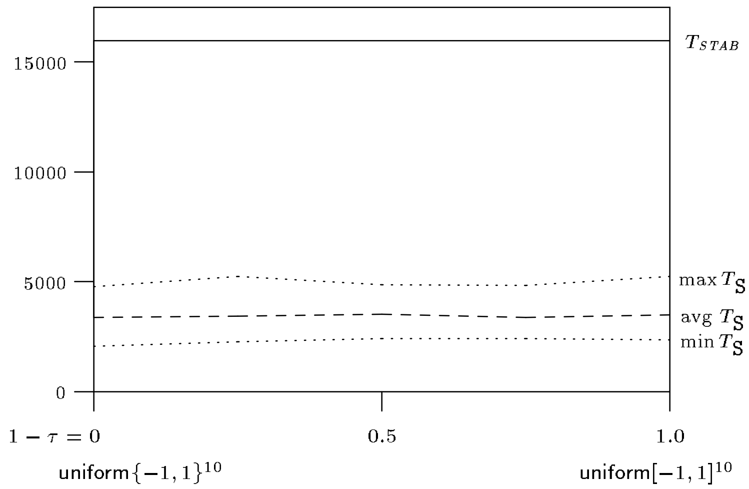

Figure 2.10: Comparing accretive transformations of the uniform$[-1,1]^{10}$ distribution. Number of training examples observed for $(\mathbb{R}^{10}, \textsf{halfspaces}, \epsilon = 0.01, \delta = 0.05)$ with accretive transformation of the uniform$[-1,1]^{10}$ distribution. *X-axis*: degree of domain distribution discreteness $(1-\tau)$. Results of 100 runs at each parameter setting.

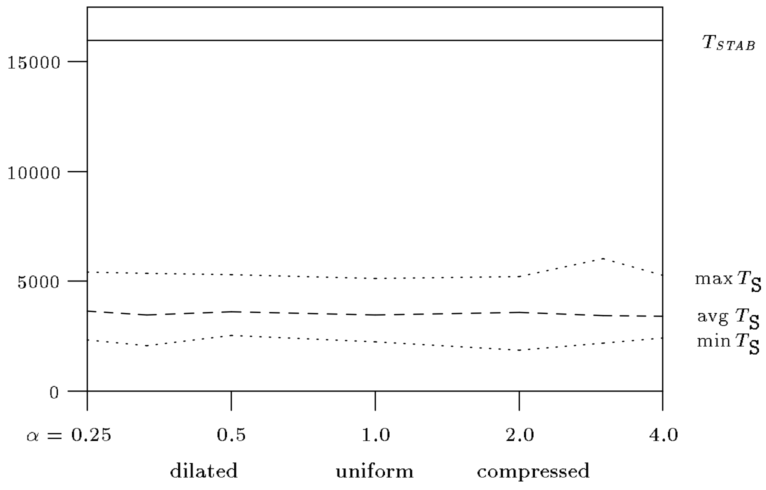

Figure 2.11: Comparing spherical transformations of the uniform$[-1,1]^{10}$ distribution. Number of training examples observed for $(\mathbb{R}^{10}, \textsf{halfspaces}, \epsilon = 0.01, \delta = 0.05)$. *X-axis*: power factor of transformed dot products. Results of 100 runs at each parameter setting.

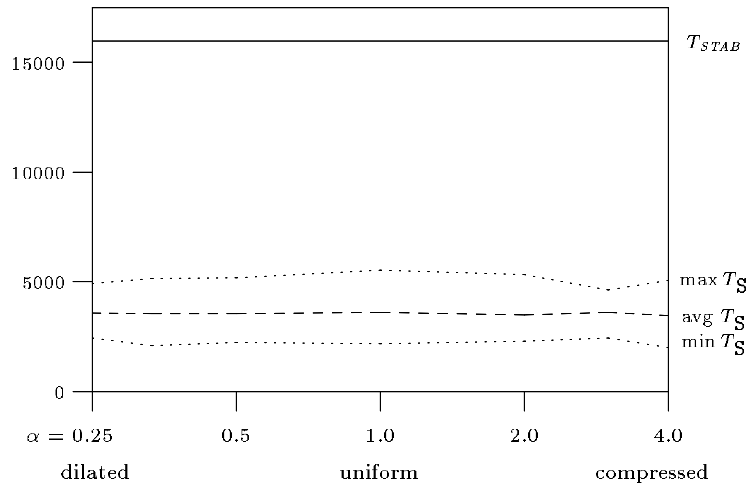

Figure 2.12: Comparing pyramidal transformations of the uniform$[-1,1]^{10}$ distribution. Number of training examples observed for ($\mathbb{R}^{10}$, halfspaces, $\epsilon = 0.01, \delta = 0.05$). *X-axis*: power factor of transformed dot products. Results of 100 runs at each parameter setting.

Figure 2.12 shows that S's empirical data-efficiency was also unaffected by pyramidal transformations of the uniform$[-1,1]^{10}$ distribution. Surprisingly, none of the wide variety of domain distributions considered had any significant effect on S's empirical data-efficiency.

Easy concept class

Another reason why S might demonstrate better than worst case performance is that the class of halfspace concepts might happen to be a particularly easy instance among concept classes with comparable VCdimension. That is, the class of halfspace concepts might have a simple structure that permits faster learning than for other classes with the same VCdimension. To counter this claim, I tested S on other concept classes with identical VCdimension. Here, it turns out that S's data-efficiency can be affected to some degree: I have been able to construct an alternative concept class, disj-π-chains, that forces S to observe nearly twice as many examples as for halfspaces; see Figure 2.13.[16] However, I have yet to devise any concept class that can double S's original data-efficiency on halfspaces for large d. In fact, S's performance often improves for other concept classes, particularly finite classes (Section 2.6). Overall, halfspaces does not appear to be a remarkably easy or hard class for a given VCdimension.

The robustness of these results suggests that S's original performance gives an accurate picture of its true efficiency at this task, if not its true worst case performance.

2.5.4 Explanations

These empirical results demonstrate a clear advantage for Procedure S over F: in every case S improves on the data-efficiency of the previous fixed-sample-size bounds T_{BEHW} and T_{STAB} by an order of magnitude. This raises the issue of explaining S's advantage over F. It appears that the primary source of S's advantage over F is that S's data-efficiency on any particular application is directly determined by the convergence properties of the specific problem at hand—not by what we can prove about the worst case. One consequence of this is that a sequential learner (like S) can automatically exploit easy learning situations to obtain a faster result [Oblow, 1992], whereas fixed-sample-size learning must always take the absolute worst case situation into account. Thus, it could be the case that S is simply exploiting an easy learning situation to improve on F.

[16]The class of disj-π-chains consists of $1/(d\epsilon)$ copies of a d-dimensional product of initial segment concepts, d-π-initials (defined in Section 3.4.4 of Chapter 3), where each product is defined on a mutually exclusive segment of $[0,1]$. This class has VCdimension d.

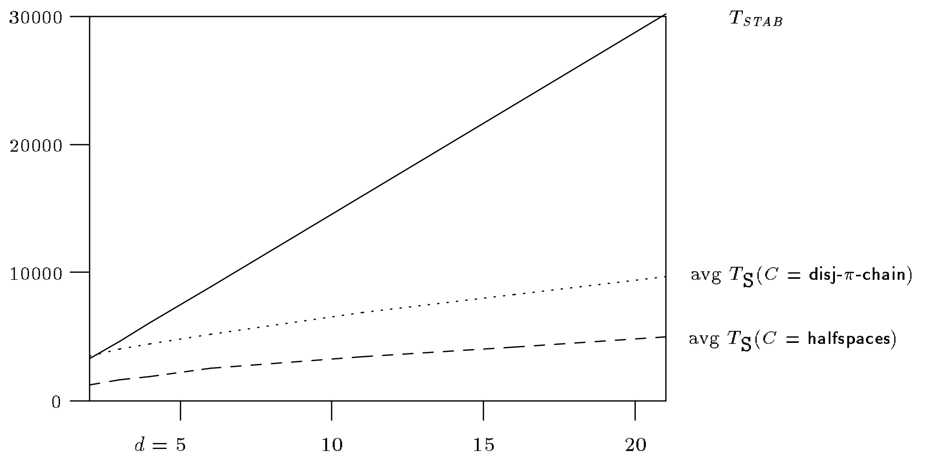

Figure 2.13: Comparing different concept classes with the same VCdimension. Average number of training examples observed by S for $(\mathbb{R}^n, C_i, \epsilon = 0.01, \delta = 0.05)$ given two different concept classes C_1 = **halfspaces** and C_2 = **disj-π-chains**, where $\mathrm{VC}(C) = 2, 3, 4, 6, 11, 16, 21$. Results of 100 runs at each parameter setting.

However, the robustness of the previous results suggest that this is not the primary source of S's advantage in our experiments.

Another explanation of S's advantage over F is that T_{STAB} might be a gross overestimate of the true worst case situation. In fact, this seems likely, given the gap between T_{STAB} and t_{EHKV}. This means that S might still demonstrate a significant advantage over T_{STAB} even in the worst case, since S's data-efficiency depends on the real convergence properties of this worst case situation, whereas T_{STAB} incorporates conservative assumptions.

A final explanation of S's advantage over F is that sequential learning might be inherently more efficient than fixed-sample-size learning. Since sequential learning generalizes the fixed-sample-size approach, clearly it can be no worse than Procedure F. The question is, how large an advantage can be obtained in principle? This is left largely unanswered by the previous empirical results, and remains an interesting open topic for future research.

2.5.5 Assessment

The sequential approach to pac-learning introduced in this section demonstrates many advantages over previous fixed-sample-size approaches. First of all, sequential learning decouples the correctness of a learning procedure from its efficiency. On the other hand, ensuring the correctness of fixed-sample-size learning requires one to prove that a particular sample size bound is sufficient, which directly determines the data-efficiency of the learning procedure. These two aspects are completely decoupled for a sequential learner. This means that we can develop correct sequential pac-learners that require far fewer training examples in practice than any bounds we can prove about their efficiency *a priori*. In fact, the previous results demonstrate not only that sequential learners observe far fewer training examples than our established bounds, but also that they use many times fewer training examples than the current fixed-sample-size bounds. Although the current fixed-sample-size bounds are loose (and any efficiency advantage currently enjoyed by S might possibly be overcome by future improvements to T_{STAB}) notice that we can enjoy S's improved performance right now, without having to wait for theoreticians to improve the bounds.

Another advantage of sequential learning is that, as mentioned above, the data-efficiency of a sequential learner depends directly on the specific case at hand, not the worst case situation. This means that sequential learning can automatically take advantage of beneficial situations like easy target concepts or domain distributions, or a good hypothesizer that makes lucky guesses—without the system designer having to realize that these beneficial situations exist *a priori*! More important, the worst case data-efficiency of a sequential learner depends directly on the true worst case properties of the concept class at hand, not on the

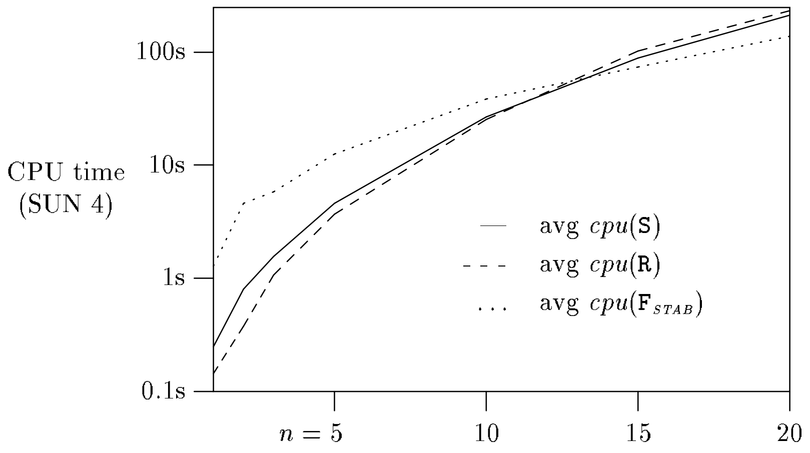

Figure 2.14: Comparing the computational overhead of S, R, and $\mathtt{F}_{STAB}$: scaling in input dimension n. Average CPU time used for ($\mathbb{R}^n$, halfspaces, $\epsilon = 0.01$, $\delta = 0.05$) with n = 1, 2, 3, 5, 10, 15, 20. Results of 100 runs at each parameter setting for a SUN4 implementation.

bounds we happen to be able to prove at the time. That is, S automatically stops sooner if the bad concepts are eliminated before the proven bounds. So, in effect, a sequential learner like S exploits the optimal worst case bounds right now, even though we are currently unable to prove exactly what these bounds are.

2.5.6 Computation

The main disadvantage of sequential over fixed-sample-size learning appears to be the additional computational overhead introduced by the sequential approach. Unlike Procedure F, which only has to call the hypothesizer H *once* during a run, a sequential learning procedure like S must call H repeatedly to produce consistent hypotheses. Moreover, Procedure S has the additional overhead of having to store these hypotheses and maintain their performance statistics. Surprisingly, however, this did not turn out to be a substantial expense in our experiments. Figure 2.14 shows that S is in fact more efficient than F for small problems, and obtains comparable efficiency on large problems. In any event, Theorem 2.11 shows that if F (and hence H) is guaranteed to terminate in polynomial time, then so is S (in polynomial expected time).

Interestingly, Figure 2.14 shows that S and R learn with about the same computational overhead for this problem. At first glance, S appears to be significantly less efficient than R, as it keeps a continually growing list of hypotheses and never discards even obviously bad candidates. It might seem intuitive that S would become bogged-down storing this list of hypotheses and maintaining their performance statistics. However, surprisingly, this does not turn out to be a significant factor in empirical case studies. It turns out that the task of *finding* consistent hypotheses (*i.e.*, calling H) takes most of the computational effort—*storing* these hypotheses once found (updating their statistics, *etc.*) does not require much overhead in comparison. In fact, for the example considered here it appears these two factors balance each other out, in that S and R obtain comparable computation times for this problem (Figure 2.14). If one wanted to improve S's computational efficiency, it seems the best approach would be to minimize the number of calls to H.

This concludes the primary discussion of distribution-free sequential pac-learning. The remainder of this chapter discusses a special case where stronger theoretical and empirical results can be obtained, and points out how sequential pac-learning is directly applicable to a much wider range of problems than fixed-sample-size learning.

2.6 Learning finite concept classes

The previous section showed how sequential learning can reduce the number of training examples needed to pac-learn in practice. However, the theoretical advantages that could be demonstrated were only slight, and only held for certain parameter settings. Here we consider the special case of *finite* concept classes, and obtain even stronger theoretical and empirical advantages. Specifically, we consider the problem of converting hypothesizers that have small mistake bounds into data-efficient pac-learners.

Mistake-bounded learning

Recall that for finite concept classes, the cardinality the class, $|C|$, provides an alternative measure of complexity (in addition to VCdimension) that determines sufficient sample sizes for pac-learning (via T_{finite}). Another, more interesting, measure of finite class complexity is given by the *mistake bound* of the class [Littlestone, 1988], which is defined as follows: Consider the task of observing a sequence of training examples on-line and attempting to classify each successive example in the sequence, given the correct classifications of all previous examples. Then the mistake bound of a finite class C is given by the smallest number of mistakes any hypothesizer can make in the worst case over all possible target concepts from C and domain object sequences from X^{∞}.

Definition 2.15 (Mistake bound) *For a finite class C and hypothesizer H, let $M(C,H)$ denote the maximum number of mistakes H makes on any example sequence consistent with some $c \in C$. Then C's* mistake bound*, denoted $M(C)$, is given by $M(C) = \inf_H M(C,H)$, the smallest mistake bound that can be achieved by any hypothesizer H.*

Notice that the mistake bound for a finite class C is at most $M(C) = |C| - 1$, since the strategy of guessing an arbitrary consistent concept h from C after each misclassified training example is guaranteed to make at most $|C| - 1$ mistakes before the target concept is identified. This bound holds for any target concept in C and any sequence of domain objects $\langle x_1, x_2, \ldots \rangle$. Littlestone observed that special guessing strategies can do even better than this: For example, guessing the "majority" concept at each stage (the concept that classifies each domain object x according to the majority vote among all consistent concepts h remaining in C) makes at most $\log_2 |C|$ mistakes in the worst case. In fact, one can do even better than this in special cases [Littlestone, 1988]. However, it can be shown that $\mathrm{VC}(C)$ always gives a lower bound on $M(C)$.

Mistake-bounded to pac conversion

In later work Littlestone [1989] considered the problem of converting hypothesizers with small mistake bounds into data-efficient pac-learning procedures. Here Littlestone developed a simple two-stage (fixed-sample-size) conversion procedure, `Li`, that achieves the best known data-efficiency for this problem; see Figure 2.15. The basic idea behind his procedure is to first observe a number of training examples that is sufficient to guarantee that an $\epsilon/2$-accurate hypothesis is obtained from H with probability at least $1-\delta/2$, and then estimate the errors of these hypotheses to within $\epsilon/2$ (with probability at least $1-\delta/2$). Choosing the hypothesis with the best estimate yields a correct pac-learning procedure, since with probability at least $1-\delta$ we return a hypothesis with error at most ϵ [Littlestone, 1989].

For $M = M(C,H)$, the data-efficiency of `Li` is given by the sum of the two training samples it observes, which Littlestone bounds by

$$T_{\mathtt{Li}}(M,\epsilon,\delta) \;=\; \frac{4}{\epsilon}\left(M + 8\ln(M+2) + 12\ln\frac{2}{\delta} - \frac{1}{2}\right).$$

Thus, the training sample size of this procedure scales as

$$T_{\mathtt{Li}}(M,\epsilon,\delta) \;=\; \Theta\left(\frac{1}{\epsilon}\left(M + \ln\frac{1}{\delta}\right)\right)$$

in terms of ϵ, δ and the mistake bound M.

Procedure `Li` $(C, \epsilon, \delta;\ H)$

- Let $M = MB(C, H)$.
- Observe $\max\left\{\frac{16}{\epsilon}\ln\frac{2}{\delta},\ \frac{4(M-1)}{\epsilon}\right\}$ training examples labelled by some unknown target concept $c \in C$.
  - Apply the mistake bounded hypothesizer H sequentially to these examples.
  - If the most recent hypothesis h_i misclassifies a training example, call H to obtain a new consistent hypothesis h_{i+1}, and add h_{i+1} to the end of the list.
  - Collect H's hypotheses in a list $\{h_0, h_1, ..., h_i\}$, $i \le M$.
- Collect an additional $\frac{32}{\epsilon}\ln\frac{2(M+2)}{\delta}$ examples, and return the hypothesis $h \in \{h_0, h_1, ..., h_i\}$ that obtains minimum observed error on these additional examples.

Figure 2.15: Procedure `Li`

A sequential conversion procedure

Here, we consider whether a sequential conversion procedure might do better than `Li`. First, notice that Procedure `S` can be applied to this conversion problem as is. However, we can obtain slightly better performance by modifying `S` to return a hypothesis as soon as the mistake bound is reached, setting $\kappa = 3.14619$, and testing each hypothesis h_i with a failure level $\delta_i = \delta/M$. This gives a conversion procedure `Smb` (Figure 2.16) that is provably more data-efficient than `Li`. (Note that the correctness of `Smb` is easy to establish given the correctness of `S`.) We derive the following reasonable upper bound on the worst case expected number of training examples `Smb` observes to solve this conversion problem.

Theorem 2.16 *For any $\epsilon > 0$, $\delta > 0$, and any finite concept class C: Given i.i.d. examples generated by any distribution P_X and target concept $c \in C$, Procedure `Smb` observes an average training sample size of at most*

$$\mathrm{E}\,T_{\mathtt{Smb}}(M, \epsilon, \delta) \ \le\ \frac{\kappa M}{\epsilon} + \left(\frac{\kappa}{\kappa - 1 - \ln\kappa}\right)\frac{1}{\epsilon}\left(\ln\frac{M}{\delta} + 1\right);$$

using a hypothesizer H with $M = M(C, H)$ and a constant $\kappa > 1$. Choosing $\kappa = 3.14619$ gives

$$\mathrm{E}\,T_{\mathtt{Smb}}(M, \epsilon, \delta) \ \le\ \frac{3.14619}{\epsilon}\left(M + \ln M + \ln\frac{1}{\delta} + 1\right). \tag{2.3}$$

Note that this bound on $\mathrm{E}\,T_{\mathtt{Smb}}$ is strictly smaller than `Li`'s fixed sample size $T_{\mathtt{Li}}$ for all values of ϵ, δ, and M. Again, although this theoretical advantage is not overwhelming, we expect the sequential procedure `Smb` to do much better in practice than the crude upper bound (2.3) would indicate. This is unlike the fixed-sample-size conversion procedure `Li` which will always perform exactly as specified by $T_{\mathtt{Li}}$.

Empirical comparison

In fact, the number of training examples `Smb` observes in practice is orders of magnitude smaller than both (2.3) and $T_{\mathtt{Li}}$. To demonstrate this, consider the following case study.

Problem: Consider the pac-learning problem $(X = \{0,1\}^n, C = \mathsf{halfspaces}, \epsilon, \delta = 0.05)$ that involves learning an unknown halfspace concept defined on $\{0,1\}^n$. I tested `Smb` on this problem by setting a failure level of $\delta = 0.05$ and considering various error levels ϵ. In particular, I used the following setup: Training objects were generated by independently setting each bit to 1 with probability $1/4$, and labelled according to a simple disjunction of the first k attributes, which happens to be a linearly separable concept. `Smb` was supplied with a hypothesizer $H =$ WINNOW [Littlestone, 1988] that happens to have a good mistake bound for this problem.

Procedure `Smb` $(C, \epsilon, \delta;\ H, \kappa)$

- Let $M = MB(C, H)$.
- Proceed exactly as Procedure `S` (Figure 2.3) except:
  - Test each hypothesis h_i with reliability $\delta_i = \delta / M$.
  - If the hypothesis h_M is ever reached, automatically return h_M since this is guaranteed to be the target concept by construction.

Figure 2.16: Procedure `Smb`

Results: Figure 2.17 shows the results obtained for 200 runs of `Smb` at $\epsilon = 2^{-1}, 2^{-2}, ..., 2^{-8}$ with settings $n = 30$ and $k = 10$. This graph compares `Smb`'s data-efficiency to the bound (2.3) and $T_{\texttt{Li}}$, using WINNOW's mistake bound. Notice that `Smb` actually used about 15 times fewer training examples than the upper bound (2.3), and 30 times fewer examples than `Li`—even considering the maximum number of examples `Smb` used in any of the 200 runs! This is a significant improvement. In fact, $T_{\texttt{Smb}}$ actually appears to scale better than (2.3) and $T_{\texttt{Li}}$ for smaller values of ϵ; obtaining a noticeably improved performance ratio as ϵ becomes small.

2.7 Range of applicability

The previous sections observed how sequential learning can solve pac-learning problems using fewer training examples than existing fixed-sample-size techniques. Here we observe that sequential learning is applicable to a much wider range of pac-learning problems as well. Specifically, we note that a sequential learning procedure like `S` can learn with arbitrary hypothesizers, even if the hypothesizer produces concepts from a class with *infinite* VCdimension. The point is that if the hypothesizer ever manages to produce a reasonable hypothesis, `S` automatically detects this and returns the acceptable candidate. Notice that a fixed-sample-size approach is inapplicable to any situation where the concept class has infinite VCdimension, since Theorem 2.7 automatically rules out any finite sample size as being sufficient. However, `S` can be applied as is to pac-learn (many) concept classes that have infinite VCdimension—the only catch is that we can no longer place a uniform prior bound on `S`'s expected training sample size. Thus, we are not really contradicting Theorem 2.14 here, as `S` can have finite expected sample size for each target concept and domain distribution, without there being a uniform upper bound over all of them (*cf.* Section 2.7.2 below).

Below we illustrate how `S` can be applied to arbitrary hypothesizers H and then consider the general problem of pac-learning concept classes of infinite VCdimension with finite data-efficiency.

2.7.1 General hypothesizers

Many popular hypothesis generation procedures implicitly consider concept classes that have infinite VCdimension. Two of the most common examples of this are the NEAREST-NEIGHBOR hypothesizer [Duda and Hart, 1973] and DECISION-TREE hypothesizer (*CART*) [Breiman et al., 1984], both of which can be applied to learning concepts defined on $\mathbb{R}^n$. In each case, the concept classes implicitly defined by these hypothesizers can shatter arbitrarily large sets, and hence have infinite VCdimension by Definition 2.3. Although this automatically rules-out any fixed-sample-size procedure for achieving pac-learning, Procedure `S` can be directly applied with these hypothesizers to yield successful pac-learning.

Demonstration

To demonstrate this, I tested Procedure `S` on the same learning problem $(X = \mathbb{R}^n, C = \textsf{halfspaces}, \epsilon, \delta)$ considered before, but here supplying `S` with a decision-tree and a nearest-neighbor hypothesizer, respectively. The idea is to show that `S` can still obtain successful pac-learning in typical applications, even though the hypothesis classes have infinite VCdimension. In particular, I tested `S` under the same setup as before:

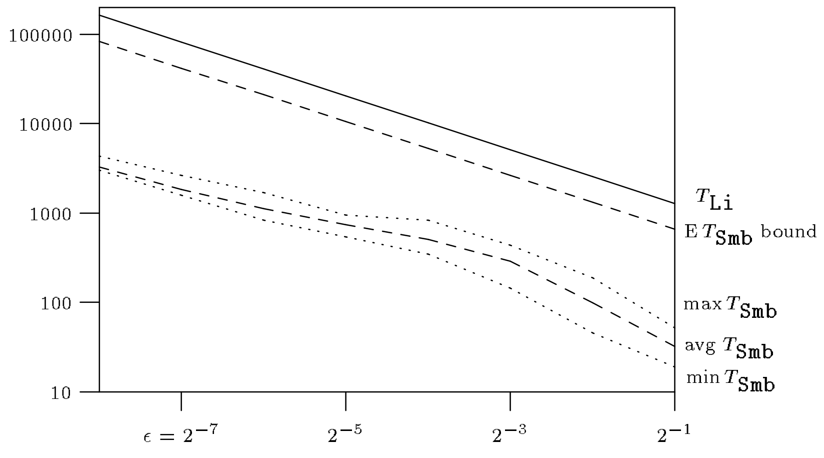

Figure 2.17: Scaling in error level ϵ. Number of training examples observed for $(X = \{0,1\}^{30}$, halfspaces, ϵ, $\delta = 0.05)$ with $\epsilon = 2^{-1}, 2^{-2}, ..., 2^{-8}$. Results of 200 runs for each parameter setting; log plot.

Training objects were generated by the uniform$[-1,1]^n$ distribution and labelled according to the same halfspace target concept considered in Section 2.5; namely, the diagonal hyperplane passing through the origin $\mathbf{0}^n$ with norm directed towards $\mathbf{1}^n$. The constant κ was set to 3.14619 and S was supplied with either $H =$ DECISION-TREE or $H =$ NEAREST-NEIGHBOR, respectively.

Decision-tree hypothesizer: The specific decision-tree hypothesizer I considered was the *CART* procedure [Breiman et al., 1984], which returns decision-tree hypotheses that make only axis-parallel cuts of $\mathbb{R}^n$. That is, each decision node only tests the value of a single object attribute x_i against some cutoff value v. This concept class clearly has infinite VCdimension, since the family of axis-parallel decision-trees can independently label any finite collection of objects in $\mathbb{R}^n$. I used an implementation of a decision-tree hypothesizer that builds small decision-trees consistent with the training examples by using a standard information-gain heuristic to choose the cuts [Breiman et al., 1984; Quinlan, 1986].

Setting $n = 1,2,3$ and running Procedure S 100 times on each problem $(X = \mathbb{R}^n, C =$ halfspaces$, \epsilon = 0.1, \delta = 0.1)$ yielded the results shown in Figure 2.18. Notice that despite having infinite VCdimension, S was easily able to meet the pac-criterion (*i.e.*, returning a decision-tree hypothesis with error at most ϵ with probability at least $1-\delta$) within a reasonable number of training examples. Of course, even these preliminary results indicate that decision-tree learning does not scale-up well: the data-efficiency is apparently exponential in the input dimension n. However, the problem here is not dimensionality *per se*, but the fact that it takes exponentially many axis-parallel cuts to approximate a diagonal hyperplane. We would expect the data-efficiency not to degrade so rapidly if the target concept were more axis-parallel. In effect, the diagonal target concept considered here contradicts the "prior knowledge" that the target concept can be approximated by a small decision-tree.

Nearest-neighbor hypothesizer: I also investigated the standard nearest-neighbor hypothesizer which simply memorizes the training examples $\langle\langle x_1, c(x_1)\rangle, ..., \langle x_t, c(x_t)\rangle\rangle$ and returns a hypothesis that classifies any subsequent test example $\langle x, c(x)\rangle$ according to the nearest training object x_i in the list (under the standard Euclidean metric). Clearly, the set of classification rules definable by nearest-neighbor classifiers has infinite VCdimension.

Setting $n = 1, ..., 4$ and running Procedure S 100 times on each problem $(X = \mathbb{R}^n, C =$ halfspaces$, \epsilon = 0.1, \delta = 0.1)$ yielded the results shown in Figure 2.18. Again, in spite of dealing with a hypothesis class that has infinite VCdimension, S has no problem returning hypotheses that meet the pac-criterion after a reasonable number of training examples. However, these results show that nearest-neighbor learning does not scale-up well in the dimensionality of the problem. This reflects the well known "curse of dimensionality" suffered by nearest-neighbor learning [Duda and Hart, 1973]: increasing the dimensionality of the problem leads to an exponential increase in the number of training examples needed to adequately cover both sides of the separating hyperplane.

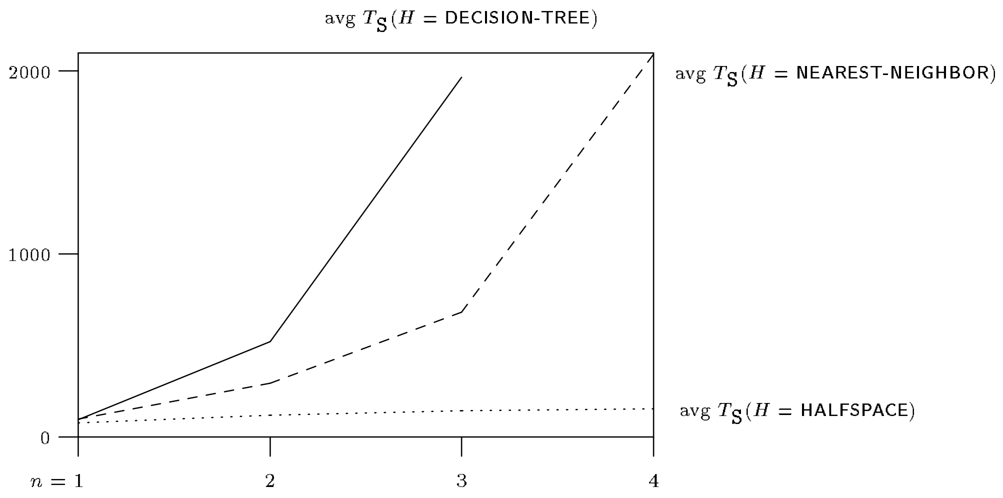

Figure 2.18: Comparing different hypothesizers. Number of training examples observed by S for $(\mathbb{R}^n, \text{halfspaces}, \epsilon = 0.1, \delta = 0.1)$ given a "diagonal" halfspace target concept, using the hypothesizers: H = DECISION-TREE, H = NEAREST-NEIGHBOR, and H = HALFSPACE. Results for $n = 1, 2, 3, 4$; 100 runs at each parameter setting.

Conclusion

Of course, these experimental results are once again anecdotal, but they do seem representative of the type of learning performance one might expect to observe in practice. These results show that the sequential learning techniques developed in this chapter have a far greater range of applicability than just considering hypothesizers that choose concepts from classes with finite VCdimension. In effect, S is a generic test procedure that identifies accurate candidates from arbitrary hypothesis sequences. That is, one can supply S with *any* hypothesizer deemed appropriate for the task at hand, thus accommodating arbitrary forms of prior knowledge and biases that go beyond the simple concept class restrictions normally considered in pac-learning theory. The point is that, whatever the hypothesis guessing strategy, if it somehow encodes appropriate knowledge about the domain, or even just gets lucky, S will quickly discover this and return an accurate hypothesis. On the other hand, if the guessing strategy is ill-suited for the task at hand, then S may take a long time to terminate, if ever. Regardless of the time taken however, the probability that S returns an ϵ-bad hypothesis is guaranteed to be less than δ, for any hypothesizer H.

These results suggest that it is possible to pac-learn concept classes with infinite VCdimension so long as we are prepared to give up a uniform bound on data-efficiency. This raises the question of characterizing the range of concept classes C that can be pac-learned under this weaker criterion.

2.7.2 Non-uniform pac-learning

In this section we consider the problem of "non-uniform" (in C) pac-learning. That is, we consider the task of pac-learning a concept class C with finite data-efficiency for each c in C, without demanding that there be a uniform bound on data-efficiency for all c in C.[17] This question has been previously addressed in the pac-learning theory literature, where many authors have considered the possibility of pac-learning concept classes that have infinite VCdimension [Benedek and Itai, 1988b; Ben-David, Benedek and Mansour, 1989; Linial, Mansour and Rivest, 1991]. Clearly, no procedure can pac-learn such a class with a bounded training sample size (Theorems 2.7 and 2.14), so the idea is instead to demand only a finite (but different) bound

[17] Notice that this is a different question than addressed by Theorems 2.7 and 2.14. There it was shown that finite VCdimension is necessary to pac-learn a concept class C with a uniform bound on data-efficiency over all c in C. Here we are only demanding that the learner produce a hypothesis with bounded data-efficiency for each c in C, without there having to be a uniform bound for all c in C.

Procedure `LMR` (C, ϵ, δ)

- Fix an arbitrary sequence $\{\delta_i\}_{i=1}^{\infty}$ such that $\delta_i > 0$ and $\sum_{i=1}^{\infty} \delta_i = \delta$.
- Decompose C into $C_1 \subset C_2 \subset ...$ such that $\bigcup_{i=1}^{\infty} C_i = C$ and $\text{VC}(C_i) < \infty$ for each C_i.
- For each concept class $C_1, C_2, ..., C_i, ...,$ *etc.* in sequence:
  - Add training examples to a global list, until a total of $T_{BEHW}(C_i, \epsilon, \delta_i)$ have been observed.
  - If there is a concept $c \in C_i$ that correctly classifies every training example, halt and return c.
  - If no such c exists in C_i, go on to the next concept class and repeat the loop.
- Repeat until a consistent concept is found in some class C_j.

Figure 2.19: Procedure `LMR`

for each c in C individually. It turns out that many concept classes with infinite VCdimension become pac-learnable under this weaker criterion.

Linial, Mansour and Rivest [1991] have developed an obvious approach to non-uniform pac-learning in these cases: First, decompose C into a sequence of concept classes $\{C_i\}_{i=1}^{\infty}$ such that $\bigcup_{i=1}^{\infty} C_i = C$ and yet each C_i has $\text{VC}(C_i) < \infty$. Then, simply apply Procedure `F` to each C_i in parallel; see Procedure `LMR` in Figure 2.19. The only trick is to use successively smaller failure levels δ_i so that the overall probability of producing an ϵ-bad hypothesis is bounded by δ. In this way, `LMR` only requires a finite number of training examples for each target concept c in C, but the precise bound depends on the smallest class that contains the target concept.

Clearly, Procedure `LMR` correctly pac-learns any concept class C that is decomposable in this way. For example, the class of decision-tree classifiers on $\mathbb{R}^n$ can be decomposed into a series of classes $\{C_i\}_{i=1}^{\infty}$ where each class consists of the concepts that can be defined by an i-node decision-tree. However, not every concept class can be decomposed in this way: For example, the class $\mathcal{B}$ of Borel concepts defined on $\mathbb{R}$ can shatter infinite sets, and hence cannot be decomposed into a series of classes with finite VCdimension [Benedek and Itai, 1988b, Lemma 1]. It turns out that this notion of decomposability is not only sufficient, but also necessary for non-uniform (in C) pac-learnability.

Theorem 2.17 [Benedek and Itai, 1988b] *The following are equivalent:*

1. *C can be pac-learned with a bounded sample size for each individual c in C.*
2. *C can be decomposed as $C = \bigcup_{i=1}^{\infty} C_i$ where $\text{VC}(C_i) < \infty$.*
3. *Procedure `LMR` pac-learns C.*

This result shows that the concept class $\mathcal{B}$ on $\mathbb{R}$ cannot be pac-learned, even in this weak sense. It also shows that Procedure `LMR` is universal for non-uniform (in C) pac-learning, in that C can be non-uniformly pac-learned if and only if `LMR` pac-learns it.

Sequential non-uniform pac-learning

It is important not to confuse the issue of sequential with non-uniform pac-learning. Even though `LMR` can be interpreted as a sequential learning procedure, since the number of training examples it observes depends on the specific training sequence it receives, it is important to realize that Procedure `LMR` has a fundamentally different motivation from Procedure `S`: `LMR` seeks to increase the *range* of pac-learnable concept classes C by sacrificing data-efficiency on those target concepts late in the decomposition $\bigcup_{i=1}^{\infty} C_i$ in favor of those that appear earlier. Procedure `S`, on the other hand, seeks to obtain a uniform improvement in data-efficiency over all target concepts c in C. In fact, these two concerns are orthogonal: One can adopt a sequential approach to non-uniform pac-learning, and use a sequential procedure like `S` to pac-learn each subclass C_i, in this way obtaining improved performance for each in addition to the non-uniform advantages.

Interestingly, `S` can directly serve as a non-uniform pac-learning procedure as is. The only trick is to supply `S` with an appropriate hypothesizer H that guesses consistent concepts from the smallest possible class in a decomposition $\{C_i\}_{i=1}^{\infty}$ where $\text{VC}(C_i) < \infty$. In this way we obtain the same range of applicability as `LMR` but also benefit from `S`'s improved data-efficiency. For the problem of pac-learning a concept class C with finite *expected* data-efficiency for each c in C, we note that the previous form of decomposability remains both necessary and sufficient for non-uniform (in C) pac-learning, as before.

Theorem 2.18 *The following are equivalent:*

1. *C can be pac-learned with a bounded* expected *sample size for each individual c in C.*
2. *C can be decomposed as $C = \bigcup_{i=1}^{\infty} C_i$ where $\text{VC}(C_i) < \infty$.*
3. *Procedure `S` pac-learns C with a hypothesizer H that produces consistent concepts from the earliest possible class in the decomposition.*

This shows that `S`, using an appropriate hypothesizer H, is also a universal non-uniform pac-learner: Not only is Procedure `S` able to pac-learn any concept class C that has finite VCdimension (by Theorem 2.11), it is also able to non-uniformly pac-learn any decomposable concept class C given an appropriate hypothesizer H.

2.8 Conclusion

This chapter raised the idea of sequential pac-learning (*i.e.*, observing training examples on-line and deciding when an accurate hypothesis can be reliably returned) in an attempt to reduce the overall (average) number of training examples needed to pac-learn.

We introduced a novel sequential learning procedure, `S`, that proved to be far more data-efficient in practice than existing fixed-sample-size learning techniques. Theoretical analysis of this procedure shows that its expected training sample size provably beats the best known fixed-sample-size bounds in some cases, but overall, scales the same as these previous bounds in terms of the desired accuracy and reliability levels, ϵ and δ, and the VCdimension of the concept class, $\text{VC}(C)$. Although the demonstrated theoretical advantage is slight, a series of experimental studies show that `S` uses many times fewer training examples in practice than previous fixed-sample-size approaches. In addition to these results, we also obtained a lower bound on the average training sample size needed to pac-learn. This showed that `S`'s expected data-efficiency is within constant and logarithmic factors of the best performance possible and also shows that the inherent data-complexity of sequential pac-learning scales the same as fixed-sample-size learning. Therefore, sequential learning can at best offer a constant improvement in data-efficiency over fixed-sample-size learning. However, we saw that these constant factors can have a significant effect in practical applications.

After investigating the general case, we then considered the special case of finite concept classes. In particular, we used a variant of Procedure `S` to convert hypothesizers with small mistake-bounds into data-efficient pac-learners. A theoretical analysis showed that this sequential conversion procedure requires fewer training examples than the best known fixed-sample-size procedure due to Littlestone [1989]. In fact, this procedure required orders of magnitude fewer training examples in empirical case studies. Finally, we showed that Procedure `S` successfully pac-learns in many situations where fixed-sample-size learning is impossible. Specifically, it was shown that `S` can pac-learn many concept classes that have infinite VCdimension (a task that is impossible for any fixed-sample-size learner).

Overall, these results demonstrate a clear advantage for sequential over fixed-sample-size learning. The main reason for this advantage is that the sequential approach decouples the correctness of a learner from its efficiency. This permits us to design correct sequential learning procedures independently of our ability to prove good bounds on their worst case data-efficiency. The primary benefit of this is that the data-efficiency of a sequential procedure is often far better than what we can prove. This is unlike the fixed-sample-size approach, where the obtainable data-efficiency is directly determined by the smallest sample size bounds that can be proved sufficient. This advantage is revealed through numerous empirical case studies that showed how sequential learning requires many times fewer training examples to achieve the exact same pac-criterion as existing fixed-sample-size approaches.

2.8.1 Research directions

A number of research directions are suggested by this work. The three most important areas of future investigation are: obtaining further improvements in data-efficiency, improving computational-efficiency, and coping with classification noise.

Data-efficiency: Even though Procedure S significantly improves the practical data-efficiency of existing fixed-sample-size learners, it seems clear that further improvements are possible. For instance, most of the empirical results reported in this chapter appear to be improved by a factor of about 10%, simply by running S with a value of $\kappa = 6$ instead of $\kappa = 3.1461932$. Perhaps a deeper analysis of the learning problem could yield additional improvements, both in practice, and in terms of the worst case bounds we can prove. If not in general, then it would at least be interesting to obtain improvements for important special cases like finite concept classes, or halfspace concepts on $\mathbb{R}^n$.

Although the worst case bounds established here are not strong enough to show a significant advantage for sequential over fixed-sample-size learning, the empirical results strongly suggest that such an advantage exists. Therefore, it remains a priority to obtain better bounds on S's worst case expected data-efficiency.

There also remains the question of whether sequential learning is inherently more data-efficient than fixed-sample-size learning; *i.e.*, whether the current advantage enjoyed by Procedure S over F is simply due to the weakness of the existing sufficient sample size bounds T_{BEHW} and T_{STAB}. Even though T_{STAB} is the best known bound, it is by no means guaranteed to be the best possible—especially since T_{STAB} and t_{EHKV} differ by a factor of about 64. Perhaps the fixed-sample-size bounds could be improved to the point where they compete with S's empirical data-efficiency. Proving that no fixed-sample-size bound can ever be as efficient as the best sequential strategy remains an interesting open challenge.

Computational-efficiency: This research was initially motivated by the observation that, in practice, it is usually more important to conserve training data than to save computation time. Thus, Procedure S was considered superior to F because it observes fewer training examples in practice. However, even given that the goal is to trade-off computational-efficiency for data-efficiency, it is still important to consider ways in which S's computational-efficiency might be improved.

Recall that the major computational expense in implementing S is not storing the sequence of hypotheses generated by H, but rather finding consistent hypotheses that correctly classified every training example (*i.e.*, calling H). Therefore, the key issue to improving S's computational-efficiency is to reduce the number of calls to H, or somehow improve the efficiency of each call. An interesting idea here is to exploit the fact that S repeatedly calls H to solve a nested series of consistency problems. That is, instead of starting H from scratch every time it is called, we should be able to exploit H's previous hypotheses to obtain a faster solution to the current consistency problem, and thereby reduce S's overall computation time. Research on incremental learning algorithms appears to have some bearing on this issue.

Classification noise: Perhaps the most important direction for future research is to extend these sequential techniques to cope with the presence of classification noise. This is the main barrier between the results presented here and real applications. In practice, object classifications are almost always noisy, and even if not, it is rarely happens that we have a prior class that we can guarantee contains a perfect hypothesis. In most applications it is impossible to achieve 100% classification accuracy, and therefore we need to strive for near-optimal rather than near-perfect classification accuracy. This raises the problem of "probably approximately class optimal" (paco), as opposed to pac-learning.

It turns out that paco-learning is not hard to achieve in the case where the class-optimal error rate, β, is known beforehand: simply modify Procedure S to test whether a hypothesis has error within ϵ/κ of β rather than within ϵ/κ of 0. As before, if S halts it is guaranteed to meet the paco-criterion. This problem becomes difficult, however, if the optimal error rate β is not known *a priori*. In fact, it is not even clear how to proceed in this case. The problem is that an appropriate stopping rule can no longer be based on the absolute error of a single hypothesis. Instead, we must consider how any such error compares to the unknown class-optimal error β, which is a property of the entire class C.

One plausible approach is to test H's hypotheses to see if they are really improving, and, if not, terminate. However, care must be taken to ensure that H is actually trying to produce good hypotheses. For example, if H holds out before producing reasonable hypotheses, it can force premature convergence to bad hypotheses. At the very least, we require a hypothesizer that honestly attempts to produce concepts with near–class-optimal empirical error. Devising an effective sequential learning procedure for this problem appears to be a difficult challenge.

2.8.2 Is pac-learning practical?

Recall that Section 2.3 motivated this research by pointing out how the fixed-sample-size approach to pac-learning seems to be impractical in terms of the training sample sizes it demands. The common belief is that these large training sample sizes are forced by the requirement that the pac-criterion be achieved in the worst case over all possible domain distributions. However, the results presented in this chapter directly counter this claim by showing that distribution-free pac-learning can be far more efficiently achieved in practice than previously thought. In fact, the empirical results suggest that distribution-free pac-learning might be achieved with practical data-efficiency in real applications. Although it remains equivocal whether distribution-free pac-learning is truly practical in the worst possible case, these results clearly show that it is premature to conclude that distribution-free pac-learning is inherently impractical.

Nevertheless, the theoretical worst case bounds are still weak, and it could perhaps be that there really are pathological domain distributions that force large training sample sizes; *i.e.*, the current lower bounds could be much weaker than the upper bounds (although this seems unlikely).[18] This belief motivates much research that makes distributional assumptions to improve the data-efficiency of pac-learning, *e.g.*, [Baum, 1990; Benedek and Itai, 1991; Aha, Kibler and Albert, 1991; Bartlett and Williamson, 1991]. Although incorporating such assumptions obviously makes pac-learning easier, notice that regardless of any such assumptions it is always still possible to consider a *sequential* approach to pac-learning. In fact, the next chapter shows that, even given strong distributional assumptions, sequential learning still obtains significant improvements over fixed-sample-size learning.

Overall, I believe the results presented in this chapter open the way to studying a much broader and more interesting class of learning algorithms than previously studied in computational learning theory.

[18] It seems clear that the current lower and upper bounds are both weak. That is, there is no reason to believe that the true worst case is significantly closer to the current upper bounds than the lower bounds. However, for the sake of argument, we adopt this position here.

Chapter 3

Distribution-*specific* sequential pac-learning

3.1 Introduction

As discussed in the previous chapter, pac-learning addresses the problem of learning an accurate approximation to some unknown target concept from random training examples, with the goal of producing an accurate hypothesis with some minimum specified probability (the pac-criterion). Of course the difficulty of achieving this criterion depends on the prior knowledge one has about the target concept and domain distribution.

Distribution-free pac-learning

Distribution-*free* (d.f.) pac-learning, as discussed in the previous chapter, adopts a model of prior knowledge where we assume the target concept belongs to some known class C but nothing is known about the underlying distribution of domain objects P.[1] Successful pac-learning under this model is characterized by the ability to meet the pac-criterion for arbitrary target concepts in C and arbitrary domain distributions P. The current theory of d.f. pac-learning provides a number of fixed-sample-size learning procedures that provably pac-learn the widest possible range of concept classes; namely, those with finite VCdimension. However, the training sample sizes required by these procedures is considered excessive in practice. This has lead to the common view that distribution-free pac-learning is inherently impractical; *i.e.*, meeting the pac-criterion for all domain distributions—even "pathological" ones—necessitates impractical training sample sizes. Despite the fact that Chapter 2 shows d.f. pac-learning can be achieved far more efficiently in practice than previously thought, via sequential learning, it is still possible that d.f. pac-learning is truly impractical in the worst case situation. This has motivated many researchers to explore distributional assumptions to improve the data-efficiency of pac-learning.

Distribution-specific pac-learning

The most extreme form of distributional assumption is to consider a *single* domain distribution P, and concentrate solely on identifying an unknown target concept from a known class C. This problem is known as distribution-*specific* (d.s.) pac-learning. Here we adopt a model of prior knowledge that assumes the domain distribution P is known *a priori*, but the target concept is known only to belong to some class C. That is, the learner's prior knowledge is characterized by concept *space* (C, P) rather than just a class C. Under this model, successful learning is characterized by the ability to meet the pac-criterion for any target concept in C, given full knowledge of P. As with d.f. pac-learning, current research [Benedek and Itai, 1988a, 1991; Kulkarni, 1991] has been able to provide correct learning procedures that provably pac-learn a wide variety of concept spaces. Moreover, a characterization of the inherent data-complexity of d.s.

[1] I will drop the subscript X from P_X throughout the remainder of this thesis (except when necessary in certain proofs).

pac-learning has been obtained in terms of the "metric entropy" of the concept space (C, P)—a measure of representational complexity that plays the same role as VCdimension in d.f. pac-learning [Benedek and Itai, 1988a, 1991; Kulkarni, 1991].

Issue and approach

However, just as in the d.f. theory, the current theory of d.s. pac-learning only considers fixed-sample-size learning procedures. The previous chapter observed how a sequential approach to learning could significantly reduce the number of training examples needed to achieve d.f. pac-learning in practice. Here we observe that the sequential approach is also directly applicable in the d.s. setting. This raises the obvious question of whether similar benefits can also be obtained here. In fact, it turns out that even stronger results can be achieved in this case.

Results

This chapter investigates the benefits of sequential learning in the d.s. context. First we propose a specific sequential learning procedure, that is then proved to be a correct pac-learner for any concept space (C, P) with finite metric entropy. An analysis of the expected data-efficiency of this procedure shows that it obtains a *five fold* improvement in data-efficiency over the previous fixed-sample-size approach. Moreover, an analysis of the intrinsic data-complexity of d.s. pac-learning shows that sequential learning scales the same as fixed-sample-size learning, so the *best* we can hope for is constant (or perhaps logarithmic) improvements in data-efficiency, as in the d.f. case. Again, however, as observed in Chapter 2, even constant reductions can have a significant impact in real applications.

Next, we explore an alternative technique for speeding up d.s. pac-learning that is orthogonal to sequential learning. The idea is to perform a *multiresolution* search for the target concept: that is, search the concept space by first finding a crude 1/2-approximation to the target, and then searching the 1/2-neighborhood of this concept for a 1/4-approximation, *etc.*; gradually refining the search until we find a sufficiently accurate hypothesis. It turns out that this strategy can yield substantial data (and even computational) efficiency improvements over previous approaches, for concept spaces with uniformly dense neighborhoods.

Finally, we observe that under the d.s. model, sequential learning procedures are able to learn with *certainty*, not just high probability. I demonstrate this by examining a few simple examples where a sequential learning procedure can be shown to return ϵ-approximations with probability 1 (wp1), not just probability $1 - \delta$ for $\delta > 0$. I refer to this problem as "certainly approximately correct" (cac) learning. Here a specific sequential learning procedure is proposed that correctly cac-learns a wide range of concept spaces. Analyzing the data-efficiency of this procedure shows that it actually cac-learns with *optimal* expected training sample size. Moreover, an analysis of the intrinsic data-complexity of cac-learning shows that this procedure is a universal cac-learner, in the sense that it solves any cac-learning problem where this is possible in principle. This analysis also reveals that there are pac-learnable concept spaces which are not cac-learnable; thus showing that cac-learning is intrinsically harder than pac-learning.

Surprisingly, for spaces which are both pac and cac learnable, the optimal cac-learning procedure is often far more data-efficient than the existing *pac*-learning procedures. This seems counterintuitive, since the cac-procedure is actually solving a harder learning problem. However, this result shows that even though cac-learning seems far removed from practical concerns, it can actually provide an effective alternative technique for deriving data-efficient pac-learning procedures.

Overview

This chapter explores the sequential approach to distribution-specific pac-learning. Before investigating various sequential learning strategies, Section 3.2 first surveys existing research on d.s. pac-learning; outlining the basic d.s. pac model and surveying the relevant results concerning the fixed-sample-size approach.

Section 3.3 then proposes a specific sequential learning procedure for the d.s. setting, proves it correct, and analyizes its expected data-efficiency. Given this efficient procedure we then study the inherent data-complexity of sequential d.s. pac-learning problems.

After introducing the sequential approach, Section 3.4 then considers a new multiresolution learning strategy that obtains additional improvements in some cases. In particular, we establish the data-efficiency of this technique for concept spaces which are uniformly dense across all neighborhoods. The real effectiveness of this approach is demonstrated through a series of case studies that reveals how it obtains improved efficiency in a variety of situations. However, it is also shown that this strategy cannot obtain a significant improvement in every possible case.

Next, Section 3.5 demonstrates how sequential strategies can learn with certainty in the d.s. setting; a problem I refer to as "certainly approximately correct" learning. Here we propose a simple generic learning procedure that turns out to cac-learn with optimal data-efficiency. An analysis of the inherent data-complexity of cac-learning problems shows that this procedure is applicable whenever cac-learning is possible. Finally, Section 3.6 reconsiders the problem of *pac*-learning, and applies this optimal cac-learning technique to substantially reduce the data-costs of pac-learning many spaces.

Section 3.7 concludes this chapter by discussing the implications of these results and suggesting directions for future research. Overall, our results demonstrate how sequential learning can reduce the number of training examples needed to pac-learn in the d.s. model.[2]

3.2 Background: distribution-specific pac-learning theory

Before investigating the effectiveness of sequential learning in the distribution-specific (d.s.) setting, we first review the basic definitions of d.s. pac-learning theory and survey existing results concerning correct learning procedures, their efficiency, and the inherent complexity of d.s. pac-learning.

3.2.1 Problem

As in the previous chapter, we consider the problem of learning a concept definition from examples under the i.i.d. model; demanding that the learner meet the pac-criterion in the worst case over all situations permitted by its prior knowledge. The difference here is that we consider an alternative model of prior knowledge where we assume the learner *knows* the domain distribution P *a priori*, but does not know which target concept c from a known class C is being used to generate the training examples. An instance of a d.s. pac-learning problem is specified by a concept space (C, P), an accuracy parameter ϵ, and a reliability parameter δ.

Definition 3.1 (Distribution-specific pac-learning) *A learner L solves the d.s. pac-learning problem $(C, \mathrm{P}, \epsilon, \delta)$ (or, "pac(ϵ, δ)-learns (C, P)"), if, given random training objects generated by* P *and labelled according to any target concept c in C, L produces a hypothesis h such that $\mathrm{P}\{h(x) \neq c(x)\} \leq \epsilon$ with probability at least $1 - \delta$.*

As before, we formalize a learner L as consisting of *(i)* a *sample size function* $T_L(C, \mathrm{P}, \epsilon, \delta)$ that determines a suitable training sample size for given C, P, ϵ, and δ; and *(ii)* a *hypothesizer* $H_L(C, \mathrm{P}, \epsilon, \delta) : (X \times \{0, 1\})^* \to \{0, 1\}^X$ that maps finite sequences of training examples to hypotheses—here assuming that we have access to the domain distribution P in addition to C, ϵ, and δ.

3.2.2 Procedures

Given the earlier results concerning d.f. pac-learning, the problem of d.s. pac-learning raises two immediate questions:

1. Do any new concept classes C (*i.e.*, with infinite VCdimension) become pac-learnable given specific domain distributions P?

2. Is the previous hypothesis-filtering strategy (collect a large training sample, then guess an arbitrary consistent hypothesis) still adequate for pac-learning these spaces?

[2]Some preliminary results from Sections 3.5 and 3.6 appear in [Schuurmans, 1996b]. Permission has been obtained from MIT Press for inclusion of this material here.

The answer to the first question is trivially *yes.* Any concept class C can be pac-learned with respect to specific distributions; *e.g.*, the distribution P that places all probability mass on a single domain object. However, the simplistic hypothesis filtering strategy is no longer adequate to achieve pac-learning in general under the d.s. model: the problem is that there are concept spaces that *cannot* be pac-learned by straightforward hypothesis filtering, and yet still can be pac-learned by more sophisticated guessing strategies. This is easily demonstrated by a simple example.

Example: Consider the concept space $(\{\text{finite sets}\} \cup \{[0,1]\}, \mathsf{uniform})$ defined on the unit interval $[0,1]$. Clearly, the concept class $\{\text{finite sets}\} \cup \{[0,1]\}$ on $[0,1]$ has infinite VCdimension. However this class can be easily pac(ϵ,δ)-learned with respect to the $\mathsf{uniform}$ distribution as follows: First, observe a single training example $\langle x, c(x)\rangle$; then, if $c(x) = 0$, output the hypothesis $\varnothing$, and otherwise (if $c(x) = 1$) output the hypothesis $[0,1]$. It is not hard to see that this procedure actually pac$(0,0)$-learns the space; for if $[0,1]$ is the target, the procedure always guesses $[0,1]$, and, on the other hand, if any finite set is the target, the procedure will guess $\varnothing$ with probability 1 (wp1). Thus in either case, the procedure's final hypothesis has *zero* error wp1. Notice that this space *cannot* be pac(ϵ,δ)-learned by guessing arbitrary consistent hypotheses, since if $[0,1]$ is the target then any finite sequence of training examples leaves consistent concepts that are a distance 1 away from $[0,1]$.

This example shows that there is much more to d.s. pac-learning than simply filtering bad hypotheses. So designing correct learning procedures for this case requires a bit more subtlety than the d.f. case. This raises the question of whether there are any general approaches to pac-learning under the d.s. model (like Procedure `F` in the d.f. case) or whether we have to resort to special methods for each problem. Fortunately, there is a simple, generic approach to d.s. pac-learning based on an intuitive metric space view of the problem [Benedek and Itai 1988a; 1991].

Metric space view

The key thing to notice about d.s. learning is that, under the i.i.d. random example model, a fixed domain distribution P induces a *metric* on the concept class, given by the probability that two concepts disagree on the classification of a random domain object:[3]

$$d_{\mathrm{P}}(c_1, c_2) \;\triangleq\; \mathrm{P}\{c_1(x) \neq c_2(x)\}.$$

Under this view, it is natural to think of C and P as comprising a concept *space* (C, P) with inter-concept distances determined by d_{P}. Here, the *error* of a hypothesis h relative to a target concept c is simply given by its distance $d_{\mathrm{P}}(h,c)$ from c. The goal of pac-learning then is to reliably produce a hypothesis h such that $d_{\mathrm{P}}(h,c) \leq \epsilon$.

Benedek and Itai's approach to d.s. pac-learning is based on exploiting the existence of a small, finite "cover" of the concept space.

Definition 3.2 (Cover) *For a concept space (C,P), an* ϵ-cover *is a finite set of concepts $V = \{h_1, h_2, ..., h_N\}$ such that for every $c \in C$ there is an $h \in V$ where $d_{\mathrm{P}}(c,h) \leq \epsilon$. The size of the* smallest ϵ-*cover of* (C,P) *is denoted $N_\epsilon(C,\mathrm{P})$.*

Given a space (C,P), accuracy parameter ϵ, and reliability parameter δ, Benedek and Itai propose a procedure that pac-learns by *(i)* constructing a small $\epsilon/2$-cover of the space, V; *(ii)* collecting a sufficiently large training sample to accurately estimate the errors of the hypotheses in V; and *(iii)* returning the cover-hypothesis that obtains the smallest observed error; see Procedure `BI` in Figure 3.1.

Correctness

The correctness of `BI` follows from the fact that its fixed training sample size is sufficient to ensure that any hypothesis with error at most $\epsilon/2$ has a smaller observed error than any hypothesis with error greater than ϵ, with probability at least $1-\delta$.

[3]It is easily verified that this definition satisfies the standard properties of a (pseudo)metric: positivity, identity zero, symmetry, and the triangle inequality.

Procedure `BI` $(C, \mathrm{P}, \epsilon, \delta)$

INPUT: target concept space (C, P),
accuracy parameter ϵ,
reliability parameter δ.

RETURN: a hypothesis h with accuracy at least $1-\epsilon$, with probability at least $1-\delta$.

PROCEDURE:

- Find an $\epsilon/2$-cover of (C, P), V, with size $|V| = N_{\epsilon/2}(C, \mathrm{P})$.
- Collect $\frac{32}{\epsilon} \ln \frac{|V|}{\delta}$ training examples labelled by some unknown target concept $c \in C$.
- Return the hypothesis $h \in V$ with minimum observed error.

Figure 3.1: Procedure `BI`

Proposition 3.3 [Benedek and Itai, 1988a] *For any $\epsilon > 0$, $\delta > 0$, and any finite collection of hypotheses $V = \{h_1, ..., h_N\}$,*

$$T_{BI}(V, \epsilon, \delta) \;=\; \frac{32}{\epsilon}\left(\ln|V| + \ln\frac{1}{\delta}\right)$$

random training examples are sufficient to ensure that, with probability at least $1-\delta$, every $h \in V$ with $d_{\mathrm{P}}(h, c) \le \epsilon/2$ has observed error less than $3\epsilon/4$, and every $h \in V$ with $d_{\mathrm{P}}(h, c) \ge \epsilon$ has observed error greater than $3\epsilon/4$.

Therefore, since V is guaranteed to contain a hypothesis with error at most $\epsilon/2$ (because V is an $\epsilon/2$-cover), `BI` cannot produce an ϵ-bad hypothesis with probability greater than δ. This means that `BI` will correctly pac-learn any concept space (C, P) for which we can find a *finite* $\epsilon/2$-cover at the appropriate scale ϵ. This turns out to be true of most concept spaces encountered in practice. In fact, any concept space (C, P) that is formed from a concept class C with finite VCdimension is guaranteed to be finitely ϵ-coverable at all scales $\epsilon > 0$, and hence can be pac-learned by `BI`.

Proposition 3.4 [Haussler, 1992] *For any $\epsilon > 0$, any class C, and any distribution* P,

$$N_\epsilon(C, \mathrm{P}) \;\le\; 2\left(\frac{2e}{\epsilon}\ln\frac{2e}{\epsilon}\right)^{\mathrm{vc}(C)}.$$

Thus, `BI` is a general d.s. pac-learning procedure that can be applied to a wide range of concept spaces. For example, `BI` easily pac-learns the space $(\{\text{finite sets}\} \cup \{[0,1]\}, \mathsf{uniform})$ considered earlier, since an ϵ-cover of this space for all $\epsilon \ge 0$ is given by $\{\varnothing, [0,1]\}$.

However, not every concept space can be finitely ϵ-covered for all $\epsilon > 0$, and hence `BI` cannot be used to pac-learn every possible space. For example, consider the concept space $(\mathcal{B}, \mathsf{uniform})$ on $[0,1]$, where $\mathcal{B}$ is the Borel subsets of $[0,1]$.[4] This space has no finite ϵ-cover for $\epsilon < 1/4$ and therefore cannot be pac-learned by Procedure `BI`. (To see this: notice that the concept $\bigcup_{i=0}^{n-1}[\,i/n\,,\ (i{+}1/2)/n\,]$, a disjoint union of n intervals, is at least a distance $1/4$ from any Borel concept containing $n/2$ or fewer subintervals. Now consider the sequence of concepts defined by $n = 1, 2, 4, 8, ...$ subintervals, all of which belong to $\mathcal{B}$. These concepts are pairwise separated by a distance of $1/2$, and therefore, for $\epsilon < 1/4$, no finite set of concepts can ϵ-cover this sequence.) The question of whether these spaces can be pac-learned by alternative strategies to `BI` is addressed in Section 3.2.4 below.

[4] The Borel subsets of $[0,1]$ are those sets that can be defined by a countable union of subintervals.

3.2.3 Efficiency

Given the correctness of a pac-learning procedure, the key issue becomes determining its efficiency; both in terms of the number of training examples it observes (data-efficiency), and the computational resources it requires to produce a hypothesis (computational-efficiency).

Data-efficiency

Since BI is a fixed-sample-size learning procedure, its data-efficiency is directly determined by the sufficient sample size function it uses. For Procedure BI this sample size is given by

$$T_{\mathtt{BI}}(C, \mathrm{P}, \epsilon, \delta) \;=\; \frac{32}{\epsilon}\left(\ln N_{\epsilon/2}(C, \mathrm{P}) + \ln\frac{1}{\delta}\right)$$

in terms of the desired accuracy and reliability parameters ϵ and δ, and the size of the smallest $\epsilon/2$-cover of the space (C, P). Interestingly, $T_{\mathtt{BI}}$ scales *linearly* in the quantity $\ln N_{\epsilon/2}(C, \mathrm{P})$, which is commonly known as the *metric entropy* of the concept space at scale $\epsilon/2$ [Kulkarni, 1991]. Metric entropy quantifies the representational complexity of a concept space in the d.s. setting in much the same way as VCdimension characterizes the representational complexity of a concept class in the d.f. setting. In both cases, these complexity measures determine linear bounds on the number of training examples sufficient to pac-learn in their respective models.

Computational-efficiency

Aside from data-efficiency, we can also consider the computational resources required by BI to produce its hypotheses. For BI, the main computational task is finding an $\epsilon/2$-cover of the space and then storing and manipulating this cover once found. Unfortunately, this is rarely feasible in practice. The problem is that the size of the minimum ϵ-cover for most concept spaces grows as $(1/\epsilon)^d$ in terms of the natural dimensionality d of space [Kolmogorov and Tihomirov, 1961; Kulkarni, 1991; Haussler, 1992]. This leads to an exponential explosion in cover sizes since d usually corresponds to the natural size parameter n for the problem. Moreover, the task of finding a smallest ϵ-cover is usually quite difficult for most concept spaces. Therefore, Procedure BI cannot usually be efficiently implemented as is in practice. Most computationally-efficient d.s. pac-learning procedures reported in the literature employ special techniques specific to the concept space (family) at hand.

Many efficient procedures have been developed for pac-learning concept space families defined on $\{0,1\}^n$, achieving computation times bounded by a polynomial in n, $1/\epsilon$, and $1/\delta$. Most research on computationally efficient pac-learning algorithms has focused on special cases of the concept space family $(\mathsf{dnf}, \mathsf{uniform})$ on $\{0,1\}^n$. First, notice that polynomial time pac-learning is trivially achieved for simple concept spaces like $(k\mathsf{dnf}, \mathsf{uniform})$ where $k\mathsf{dnf}$ itself can be pac-learned in polynomial time under the d.f. model. In fact, this will be true for any concept space family $\{(C_n, \mathrm{P}_n)\}$ where the concept classes $\{C_n\}$ themselves are pac-learnable in polynomial time under the d.f. model. However, it is also possible to efficiently learn certain spaces $\{(C_n, \mathrm{P}_n)\}$ for which no polynomial time learning algorithm is known for the corresponding classes $\{C_n\}$. Two examples of this are the spaces $(\mu\mathsf{dnf}, \mathsf{uniform})$ [Kearns et al., 1987a] and $(k\mu\mathsf{dnf}, \mathsf{uniform})$ [Hancock and Mansour, 1991] where polynomial time learning procedures have been achieved, even though it is not known whether $\mu\mathsf{dnf}$ and $k\mu\mathsf{dnf}$ themselves can be pac-learned in polynomial time under the d.f. model. It is currently an open question as to whether $(\mathsf{dnf}, \mathsf{uniform})$ can be pac-learned in polynomial time under the d.s. model (as for dnf under the d.f. model). However, Verbeurgt [1990] has developed a quasi-polynomial time procedure for pac-learning this space.[5]

Aside from these restricted classes of dnf formulae, polynomial time procedures have also been developed for more general classes of boolean formulae under the $\mathsf{uniform}$ distribution. For example, it has been shown that $(\mu\text{-}\mathsf{maj\text{-}formulae}, \mathsf{uniform})$ [Goldman, Kearns and Schapire, 1990] and $(\mu\mathsf{formulae}, \mathsf{uniform})$ [Schapire, 1992] can both be pac-learned in polynomial time. Quasi-polynomial time learning procedures have also been developed for $(\mathsf{AC}^0, \mathsf{uniform})$ [Linial, Mansour and Nisan, 1989; Furst, Jackson and Smith, 1991].

[5] Jackson [1994] has recently demonstrated a polynomial time procedure for pac-learning $(\mathsf{dnf}, \mathsf{uniform})$. However, his procedure actively *queries* the domain for the classifications of specific domain objects, and hence does not properly fall under the passive observation model we are considering in this thesis.

3.2.4 Complexity

Aside from determining the correctness and efficiency of specific pac-learning procedures, it is also important to considers the intrinsic difficulty of solving d.s. pac-learning problems; *i.e.*, the minimum resources required for *any* learner to meet the pac-criterion for a particular concept space (C, P). Again, this difficulty can be measured along two distinct dimensions: data-complexity and computational-complexity.

Data-complexity

Restricting attention to the fixed-sample-size approach, we can characterize the (fixed) data-complexity of pac-learning a concept space (C, P) by the minimum number of training examples required by any (fixed-sample-size) learning procedure to successfully meet the pac-criterion in the worst case over all $c \in C$. This type of analysis allows us to assess the true data-efficiency of pac-learning procedures like BI, by comparing their data-efficiency to the best performance possible, and determining what, if any, improvements are possible. Benedek and Itai [1988a; 1991] have established the following lower bound on the inherent data-complexity of solving a d.s. pac-learning problem $(C, \mathrm{P}, \epsilon, \delta)$ in terms of the metric entropy of the space.

Theorem 3.5 [Benedek and Itai, 1988a; 1991] *For any $\epsilon > 0$, $\delta > 0$, and any concept space (C, P): Any (fixed-sample-size) learning procedure that observes fewer than*

$$t_{BI}(C, \mathrm{P}, \epsilon, \delta) \quad = \quad \log_2 \left[N_{2\epsilon}(C, \mathrm{P}) \cdot (1 - \delta) \right]$$

random training examples will fail to meet the pac(ϵ, δ)-criterion for some $c' \in C$.

Thus, the metric entropy of a concept space gives a non-trivial lower bound on the minimum number of training examples needed to achieve pac-learning. This shows that finite metric entropy is not only sufficient for BI to correctly pac-learn, but is also necessary for there to be any learning procedure that successfully pac-learns the space (C, P). This proves, for example, that the space $(\mathcal{B}, \mathsf{uniform})$, which has infinite metric entropy for all $\epsilon < 1/4$, cannot be pac-learned by any procedure with a bounded number of training examples. This also shows that BI is a *universal* d.s. pac-learning procedure in the sense that it can pac-learn any concept space for which this is possible in principle.

It is also interesting to note that the lower bound t_{BI} scales linearly in the metric entropy $N_{2\epsilon}(C, \mathrm{P})$ of the space, which matches BI's scaling behavior (for fixed ϵ and δ). Therefore, not only is BI a universal d.s. learner, its data-efficiency scales-up near optimally in terms of the metric entropy of the concept space (up to $1/\epsilon$ and $\ln(1/\delta)$ factors). So, metric entropy measures the representational complexity of a concept space in much the same way as VCdimension measures the complexity of a concept class: both determine linear bounds on the optimum achievable data-efficiency under their respective learning models.

Computational-complexity

Aside from determining the minimum number of training examples needed to pac-learn, it is also important to consider the minimum computational resources required. As with d.f. learning, it is customary to consider how the difficulty of learning a parameterized *family* of concept spaces scales up with problem size.

Definition 3.6 (Feasible learnability) *A family $\{(C_n, \mathrm{P}_n)\}$ of concept spaces is feasibly pac-learnable if there is a learning algorithm L, taking n, ϵ, and δ as input, that solves the pac-learning problems $(C, \mathrm{P}, \epsilon, \delta)$ with training sample size and running time bounded by a polynomial in n, $1/\epsilon$, and $1/\delta$.*

As usual, proving that a family of concept spaces is feasibly learnable is simply a matter of producing a polynomial time learning algorithm for the family, *e.g.*, as in [Kearns et al., 1987a] for $(\mu\mathsf{dnf}, \mathsf{uniform})$. Moreover, infeasibility follows trivially for concept space families that have super-polynomial data-complexity. However, it is usually quite difficult to determine whether a family that has polynomial data-complexity also has a computationally feasible pac-learning procedure. For example, it remains an open question as to whether the space $(\mathsf{dnf}, \mathsf{uniform})$ is pac-learnable in polynomial as opposed to quasi-polynomial time, even though it clearly has polynomial *data*-complexity. Overall, proving that any data-feasible concept family is nonetheless computationally-*in*feasible appears to be an extremely difficult challenge. In fact, the first results in this area have only recently been achieved by Kharitonov [1993], who has shown that pac-learning the space (boolean-formulae, uniform) is infeasible, given certain cryptographic assumptions. (It is not hard to show that this space has polynomial data-complexity.)

3.2.5 Relationship to d.f. pac-learning

Clearly, there are strong relationships between the difficulty of pac-learning a concept class C under the d.f. model, and pac-learning a corresponding concept space (C, P) under the d.s. model.

Data costs

In terms of the number of training examples required, it is obviously easier to pac-learn under the d.s. model than the d.f. model; *i.e.*, it can take no more training examples to pac-learn a concept class C with respect to a single domain distribution P, than it can to pac-learn C with respect to arbitrary distributions. So if C is pac-learnable under the d.f. model, then automatically so is (C, P) under the d.s. model for any P. However, there are pac-learnable concept spaces (C, P) where C cannot be pac-learned under the d.f. model.

Observation 3.7 *For pac-learning with bounded data-efficiency for every $\epsilon > 0$, $\delta > 0$:*

$$\begin{array}{rcl} C \text{ pac-learnable} & \Leftrightarrow & \mathrm{VC}(C) < \infty \\ \Downarrow \ \ \not\Uparrow & & \\ (C, \mathrm{P}) \text{ pac-learnable} & \Leftrightarrow & N_\epsilon(C, \mathrm{P}) < \infty \quad \forall \epsilon > 0. \end{array}$$

The preceding results also show how the difficulty of pac-learning a class C, either with respect to all possible domain distributions, or just with respect to a single distribution P, can be characterized by different measures of problem complexity.

Computational costs

In terms of computational requirements, it is again obvious that distributional restrictions can only make pac-learning easier. In fact, even natural domain distributions sometimes significantly reduce the computational-complexity of pac-learning. For example, Schapire [1992] has shown that (μformulae, uniform) is efficiently learnable, whereas μformulae itself *cannot* be pac-learned in polynomial time under the d.f. model, unless certain cryptographic assumptions are false [Kearns and Valiant, 1989]. In this case, the additional information provided by the uniform distribution, while only slightly reducing the required number of training examples, significantly reduces the computational difficulty of pac-learning.

3.3 Sequential d.s. pac-learning

Even though the strong distributional assumptions made by the d.s. model can significantly reduce the data-complexity of pac-learning, it is still possible to consider a sequential approach to learning in hopes of obtaining even further efficiency improvements. Here we pursue the same strategy as Chapter 2 and ask whether sequential learning strategies can substantially reduce the number of training examples needed to pac-learn under the d.s. model. This section introduces a particular sequential learning strategy for the d.s. case, investigates its data-efficiency, and determines the inherent data-complexity of sequential d.s. pac-learning.

3.3.1 Problem

Here we are addressing the same learning task outlined in the previous section: for a specified concept space (C, P), accuracy parameter ϵ, and reliability parameter δ, we demand that the learner produce an ϵ-accurate hypothesis with probability at least $1 - \delta$, for any target concept c in C. The only difference is that now we allow the learner to autonomously choose the size of its own training sample based on the particular training sequence it observes, rather than just observing a fixed number of training examples. So, as in Chapter 2, we generalize the definition of a learner's stopping rule T_L to depend on the observed training sequence, as well as the problem parameters C, ϵ, δ, and P.

Procedure `Sbi` $(C, \mathrm{P}, \epsilon, \delta)$

INPUT: target concept space (C, P),
accuracy parameter ϵ,
reliability parameter δ.

RETURN: a hypothesis h with accuracy at least $1-\epsilon$, with probability at least $1-\delta$.

PROCEDURE:

- Find an $\epsilon/2$-cover of (C, P), V, with size $|V| = N_{\epsilon/2}(C, \mathrm{P})$.
- Sequentially observe training examples $\langle x_t, c(x_t)\rangle$, $t = 1, 2, ...,$ *etc.*, labelled by some unknown target concept $c \in C$:
  - Subject each $h_i \in V$ to a statistical test that decides

$$H_{acc} : \mathrm{P}\{h_i(x) \neq c(x)\} \leq \epsilon/2 \quad \text{versus} \quad H_{rej} : \mathrm{P}\{h_i(x) \neq c(x)\} \geq \epsilon,$$

    with a probability of deciding H_{acc} when H_{rej} is true bounded by $\delta/|V|$, and *zero* probability of deciding H_{rej}. (This is done by calling the subroutine `sprt(` $h_i(x) \neq c(x)$, $\epsilon/2$, ϵ, $\delta/|V|$, 0 `)`; see Figure 2.4 in Chapter 2.)
  - If some h_i in the list passes the test, halt and return h_i.
- Repeat until some h_i passes the test.

Figure 3.2: Procedure `Sbi`

3.3.2 Procedure

As mentioned, the generic approach to d.s. pac-learning, Procedure `BI`, is based on finding a finite cover of the concept space and then estimating the errors of these cover-concepts to identify any acceptable candidate. Here we consider a sequential approach to this problem that also follows the cover-based approach, but now adopting a sequential rather than fixed-sample-size approach to error estimation. In particular, we consider a procedure, `Sbi`, that first finds an $\epsilon/2$-cover of the concept space, but then tests each cover-hypothesis (in parallel) by subjecting it to a sequential probability ratio test (`sprt`). The first cover-concept that passes this test is returned as the final hypothesis; see Figure 3.2. The idea is to improve the data-efficiency of `BI` simply by using `sprt` to quickly identify accurate candidate hypotheses.

It is not hard to see that Procedure `Sbi` is a correct pac-learner for any concept space that has a finite $\epsilon/2$-cover. The key property of `Sbi` is that its call to `sprt` eventually accepts any $\epsilon/2$-good hypothesis wp1, but only accepts an ϵ-bad hypothesis with probability at most $\delta/|V|$, where $|V|$ is the size of the $\epsilon/2$-cover it constructs. So `Sbi` returns an ϵ-bad hypothesis with probability at most $|V| \times \delta/|V| = \delta$, and yet must eventually terminate wp1 since V is guaranteed to contain an $\epsilon/2$-good hypothesis by construction. Therefore, the probability that `Sbi` returns an ϵ-good hypothesis must be at least $1-\delta$.

Theorem 3.8 (Correctness) *For any $\epsilon > 0$, $\delta > 0$, and any concept space (C, P) with $N_{\epsilon/2}(C, \mathrm{P}) < \infty$: Given i.i.d. examples generated by* P *and any target concept $c \in C$, Procedure* `Sbi` *returns a hypothesis h such that $\mathrm{P}\{h(x) \neq c(x)\} \leq \epsilon$ with probability at least $1-\delta$.*[6]

As before we note that this is a wide range of concept spaces, covering most situations encountered in practice.

[6] Proofs of all (original) results stated in this chapter are given in Appendix B.

3.3.3 Efficiency

Given the correctness of `Sbi`, the key issue becomes determining its data-efficiency, and comparing this to the efficiency of the fixed-sample-size approach. As in Chapter 2, we compare the relative data-efficiencies of sequential and fixed-sample-size learners by comparing their average and fixed training sample sizes directly.

Definition 3.9 (Expected data-efficiency) *The* expected data-efficiency *of a sequential learner L, for solving a d.s. pac-learning problem $(C, \mathrm{P}, \epsilon, \delta)$, is given by the maximum expected training sample size L uses in the worst case over all possible targets $c \in C$.*

Using the analysis of the `sprt` procedure derived in the previous chapter it is possible to determine a reasonable upper bound on the expected number of training examples `Sbi` observes in the worst case over all targets in C.

Theorem 3.10 (Data-efficiency) *For any $\epsilon > 0$, $\delta > 0$, and any concept space (C, P) with $N_{\epsilon/2}(C, \mathrm{P}) < \infty$: Given i.i.d. examples generated by* P *and any target concept $c \in C$, Procedure* `Sbi` *observes an average training sample size of at most*

$$\mathrm{E}\, T_{\mathtt{Sbi}}(C, \mathrm{P}, \epsilon, \delta) \ \leq \ \frac{6.5178}{\epsilon}\left(\ln N_{\epsilon/2}(C, \mathrm{P}) + \ln \frac{1}{\delta} + 1\right). \tag{3.1}$$

Interestingly, this upper bound scales the same as `BI`'s fixed-sample-size bound $T_{\mathtt{BI}}$ in terms of ϵ, δ, and the metric entropy of the concept space. However, this result shows that `Sbi` observes about five times fewer training examples on average than Procedure `BI` for all values of ϵ and δ. This is a much stronger theoretical improvement than was achieved in the d.f. case, as well as a significant practical improvement.

3.3.4 Complexity

However, the preceding analysis leaves open the question of whether further efficiency improvements are possible, and, if so, how much of an improvement is possible in principle. To this end, we consider the intrinsic data-complexity of sequentially pac-learning a concept space (C, P); *i.e.*, the minimum average number of training examples any sequential learner must observe to meet the pac-guarantees in the worst case.

Definition 3.11 (Expected data-complexity) *The* expected data-complexity *of a d.s. pac-learning problem $(C, \mathrm{P}, \epsilon, \delta)$ is given by the smallest average number of training examples any learning procedure must observe to meet the pac(ϵ, δ)-criterion (for any fixed $c \in C$), in the worst case over all possible target concepts in C.*

We have seen that `Sbi`'s expected data-efficiency is at least five times smaller than `BI`'s fixed sample size. Theorem 3.5 in Section 3.2 shows that no fixed-sample-size learner can improve on the data-efficiency of `BI` by more than a constant factor in terms of the metric entropy of the space, for fixed ϵ and δ. However, this leaves open the question of whether a sequential learning procedure might do substantially better than this; *e.g.*, scale up sub-linearly in the metric entropy of the space. It turns out the answer to this question is *no*. The next result establishes a lower bound on the minimum average number of training examples any sequential learner must observe to successfully pac-learn a concept space.

Theorem 3.12 (Data-complexity) *For any $\epsilon > 0$, $\delta > 0$, and any concept space (C, P): A learner that always observes an average training sample size less than*

$$t_{avg}(C, \mathrm{P}, \epsilon, \delta) \ = \ \frac{1}{2}\log_2\left[N_{2\epsilon}(C, \mathrm{P})\left(\frac{1}{2} - \delta\right)\right]$$

for every fixed $c \in C$ will fail to meet the pac(ϵ, δ)-criterion for some fixed target $c' \in C$.

This shows that no new concept spaces become pac-learnable simply by considering a sequential over fixed-sample-size learning approach. That is, the concept space (C, P) must have finite metric entropy for any learner to be able to meet the pac-criterion for every target concept in C with a uniform bound on expected training sample size. Thus, having finite metric entropy is not only sufficient for `Sbi` to be able to pac-learn a concept space, but also necessary for any sequential learner to be successful. This also shows that `Sbi` is a *universal* pac-learning strategy in the same sense as `BI`: `Sbi` correctly pac-learns any concept space (C, P) for which this is possible via any learning procedure.

Notice that the lower bound t_{avg} actually scales the same as the fixed-sample-size bound t_{BI}. This suggests that the data-efficiency of sequential learning cannot scale up significantly better than fixed-sample-size learning, as their respective lower bounds differ only by constant factors. This result also shows that `Sbi`'s data-efficiency scales up near optimally in terms of the metric entropy of the concept space (up to $1/\epsilon$ and $\ln(1/\delta)$ factors). Of course, `Sbi` still may not be the most efficient learning procedure possible. In fact, we will see that its performance can be substantially improved in special cases.

Overall, the results of this section show that metric entropy continues to be a natural measure of concept space complexity: it continues to determine linear bounds on the achievable data-efficiency of sequential, as well as fixed-sample-size d.s. pac-learning.

3.4 Multiresolution learning

Independent of whether we pursue a sequential or fixed-sample-size approach to learning, another simple idea for improving data-efficiency is to perform a *multiresolution* search for the target concept. That is, rather than searching a fixed $\epsilon/2$-cover of the entire space, we can first perform a crude search at some large scale (say $1/2$) and then refine this search to within smaller neighborhoods of the space, progressively zeroing in on the target concept. In this way we can substantially reduce the number of candidate hypotheses the learner must consider, which can lead to both a significant improvement in the data-efficiency of pac-learning as well as a substantial reduction in computational costs.

3.4.1 Procedure

The particular multiresolution search procedure we consider, Procedure `Sfoc`, works by first fixing a crude $1/4$-cover of the space and estimating errors to within $1/4$, yielding a hypothesis with error at most $1/2$; then fixing a $1/8$-cover of the $1/2$-neighborhood of this hypothesis and estimating these errors to within $1/8$, yielding a hypothesis with error at most $1/4$; then fixing a $1/16$-cover of the $1/4$-neighborhood of this hypothesis and estimating errors to within $1/16$, yielding a hypothesis with error at most $1/8$; *etc.*; until a hypothesis with error at most ϵ is found with high probability (after $\log_2(1/\epsilon)$ stages); see Figure 3.3.

The benefit of this over the global-cover based approach is twofold: First, `Sfoc` considers fewer candidate hypotheses since it only constructs fine-grained covers for small local neighborhoods of the target concept. Second, `Sfoc` only estimates the errors of hypotheses *near* the target concept with high accuracy, as crude estimates suffice to eliminate candidates that are further away. In this way `Sfoc` finds an accurate hypothesis without having to consider a complete $\epsilon/2$-cover of the entire space, and without having to estimate the error of every candidate hypothesis with the same degree of accuracy. This can lead to significant savings in practice, both in terms of the number of training examples needed to meet the pac-criterion, and in terms of the computational complexity of pac-learning. In fact, we will see that `Sfoc` even can achieve polynomial time pac-learning for certain concept spaces that have exponentially large covers.

Notice however that `Sfoc` cannot achieve significant improvements in every possible case. The problem is that the concept space might be extremely dense in certain small neighborhoods of the space—to the extent that an α-cover of a single neighborhood might be nearly as large as the α-cover of the entire space. (For example, consider a space consisting of d disjoint sets of size $\epsilon < 1/d$, and $\varnothing$. For $\beta > \alpha > \epsilon$, any α-cover of $\varnothing$'s β-neighborhood is also an α-cover of the entire space.) In such cases `Sfoc` cannot avoid considering a large number of candidates, and estimating their errors with a high degree of accuracy. Therefore, we only expect `Sfoc` to obtain a significant savings over `BI` and `Sbi` for spaces which are more or less uniformly dense over local neighborhoods. I formalize this notion below.

Procedure `Sfoc` $(C, \mathrm{P}, \epsilon, \delta)$

INPUT: target concept space (C, P),
accuracy parameter ϵ,
reliability parameter δ.

RETURN: a hypothesis h with accuracy at least $1-\epsilon$, with probability at least $1-\delta$.

PROCEDURE:

- Let $B_\alpha(c) \triangleq \{h \in C : d_{\mathrm{P}}(h, c) \leq \alpha\}$ denote the (closed) ball of radius α centered at concept $c \in C$.
- Sequentially observe training examples $\langle x_t, c(x_t)\rangle$, $t = 1, 2, \ldots$, *etc.*, labelled by some unknown target concept $c \in C$:
- Repeat for stages $i = 1, \ldots, S$, where $S = \lceil \log_2(1/\epsilon) \rceil$:

```
Stage 1: Find a 1/4-cover of $(C,\mathrm{P})$, $V_1$, with size $|V_1| = N_{1/4}(C,\mathrm{P})$.
    Call sprt( $h(x) \neq c(x)$, $1/4$, $1/2$, $\delta/(|V_1|S)$, $0$ ) for each $h \in V_1$.
    Let $h_1$ be the first candidate sprt accepts from $V_1$.
    (After Stage 1 we know that $c \in B_{1/2}(h_1)$ with probability at least $1 - \delta/S$.)
Stage 2: Find a 1/8-cover of $B_{1/2}(h_1)$, $V_2$, with size $|V_2| = N_{1/8}B_{1/2}(h_1)$.
    Call sprt( $h(x) \neq c(x)$, $1/8$, $1/4$, $\delta/(|V_2|S)$, $0$ ) . for each $h \in V_2$.
    Let $h_2$ be the first candidate sprt accepts from $V_2$.
    (After Stage 2 we know that $c \in B_{1/4}(h_2)$ with probability at least $1 - 2\delta/S$.)
    ⋮
Stage i: Find a $2^{-(i+1)}$-cover of $B_{2^{-(i-1)}}(h_{i-1})$, $V_i$, with size $|V_i| = N_{2^{-(i+1)}}B_{2^{-(i-1)}}(h_{i-1})$.
    Call sprt( $h(x) \neq c(x)$, $2^{-(i+1)}$, $2^{-i}$, $\delta/(|V_i|S)$, $0$ ) for each $h \in V_i$.
    Let $h_i$ be the first candidate sprt accepts from $V_i$.
    (After Stage $i$ we know that $c \in B_{2^{-i}}(h_i)$ with probability at least $1 - i\delta/S$.)
    ⋮
Stage S: Find a $\epsilon/2$-cover of $B_{2\epsilon}(h_{S-1})$, $V_S$, with size $|V_S| = N_{\epsilon/2}B_{2\epsilon}(h_{S-1})$.
    For each $h \in V_S$ call sprt( $h(x) \neq c(x)$, $\epsilon/2$, $\epsilon$, $\delta/(|V_S|S)$, $0$ ).
    Let $h_S$ be the first candidate sprt accepts from $V_S$.
    (After Stage $S$ we know that $c \in B_\epsilon(h_S)$ with probability at least $1 - \delta$.)
```

- Return h_S.

Figure 3.3: Procedure `Sfoc`

3.4.2 Spaces with invariant dimension

Section 3.2.3 noted that for most spaces, the size of the smallest ϵ-cover grows roughly as $(1/\epsilon)^d$ in terms of the natural dimensionality d of the space. Therefore, it is natural to define the "effective" dimension of a concept space (C, P) by

$$\dim_\epsilon(C, \mathrm{P}) \ \triangleq \ \frac{\ln N_\epsilon(C, \mathrm{P})}{\ln(1/\epsilon)};$$

in this way picking out the dimensionality exponent d. However, notice that this measure can change for different values of ϵ; *e.g.*, the effective dimension of any finite space decreases to zero for small ϵ. To avoid this variability, many authors have sought scale-independent notions of effective dimensionality. For example, Kolmogorov and Tihomirov [1961], and Haussler [1992] define the upper and lower metric dimension of a space by $\overline{\lim}\dim_\epsilon$ and $\underline{\lim}\dim_\epsilon$ respectively. In the case where these coincide, Haussler refers to their common limit as the *metric dimension* of (C, P). Notice that the metric dimension of a space with $N_\epsilon(C, \mathrm{P}) = (polylog(\epsilon)/\epsilon)^d$ is d, so the metric dimension basically picks out the exponent of the cover size as a function of $1/\epsilon$.

However, here we are mainly interested in upper bounds, so I adopt the following, simpler definition:

Definition 3.13 (Scale-invariant (upper) dimension) *A concept space (C, P) has scale-invariant (upper) dimension d if there is some constant α such that $N_\epsilon(C, \mathrm{P}) \le (\alpha/\epsilon)^d$ for all $\epsilon > 0$. If no such d and α exists, then (C, P) has infinite (upper) dimension.*

As mentioned above, we expect `Sfoc` to demonstrate an advantage for concept spaces whose effective dimensionality is invariant across neighborhoods, not just independent of scale. So we need a stronger definition to capture this form of invariance.

Definition 3.14 (Neighborhood-invariant (upper) dimension)

- *For a concept $c \in C$, let $B_\epsilon(c)$ denote the ϵ-neighborhood of c in (C, P).* I.e., *$B_\epsilon(c) \triangleq \{h \in C : d_{\mathrm{P}}(h, c) \le \epsilon\}$; the (closed) ball of radius ϵ centered at c.*
- *Then, for a constant $k > 1$ we say that a concept space (C, P) has neighborhood-invariant (upper) dimension d if there is some constant $b > 1$ such that $N_\epsilon B_{k\epsilon}(c) \le (bk)^d$ for all $c \in C$ and all $\epsilon > 0$;* i.e., *the $k\epsilon$-neighborhood of any concept can be ϵ-covered by at most $(bk)^d$ concepts.*
- *If no such d exists, then (C, P) has infinite (upper) dimension.*
- *The size of the largest neighborhood cover is $NB_k(C, \mathrm{P}) \triangleq \sup_{c \in C, \epsilon > 0} N_\epsilon B_{k\epsilon}(c)$.*

Clearly then, neighborhood-invariance implies scale-invariance.

Proposition 3.15 $\quad NB_k(C, \mathrm{P}) \le (bk)^d$ *implies* $N_\epsilon(C, \mathrm{P}) \le \left(\frac{k}{\epsilon}\right)^{d\left(\frac{\ln b}{\ln k} + 1\right)}$.

This definition captures the idea that a space with neighborhood-invariant dimension should be uniformly dense across all local neighborhoods of the space; *i.e.*, the relative covers of local neighborhoods should be the same size regardless of their location in the space. Not surprisingly, many natural concept spaces encountered in practice are uniformly dense in this manner, *cf.* Section 3.4.4 below. It is for these spaces that we expect `Sfoc` to obtain a substantial performance advantage over the global-cover based learning techniques `BI` and `Sbi`.

3.4.3 Efficiency

It turns out that we can prove a substantial data-efficiency advantage for Procedure `Sfoc` over the global-cover based procedures `BI` and `Sbi` for concept spaces which are uniformly dense across all local neighborhoods. In particular, for a concept space (C, P) with finite neighborhood-invariant dimension, we can derive a reasonable upper bound on the expected number of training examples `Sfoc` uses to pac-learn the space.

Proposition 3.16 (Data-efficiency) *For any $\epsilon > 0$, $\delta > 0$, and any concept space (C, P) with $NB_4(C, \mathrm{P}) < \infty$: Procedure* `Sfoc` *observes an average training sample size of at most*

$$\mathrm{E}\, T_{\mathtt{Sfoc}}(C, \mathrm{P}, \epsilon, \delta) \;<\; \frac{13.0356}{\epsilon}\left(\ln NB_4(C,\mathrm{P}) + \ln\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta} + 1.4\right). \tag{3.2}$$

for any target concept $c \in C$.

This result shows that `Sfoc` is fundamentally more data-efficient than the global-cover based techniques `BI` and `Sbi` for uniformly dense concept spaces. That is, if the space (C, P) has neighborhood-invariant dimension d and also effective dimension d for all ϵ, then it is easy to see `Sfoc`'s data-efficiency scales fundamentally better than $T_{\mathtt{BI}}$ in terms of ϵ and d.

Observation 3.17 *If $NB_4(C, \mathrm{P}) = e^{O(d)}$ and $N_\epsilon(C, \mathrm{P}) = (1/\epsilon)^{\Theta(d)}$, then*

$$T_{\mathtt{BI}}(C, \mathrm{P}, \epsilon, \delta) \;=\; \Theta\left(\frac{1}{\epsilon}\left(d \ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right),$$

$$\mathrm{E}\, T_{\mathtt{Sfoc}}(C, \mathrm{P}, \epsilon, \delta) \;=\; O\left(\frac{1}{\epsilon}\left(d + \ln\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right).$$

So we obtain a better than constant improvement in data-efficiency in these cases.

3.4.4 Examples

To make the discussion more concrete I consider a few simple examples that illustrate `Sfoc`'s advantage over `BI` and `Sbi`, both in terms of data-efficiency and computational-efficiency. These examples show that for concept spaces which are uniformly dense across local neighborhoods, `Sfoc` obtains substantial data-efficiency improvements and can even yield polynomial time learning in cases where `BI` and `Sbi` require exponential space. However, we also show that spaces with dense local neighborhoods negate these advantages.

Example: **(initials, uniform)** ***on*** $[0, 1]$

First, consider the simple task of learning an initial segment of the unit interval $[0, 1]$ under the uniform distribution. That is, consider the concept space (initials, uniform) where initials $\triangleq \{[0, x_c] : 0 \leq x_c \leq 1\}$ and domain objects are generated according to the uniform distribution over $[0, 1]$. Here each target concept $c \in$ initials is defined by an endpoint $x_c \in [0, 1]$ and classifies all points $0 \leq x \leq x_c$ as 1; see Figure 3.4. Given the simplicity of this space, it is easy to construct minimal α-covers of the entire space, as well as small α-covers of local neighborhoods.

Proposition 3.18 *For any $0 < \alpha \leq 1$*

$$N_\alpha(\text{initials}, \text{uniform}) \;=\; \left\lceil \frac{1}{2\alpha} \right\rceil$$

$$NB_4(\text{initials}, \text{uniform}) \;\leq\; 4$$

(For a global α-cover: just pick concepts 2α apart at endpoints $\alpha, 3\alpha, 5\alpha, ...,$ *etc.* For an α-cover of a local 4α-neighborhood: just pick concepts at endpoints $x_c - 3\alpha, x_c - \alpha, x_c + \alpha, x_c + 3\alpha$.)

Given these covers, we can directly determine the data-efficiency of the various learning procedures `BI`, `Sbi`, and `Sfoc`.

Observation 3.19

$$T_{\mathtt{BI}}(\text{initials}, \text{uniform}, \epsilon, \delta) \;=\; \Theta\left(\frac{1}{\epsilon}\left(\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right)$$

$$\mathrm{E}\, T_{\mathtt{Sfoc}}(\text{initials}, \text{uniform}, \epsilon, \delta) \;=\; O\left(\frac{1}{\epsilon}\left(\ln\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right)$$

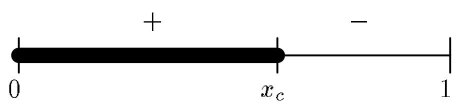

Figure 3.4: An initial segment concept c of $[0,1]$ defined by right endpoint x_c.

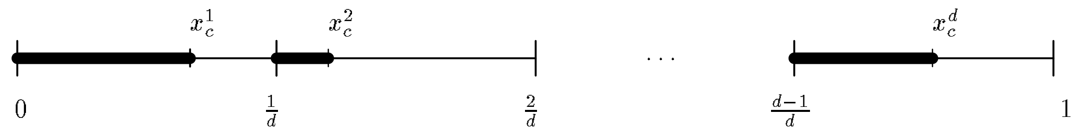

Figure 3.5: A d-π-initial segment concept c defined by right endpoints $x_c^1, ..., x_c^d$.

Notice that `Sfoc`'s data-efficiency scales fundamentally better than `BI` and `Sbi` in terms of the accuracy parameter ϵ. In fact, not only is `Sfoc` more data-efficient than the global-cover based techniques, it also has a slight computational advantage: `Sfoc` learns in $O(1)$ space, whereas `BI` and `Sbi` require $\Omega(1/\epsilon)$ space just to store the cover. Although this makes little difference on a simple space like (initials, uniform), it can become a significant factor in more complex examples.

Example: **(d-π-initials, uniform)** ***on*** **$[0,1]$**

To illustrate how these results scale-up to harder problems we consider a more complicated space (d-π-initials, uniform). This is a natural d-dimensional generalization of the simple space (initials, uniform): Here the target class d-π-initial consists of concepts defined by d independent initial segments on disjoint regions of $[0,1]$, and domain objects are generated according to the uniform distribution over $[0,1]$. Formally, we define the class of d-π-initial concepts as follows: First, partition the domain $X = [0,1]$ into d disjoint intervals $X_1 = [0, 1/d), X_2 = [1/d, 2/d), ..., X_d = [(d-1)/d, d]$. Then, define a concept $c \in d$-π-initials by independently choosing an endpoint from each of the d segments, $x_c^1 \in X_1$, ..., $x_c^d \in X_d$; so that $c(x) = 1$ if and only if $x \in [0, x_c^1] \cup ... \cup [(d-1)/d, x_c^d]$; see Figure 3.5. Thus, the entire class of d-π-initial segment concepts is generated by independently choosing d endpoints, one from each subinterval $X_1, ..., X_d$.

The fact that (d-π-initials, uniform) is a natural d-dimensional generalization of (initials, uniform) is born out by the following result characterizing the size of α-covers for this space and the size of the relative covers for local neighborhoods.

Proposition 3.20 *For any* $0 < \alpha < 1/d$,

$$\left(\frac{1}{2e\alpha}\right)^d \leq N_\alpha(d\text{-}\pi\text{-initials, uniform}) \leq \left\lceil\frac{1}{2\alpha}\right\rceil^d$$
$$NB_4(d\text{-}\pi\text{-initials, uniform}) \leq 1.4(6e)^d$$

(The upper bound for the global α-cover is obvious: for each subdomain X_i just pick $1/(2\alpha)$ endpoints $2\alpha/d$ apart at $(i-1+\alpha)/d$, $(i-1+3\alpha)/d$, $(i-1+5\alpha)/d$, ..., *etc.*; then compose cover concepts by independently choosing endpoints from each subdomain. For an α-cover of a local 4α-neighborhood: repeat the above construction for the 4α-neighborhood of each component x_c^i of c, thus α-covering each interval of length 8α with $4d$ endpoints $2\alpha/d$ apart. Then compose cover concepts by choosing endpoints from each subdomain, maintaining a total distance from c of at most 4α. See Appendix B for details.)

Given these covers, we can directly determine the data-efficiency of the various learning procedures, as before.

Observation 3.21

$$T_{\texttt{BI}}(d\text{-}\pi\text{-initials, uniform}, \epsilon, \delta) = \Theta\left(\frac{1}{\epsilon}\left(d \ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right)$$

$$\mathrm{E}\, T_{\texttt{Sfoc}}(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}, \epsilon, \delta) \quad = \quad O\left(\frac{1}{\epsilon}\left(d + \ln\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right)$$

Again, we see that `Sfoc`'s data-efficiency scales inherently better than `BI` and `Sbi`, both in terms of the dimensionality d and the accuracy parameter ϵ. In fact, `Sfoc` reduces a $d\ln(1/\epsilon)$ term to a $d + \ln\ln(1/\epsilon)$ term in this case, which is a significant improvement for small ϵ. However, the data-efficiency of all procedures remains a small order polynomial in the relevant parameters.

Interestingly, `Sfoc` also permits a polynomial time solution to this problem, even though Proposition 3.20 shows that any $\epsilon/2$-cover of $(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform})$ must be exponentially large in d; *i.e.*, $\Omega(1/\epsilon^d)$ for $\epsilon < 1/d$. This means that any global-cover based technique like `BI` or `Sbi` must take exponential time just to construct the cover! Although it is not obvious that `Sfoc` can solve this problem in polynomial time either, since the size of the $4\times$-neighborhood covers constructed in Proposition 3.20 are also exponential in d, it turns out that the set of neighborhood-cover concepts can be implicitly tested in polynomial time by independently considering each subdomain X_i (see Appendix B for details).

Proposition 3.22 `Sfoc` *can be implemented to solve* $(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}, \epsilon, \delta)$ *in time*

$$O\left(\frac{d^2}{\epsilon}\left(d\ln d + \ln\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right).$$

Thus, `Sfoc` not only reduces the data-complexity of pac-learning in this case, it also obtains an exponential reduction in computational costs.

Example: **(monomials, uniform)** ***on*** $\{0,1\}^n$

However, not every natural concept space is sufficiently uniformly dense to permit `Sfoc` to obtain these advantages. For example, consider the space (monomials, uniform) defined on $\{0,1\}^n$. This particular space has been much studied in the machine learning and computational learning theory literature [Pazzani and Sarrett, 1990; Haussler et al., 1994]. Here, the class monomials consists of concepts defined by conjunctions of (positive) boolean attributes, and domain objects are generated according to a uniform distribution over $\{0,1\}^n$. Formally, a monomial concept c is defined by a list of attribute indices $\{i_1, ..., i_k\}$, such that for a domain object $x = \langle a_1, ..., a_n\rangle \in \{0,1\}^n$, $c(x) = 1$ if and only if $a_{i_j} = 1$ on all indices specified by c. We consider the problem of learning an accurate approximation to a monomial target concept given training objects generated by a uniform distribution over $\{0,1\}^n$. It turns out that this simple space is sufficiently ill-behaved in certain neighborhoods to prevent `Sfoc` from obtaining any advantage over `Sbi` in the worst case.

To demonstrate this, first note that the space (monomials, uniform) actually has an interesting metric structure: the distance between any two monomial concepts c_1 and c_2 with respect to the uniform distribution is given by

$$d_{\mathsf{U}}(c_1, c_2) \quad = \quad 2^{-|c_1|} + 2^{-|c_2|} - 2\cdot 2^{-|c_1\cup c_2|} \tag{3.3}$$

(thinking of monomial concepts as sets of attributes), which follows from the simple relation $\mathrm{P}(A \bigtriangleup B) = \mathrm{P}A + \mathrm{P}B - 2\,\mathrm{P}(A\cap B)$. Therefore, concepts defined by few attributes (and pairs having few attributes in common) are much further apart than concepts defined by many attributes (and pairs having many attributes in common). The simple discrete structure of this space actually allows us to construct minimum size covers without too much difficulty.

Proposition 3.23 *For* $\alpha = 2^{-k}$,

$$\begin{aligned}\sum_{i=0}^{\log_2(1/\alpha)-1}\binom{n}{i} \quad \le \quad N_\alpha(\mathsf{monomials}, \mathsf{uniform}) \quad &\le \quad \sum_{i=0}^{\log_2(1/\alpha)}\binom{n}{i} \\ &\le \quad \left(\frac{en}{\log_2(1/\alpha)}\right)^{\log_2(1/\alpha)}\end{aligned}$$

The minimal α-cover simply consists of all $\mathsf{monomial}$ concepts containing $\log_2(1/\alpha)$ or fewer attributes. Also, the collection of $\mathsf{monomial}$ concepts containing $\log_2(1/\alpha) - 1$ or fewer attributes is pairwise separated by at least 2α, and therefore any α^{-}-cover of the space must contain at least this many concepts.

The problem with this space is that it is extremely dense around the smallest concept (extensionally) in the space, $c_n = \{a_1, ..., a_n\}$ (*i.e.*, around $\emptyset$). In fact, the cover of any local neighborhood of c_n has a size comparable to any efficient cover of the entire space.

Proposition 3.24 *For* $\alpha = 2^{-k} > 2^{-n-2}$, *the smallest* $\mathsf{monomial}$ $c_n = \{a_1, ..., a_n\}$ *has*

$$N_\alpha B_{4\alpha}(c_n) \quad > \quad \binom{n}{\log_2(1/\alpha) - 1} \quad = \quad \Omega\left(\frac{n}{\ln(1/\alpha)}\right)^{\ln(1/\alpha)} .$$

This prevents `Sfoc` from obtaining a significant data-efficiency advantage over `BI`.

Observation 3.25 *For* $\epsilon > e^{-n}$

$$T_{\mathtt{BI}}(\mathsf{monomials}, \mathsf{uniform}, \epsilon, \delta) \quad = \quad \Theta\left(\frac{1}{\epsilon}\left((\ln n)\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right)$$

$$\mathrm{E}\, T_{\mathtt{Sfoc}}(\mathsf{monomials}, \mathsf{uniform}, \epsilon, \delta) \quad = \quad O\left(\frac{1}{\epsilon}\left((\ln n)\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right)$$

In terms of computational-efficiency, note that the global-cover based procedures run in quasi-polynomial time since the $\epsilon/2$-cover constructed in Proposition 3.23 has size at least $\Omega(n^{\ln(1/\epsilon)})$. Unfortunately, the straightforward implementation of `Sfoc` fails to improve on this since any neighborhood cover for c_n also has quasi-polynomial size, and it is not clear that these cover-concepts can be implicitly tested in polynomial time (as for (d-π-$\mathsf{initials}, \mathsf{uniform}$) in Proposition 3.22 above). Thus, Procedure `Sfoc` does not seem to yield any significant advantages in this case beyond those already obtained by `Sbi`.

Overall, these examples show how `Sfoc` can substantially improve the data and computational efficiency of d.s. pac-learning for many natural concept spaces. However, these advantages are restricted to spaces that have a uniformly dense structure across local neighborhoods, thus permitting the early stages of a multiresolution search to have a significant effect.

3.4.5 Assessment

The previous two sections have shown how sequential learning can uniformly improve the data-efficiency of pac-learning under the d.s. model, by a significant constant factor in general via Procedure `Sbi`, and by logarithmic factors for uniformly dense concept spaces by using a multiresolution learning procedure `Sfoc`. Although the primary motivation of this work is to improve the data-efficiency of pac-learning, multiresolution learning also has the advantage of providing computationally-efficient learning procedures for many problems where the generic global-cover based approaches are inherently infeasible. In each case, these are simple generic learning procedures, based on an intuitive metric space view of d.s. learning [Benedek and Itai 1988a, 1991; Kulkarni 1991].

The next two sections consider an alternative view of d.s. learning. We explore a stronger criterion than pac-learning which sequential learners are able to achieve under the d.s. model. It turns out that the learning procedures derived for this problem can obtain additional efficiency improvements in many cases.

3.5 Learning with *certainty*

Beyond improving data-efficiency, another benefit of sequential learning is the ability to learn with *certainty* under the d.s. model, rather than just high probability. That is, there are concept spaces where a sequential learner can return an ϵ-approximation to the target with probability 1 (wp1), not just probability $1 - \delta$ for some $\delta > 0$. Interestingly, this level of reliability cannot be achieved by fixed-sample-size learning for any non-trivial space. In fact, none of the previous sequential procedures introduced in this chapter can learn

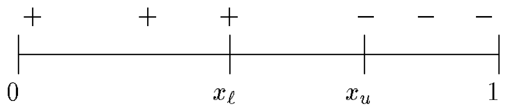

Figure 3.6: An "uncertainty interval" for (initials, uniform) generated by a set of training examples (indicated by $+$, $-$). Any consistent initial segment concept h has a right endpoint x_h lying between the largest positive and smallest negative example. *I.e.*, $x_\ell \leq x_h < x_u$.

with certainty either, as setting $\delta = 0$ prevents finite termination. However, learning with certainty can be achieved in the d.s. setting by other sequential learning strategies.

Example: To demonstrate this, consider the concept space (initials, uniform) introduced in Section 3.4.4. Here, the task is to identify an ϵ-approximation to an unknown initial segment of $[0,1]$ given uniformly distributed training examples—which amounts to locating the right endpoint of the target interval c to within a tolerance of ϵ on either side of x_c. To do this we notice that any set of training examples determines an "uncertainty interval" around x_c which contains the endpoints of all initial segment concepts consistent with the training data; see Figure 3.6. Therefore, an obvious learning strategy for this space is just to keep track of this uncertainty interval and halt as soon as it becomes smaller than 2ϵ, returning the initial segment defined by the midpoint of the final interval; see Procedure Scov(initials, uniform) in Figure 3.7. Clearly, *any* hypothesis returned by this procedure can have error at most ϵ by construction, since the furthest consistent initial segment from the midpoint concept can be at most ϵ away. Moreover, this procedure halts wp1, since the probability of observing a domain object within ϵ of both sides of x_c goes to 1 as the size of the training sample increases. Therefore, Scov is guaranteed to return an ϵ-accurate hypothesis wp1, and hence pac$(\epsilon, 0)$-learns (initials, uniform). Interestingly, Proposition 3.28 below shows that *no* fixed-sample-size procedure can achieve this level of certainty for this (or any non-trivial) space.

This example shows that there are situations under the d.s. model where a sequential procedure can learn with certainty, but any fixed-sample-size procedure must fail with some non-zero probability. This raises the question of determining the general conditions when certain learning can be obtained, and the amount of training data needed to do so. In this section we investigate the difficulty of this learning with certainty task: identifying those situations where certain learning can be achieved and analyzing the data-efficiency of proposed solutions. One indirect benefit of this study is that it yields an alternative technique for deriving data-efficient *pac*-learning procedures (which we explore in Section 3.6 below).

3.5.1 Problem

We are addressing a slightly different task than the standard pac-learning problem considered before. Now, for a given ϵ, we demand that the learner return an ϵ-approximation to the target concept with certainty, not just probability at least $1 - \delta$ for some $\delta > 0$. I refer to this task as "certainly approximately correct" (cac) learning. Clearly, cac-learning is harder than pac-learning simply because we are demanding a strictly higher level of reliability. Under the d.s. model, we specify an instance of a cac-learning problem by a concept space (C, P) and an accuracy parameter ϵ.

Definition 3.26 (Cac-learning problem) *A learner L solves the d.s. cac-learning problem $(C, \mathrm{P}, \epsilon)$ (or, "cac(ϵ)-learns (C, P)") if, given random training objects generated by* P *and labelled according to any $c \in C$, L produces a hypothesis h such that $\mathrm{P}\{h(x) \neq c(x)\} \leq \epsilon$ with probability* 1.

It is no accident that we are investigating cac-learning under the d.s. and not the d.f. model, since it is impossible to achieve cac-learning under the d.f. model with any learning strategy.

Proposition 3.27 (D.f. cac-learning impossible) *For any $0 < \epsilon < 1$, and any concept class C containing two non–mutually-exclusive concepts: Any learner L must fail to meet the cac(ϵ)-criterion for some target concept in C, for some domain distribution* P.

Procedure Scov (initials, uniform; ϵ)

INPUT: accuracy parameter ϵ.

RETURN: a hypothesis h with error at most ϵ.

PROCEDURE:

- Sequentially observe training examples $\langle x_t, c(x_t)\rangle$, $t = 1, 2, \ldots$, *etc.*, labelled by some unknown target initial concept c:
  - Let x_ℓ be the largest observed positive example; *i.e.*, the largest x_t such that $c(x_t) = 1$ ($x_\ell = 0$, if none exists).
  - Let x_u to be the smallest observed negative example; *i.e.*, the smallest x_t such that $c(x_t) = 0$ ($x_u = 1$, if none exists).
  - Observe random training examples until $x_u - x_\ell \leq 2\epsilon$; the *stopping condition.*
- Once the stopping condition is reached, return the initial segment $h = [0, x_h]$ defined by the midpoint $x_h = (x_u - x_\ell)/2$.

Figure 3.7: Procedure for learning (initials, uniform) with certainty.

Therefore, distributional assumptions are necessary to achieve cac-learning, so we restrict ourselves to the d.s. case here. Moreover, not only is it necessary to make distributional assumptions, it is also necessary to adopt a sequential rather than fixed-sample-size learning approach in order to achieve cac-learning here.

Proposition 3.28 (Fixed-sample-size cac-learning impossible) *For any* $0 < \epsilon < 1$, *and any space* (C, P) *that contains two concepts* c_1 *and* c_2 *separated by* $\epsilon < d_{\mathrm{P}}(c_1, c_2) < 1$: *Any fixed-sample-size learner* L *must fail to meet the cac*(ϵ)*-criterion for some* $c \in C$.

Therefore, to study this cac-learning problem we must restrict our attention to sequential learning procedures under the d.s. model. Despite the apparently demanding nature of the cac-criterion, it turns out that a number of surprisingly strong results can be obtained under this model.

3.5.2 Procedure

First, notice that the simple procedure Scov(initials, uniform) can easily be generalized to obtain a generic cac-learning procedure that is applicable to arbitrary concept spaces; see Procedure Scov in Figure 3.8. The idea behind Scov is straightforward: we simply observe training examples until the neighborhood of consistent concepts remaining in C is reduced to an ϵ-ball around a single hypothesis h (not necessarily in C), and return h as the final hypothesis. Obviously, any hypothesis returned by Scov is guaranteed to be an ϵ-approximation to the target concept by construction, so proving that Scov meets the cac-criterion is then a simple matter of establishing that it halts wp1. Thus, we can see that Scov correctly cac-learns any concept space (C, P) for which it is guaranteed to halt wp1 for every target concept in C. This turns out to be true of a wide range of concept spaces.

An obvious way to prove Scov meets the cac-criterion is to show that every ϵ-bad concept in the space is eventually eliminated wp1. A weaker condition that implies this is to show that every ϵ-bad concept must be eliminated from the space with some non-zero probability after a finite number of training examples.

Definition 3.29 (Reduction) *For a concept space* (C, P):

- *For* $c \in C$, *let* $P_\epsilon(C, \mathrm{P}, t, c)$ *denote the probability that* (C, P) *is reduced to a (closed)* ϵ*-ball around* c *after* t *training examples.*

Procedure `Scov` $(C, \mathrm{P}, \epsilon)$

INPUT: target concept space (C, P),
accuracy parameter ϵ.

RETURN: a hypothesis h with error at most ϵ.

PROCEDURE:

- Sequentially observe training examples $\langle x_t, c(x_t)\rangle$, $t = 1, 2, ...,$ *etc.*, labelled by some unknown target concept $c \in C$:
  - Halt when there exists a single concept h (not necessarily in C) within ϵ of each consistent c remaining in C.
- Return h.

Figure 3.8: Procedure `Scov`

- *Let* $P_\epsilon(C, \mathrm{P}, t)$ *be the minimum such probability over possible target concepts* $c \in C$*; i.e.,* $P_\epsilon(C, \mathrm{P}, t) \triangleq \inf_{c\in C} P_\epsilon(C, \mathrm{P}, t, c)$.
- *Then let* $R_\epsilon(C, \mathrm{P})$ *be the least* t *such that* $P_\epsilon(C, \mathrm{P}, t) > 0$*. (If no such* t *exists, we define* $R_\epsilon(C, \mathrm{P}) = \infty$*.) Thus,* $R_\epsilon(C, \mathrm{P}) < \infty$ *means that there exists some* $\gamma > 0$ *such that for all* $c \in C$*, with probability at least* γ*,* (C, P) *is reduced to an* ϵ*-ball around* c *after* $R_\epsilon(C, \mathrm{P})$ *training examples.*
- $R_\epsilon(C, \mathrm{P})$ *is called the* ϵ-reduction number *of* (C, P)*. Any concept space* (C, P) *with a finite* ϵ*-reduction number is said to be* ϵ-reducible.

It is not hard to show that `Scov` correctly cac-learns any concept space that is ϵ-reducible at the desired error ϵ.

Proposition 3.30 (Correctness) *For any* $\epsilon > 0$*, and any concept space* (C, P) *with* $R_\epsilon(C, \mathrm{P}) < \infty$*:* `Scov` *halts wp1 and meets the cac(*ϵ*)-criterion for every target* $c \in C$*.*

This proves that `Scov` cac-learns a broad class of concept spaces. In fact, most concept classes encountered in practice are ϵ-reducible, and hence cac-learnable by `Scov`. For example, this is true of any concept space (C, P) where C has finite VCdimension.

Proposition 3.31

1. $\mathrm{VC}(C) < \infty$ *implies* $R_\epsilon(C, \mathrm{P}) < \infty$ *for all* $\epsilon > 0$.

   (In fact, $\mathrm{VC}(C) < \infty$ *if and only if for all* $\epsilon > 0$ *there exists an* $r < \infty$ *and* $p > 0$ *such that* $P_\epsilon(C, \mathrm{P}, r) \geq p$ *for all domain distributions* P*.)*

2. *There are spaces* (C, P) *for which* $\mathrm{VC}(C) = \infty$ *and yet* $R_\epsilon(C, \mathrm{P}) < \infty$.

However, `Scov` does not cac-learn every concept space which is *pac*-learnable in the d.s. setting. That is, there are concept spaces (C, P) that are finitely coverable for all $\epsilon > 0$, but not ϵ-reducible for any $\epsilon > 0$.

Proposition 3.32

1. *If* $R_\epsilon(C, \mathrm{P}) < \infty$ *for all* $\epsilon > 0$*, then* $N_\epsilon(C, \mathrm{P}) < \infty$ *for all* $\epsilon > 0$.
2. *There are concept spaces* (C, P) *for which* $R_\epsilon(C, \mathrm{P}) = \infty$ *and yet* $N_\epsilon(C, \mathrm{P}) < \infty$.

An example of such a space is $(\{\text{finite sets}\} \cup \{[0,1]\}, \textsf{uniform})$ defined on $[0,1]$. This space is not ϵ-reducible at any scale $\epsilon < 1$ since, as we saw in Section 3.2, if $[0,1]$ is the target concept then any finite training sample leaves consistent concepts a distance 1 from $[0,1]$. On the other hand, we also saw that $\{\varnothing, [0,1]\}$ is an ϵ-cover of this space for any $\epsilon \geq 0$, which means that any of the cover-based learning procedures discussed in Sections 3.2–3.4 can successfully pac-learn this space.

However, most natural concept spaces are finitely reducible, as suggested by Proposition 3.31—only pathological examples appear not to be. This means `Scov` is a fairly general cac-learning procedure that is applicable to most concept spaces normally encountered in practice. The question of whether non-ϵ-reducible concept spaces are nevertheless cac-learnable is addressed in Section 3.5.4 below.

3.5.3 Efficiency

Given the correctness of `Scov`, the main issue is determining its efficiency. As usual, we measure the data-efficiency of a sequential learner by its worst case expected sample size over all target concepts in C. Here it is not hard to derive a simple upper bound on `Scov`'s expected sample size in terms of the reduction parameters R_ϵ and P_ϵ.

Proposition 3.33 (Data-efficiency) *For any space (C, P) with $R_\epsilon = R_\epsilon(C, \mathrm{P}) < \infty$:*

$$\mathrm{E}\, T_{\texttt{Scov}}(C, \mathrm{P}, \epsilon) \;\leq\; \frac{R_\epsilon}{P_\epsilon(C, \mathrm{P}, R_\epsilon)}. \tag{3.4}$$

Unfortunately, this bound is quite loose, and does not generally give an accurate indication of `Scov`'s true data-efficiency in many cases. Developing a *tight* characterization of `Scov`'s data-efficiency in terms of the reduction parameters, or any other simple measure of concept space complexity appears to be quite difficult.

However, despite the difficulty of obtaining a precise quantitative characterization of `Scov`'s data-efficiency, it turns out that we can show `Scov` has the following rather remarkable optimality property.

Theorem 3.34 (Optimality) *For any ϵ, and any (well behaved[7]) concept space (C, P) with $R_\epsilon(c, \mathrm{P}) < \infty$:* `Scov` *meets the cac($\epsilon$)-criterion with* optimal *expected training sample size for any target concept $c \in C$.*

Therefore, even though `Scov` is an incredibly simple-minded technique, it is the optimum possible procedure for cac-learning any ϵ-reducible concept space (C, P). That is, `Scov` minimizes the expected number of training examples needed to cac-learn any target concept in the space. This means `Scov`'s data-efficiency cannot be improved upon, even for a single target concept in C, if the certainty guarantees are to be maintained.

3.5.4 Complexity

Since `Scov` cac-learns with optimum data-efficiency for ϵ-reducible spaces, it follows that the inherent data-complexity of cac-learning these spaces coincides with `Scov`'s worst case data-efficiency. That is, the minimum average number of training examples *any* learner requires to successfully meet the cac-criterion simply corresponds to `Scov`'s expected training sample size for the problem. I have yet to derive a tight lower bound on `Scov`'s expected data-efficiency in terms of the ϵ-reduction parameters R_ϵ and P_ϵ. However, it can be shown that finite reducibility is not only sufficient for `Scov` to cac-learn, it is also necessary for any learning procedure to be able to cac-learn with bounded data-efficiency.

Theorem 3.35 (Cac-learnability) *For a (well behaved) space (C, P): If $R_\epsilon(C, \mathrm{P}) = \infty$ for some $\epsilon > 0$, then no learner L can cac(ϵ)-learn (C, P) with a bounded expected training sample size for every $\epsilon > 0$.*

For example, the space $(\{\text{finite sets}\} \cup \{[0,1]\}, \textsf{uniform})$ cannot be cac-learned by any learning procedure, since it is not ϵ-reducible for any $\epsilon < 1$. Therefore, in addition to being an optimal cac-learning procedure, `Scov` is also a universal cac-learner in the sense that `Scov` successfully cac-learns any concept space for which this is possible in principle—doing so optimally.

[7] This result assumes the space (C, P) satisfies a certain separability condition, which is satisfied by all concept spaces normally encountered in practice. See Appendix B for details.

Given the strength of these results, it seems that achieving cac-learning in practice amounts to little more than finding an efficient implementation of Procedure $\mathtt{Scov}$. Given its optimality, this will always yield a data-efficient learning procedure. However, $\mathtt{Scov}$ may not be efficiently realizable from a computational standpoint, so heuristic implementations will generally have to be considered in practice.

3.5.5 Examples

As mentioned, the crude upper bound derived in Proposition 3.33 does not give a precise characterization of $\mathtt{Scov}$'s true data-efficiency in real applications. To obtain a more accurate picture of $\mathtt{Scov}$'s real data-efficiency, I consider a few simple case studies where we can precisely analyze the expected training sample size of $\mathtt{Scov}$. These case studies show that $\mathtt{Scov}$ is an extremely data-efficient learning procedure in many practical circumstances (which is not entirely surprising given its optimality).

***Example:* (initials, uniform) *on* $[0,1]$**

First we consider the concept space $(\mathsf{initials}, \mathsf{uniform})$ introduced in Section 3.4.4. Here we have already seen that $\mathtt{Scov}$ can be easily implemented for this space (Figure 3.7). In fact, this implementation is also computationally-efficient as it requires only constant time per training example. Moreover, $\mathtt{Scov}$ is extremely *data*-efficient for this problem as well.

The crude bound from Proposition 3.33 can be applied to this space by noticing that $(\mathsf{initials}, \mathsf{uniform})$ can be reduced to an ϵ-ball around a target concept c by a observing domain object in each subinterval $[x_c-\epsilon, x_c]$ and $(x_c, x_c+\epsilon]$. Given this fact, we can determine that $R_\epsilon(\mathsf{initials}, \mathsf{uniform}) = 2$ and $P_\epsilon(\mathsf{initials}, \mathsf{uniform}, 2) = 1/(2\epsilon^2)$; and hence derive $\mathrm{E}\, T_{\mathtt{Scov}}(\mathsf{initials}, \mathsf{uniform}, \epsilon) \le 1/\epsilon^2$ from Proposition 3.33. Clearly, however, this is a loose bound. A more precise characterization of $\mathtt{Scov}$'s data-efficiency can be obtained by a direct analysis. In fact, we can derive an exact characterization of $\mathtt{Scov}$'s stopping time for this simple space.

Proposition 3.36 *For $0 < \epsilon < 1/2$, and any $c \in \mathsf{initials}$ with $x_c \in [2\epsilon, 1-2\epsilon]$,*

$$T_{\mathtt{Scov}}(\mathsf{initials}, \mathsf{uniform}, \epsilon) \quad \sim \quad \mathsf{negative\text{-}binomial}(p = 2\epsilon,\ k = 2),$$

which gives

$$\mathrm{E}\, T_{\mathtt{Scov}}(\mathsf{initials}, \mathsf{uniform}, \epsilon) \quad \le \quad \frac{1}{\epsilon}$$

(since $\mathtt{Scov}$ stops even faster for concepts with endpoints nearer to 0 or 1).

This is extremely efficient learning performance: $\mathtt{Scov}$ only requires an average of $1/\epsilon$ training examples to return an ϵ-accurate hypothesis with certainty for the space $(\mathsf{initials}, \mathsf{uniform})$.

***Example:* (d-π-initials, uniform) *on* $[0,1]$**

Similar results can be obtained for the space $(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform})$. For this more complicated space we must extend the previous version of $\mathtt{Scov}$ to handle d independent $\mathsf{initial}$ segments defined on d disjoint subintervals of $[0,1]$. This can be done simply by keeping uncertainty intervals around each of the d unknown endpoints, and halting as soon as the sum of interval-widths shrinks below 2ϵ; see Procedure $\mathtt{Scov}(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform})$ in Figure 3.9. The correctness of this procedure is fairly obvious, since any hypothesis it returns is guaranteed to be ϵ-accurate by construction, and it clearly terminates wp1. As before, not only is $\mathtt{Scov}$ easy to implement for this space it is also computationally-efficient as well, requiring at most $O(d)$ time per training example. Here again, $\mathtt{Scov}$ also turns out to be extremely data-efficient.

The crude bound from Proposition 3.33 can be applied to this space by noticing that $(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform})$ is reduced to an ϵ-ball around a target concept c by observing a domain object in each of the "ϵ-brackets" $[x_c^i - \epsilon, x_c^i]$ and $(x_c^i, x_c^i + \epsilon]$ surrounding c's d independent endpoints $x_c^1, ..., x_c^d$. This ensures that the sum of largest brackets from each pair has length at most 2ϵ. Given this observation we can determine that $R_\epsilon(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}) = 2d$ and $P_\epsilon(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}, 2d) < (2d)!\,\epsilon^{2d}$; and hence from Proposition 3.33 derive $\mathrm{E}\, T_{\mathtt{Scov}}(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}, \epsilon) < 2d/[(2d)!\,\epsilon^{2d}]$. However, as before, this bound is loose. A far better characterization of $\mathtt{Scov}$'s data-efficiency can be determined by a direct analysis of the problem. In fact, we

Procedure Scov (d-π-initials, uniform; ϵ)

INPUT: accuracy parameter ϵ.

RETURN: a hypothesis h with accuracy at least $1-\epsilon$.

PROCEDURE:

- Sequentially observe training examples $\langle x_t, c(x_t)\rangle$, $t = 1, 2, ...,$ *etc.*, labelled by some unknown target d-π-initial concept c:
  - Let $lb[i]$ be the largest positive example observed in subdomain X_i ($lb[i] = 0$, if none exists).
  - Let $ub[i]$ be the smallest negative example observed in subdomain X_i ($ub[i] = 1$, if none exists).
  - Observe random training examples until $\sum_{i=1}^{d}(ub[i] - lb[i]) \leq 2\epsilon$; the *stopping condition*.
- Once the stopping condition is reached, return the d-π-initial concept h defined by the midpoints in each subdomain; *i.e.*, $x_h^i = (ub[i] + lb[i])/2$ for $i = 1, ..., d$.

Figure 3.9: Procedure for learning (d-π-initials, uniform) with certainty.

can scale up the previous analysis for (initials, uniform) to obtain an exact characterization of Scov's stopping time for this more complicated space.

Theorem 3.37 *For any* $0 < \epsilon < 1/(4d)$, *and any target concept* $c \in$ d-π-initials *defined by endpoints* $x_c^1, ..., x_c^d$, *where* $x_c^i \in [\,(i-1)/d + 2\epsilon,\ i/d - 2\epsilon\,]$ *for* $i = 1, ..., d$,

$$T_{\text{Scov}}(d\text{-}\pi\text{-initials}, \text{uniform}, \epsilon) \ \sim\ \text{negative-binomial}(p = 2\epsilon,\ k = 2d),$$

which gives

$$\mathrm{E}\, T_{\text{Scov}}(d\text{-}\pi\text{-initials}, \text{uniform}, \epsilon) \ \leq\ \frac{d}{\epsilon}$$

(since Scov *stops even faster for concepts nearer the subdomain boundaries).*

Again, this shows that Scov obtains extremely efficient performance, even while learning with certainty.

Example: **(monomials, uniform)** ***on*** $\{0,1\}^n$

Although usually data-efficient, it is not always easy to find a computationally-efficient implementation of Scov. An example of this is the space (monomials, uniform) defined on $\{0,1\}^n$ which was introduced in Section 3.4.4. Although Scov is still data-efficient for this space, any straightforward implementation of Scov for (monomials, uniform) requires at least quasi-polynomial time to produce a final hypothesis in the worst case. We demonstrate this below by considering what a straightforward implementation of Scov would look like for this space.

Recall that (monomials, uniform) has an uneven metric structure where the (extensionally) large monomial concepts (defined by few attributes) are widely spaced, but the small monomial concepts (defined by many attributes) are tightly clustered around $\varnothing$. Therefore, the difficulty of reducing this space to an ϵ-ball around some target concept depends strongly on the size of the target concept.

Proposition 3.38 *For distinct* monomial *concepts* c_1 *and* c_2 *(assuming* $\epsilon = 2^{-k}$*):*

1. *If* $|c_1| \leq \log_2(1/\epsilon) - 1$, *then* $d_U(c_1, c_2) \geq \epsilon$ *for all* $c_2 \neq c_1$.
2. *If* $|c_1| = \log_2(1/\epsilon)$, *then* $d_U(c_1, c_2) < \epsilon$ *iff* $c_1 \subset c_2$, *but* $d_U(c_1, c_2) \geq \epsilon/2$ *for all* $c_2 \neq c_1$.
3. *If* $|c_1| \geq \log_2(1/\epsilon) + 1$, *then* $d_U(c_1, c_2) < \epsilon$ *if* $|c_2| \geq \log_2(1/\epsilon) + 1$.

Procedure Scov (monomials, uniform; ϵ)

INPUT: accuracy parameter ϵ.

RETURN: a hypothesis h with accuracy at least $1 - \epsilon$.

PROCEDURE:

- Let $h_0 = \{a_1, ..., a_n\}$ be the initial hypothesis; *i.e.*, the monomial concept that conjoins every attribute.
- Sequentially observe training examples $\langle x_t, c(x_t)\rangle$, $t = 1, 2, ...,$ *etc.*, labelled by some unknown target monomial c:
  - If x_t is a positive example (*i.e.*, $c(x_t) = 1$), then update the current hypothesis to include this example; *i.e.*, let $h_i = h_{i-1} \cap \{\text{positive attributes in } x_t\}$.
  - If no monomial defined by $\log_2(1/\epsilon)$ or more attributes is consistent with the observed training examples (the stopping condition), then *halt* and return h_i.
- Repeat until some hypothesis h_i is returned.

Figure 3.10: Procedure for learning (monomials, uniform) with certainty.

That is, for large target concepts, defined by fewer than $\log_2(1/\epsilon)$ attributes, the only way to achieve ϵ^--reduction is to eliminate every other monomial concept in the space. On the other hand, for small targets, defined by more than $\log_2(1/\epsilon)$ attributes, it suffices to eliminate only the large concepts from the space.

Given these observations, we can derive a straightforward implementation of Scov: Keep track of the smallest monomial concept consistent with the training examples, and continue to observe training examples until every large concept (defined by $\log_2(1/\epsilon)$ or fewer attributes) has been eliminated from the space; see Procedure Scov(monomials, uniform) in Figure 3.10. This procedure works because the smallest monomial concept consistent with the training examples is guaranteed to be unique [Pazzani and Sarrett, 1990; Haussler, 1988]. Also, by Proposition 3.38, any large monomial concept defined by $\log_2(1/\epsilon)$ or fewer attributes is guaranteed to be more than ϵ away from the smallest consistent monomial concept, so we cannot halt before all large concepts have been eliminated.

Unfortunately, although this is a simple procedure, it does not quite run in polynomial time. The problem is that there are too many large concepts in the space, and there is no obvious way to test them all for consistency, short of enumeration. Specifically, there are more than $\binom{n}{\log_2(1/\epsilon)}$ monomial concepts defined by $\log_2(1/\epsilon)$ or fewer attributes, which means that any implementation of Scov is likely to require at least quasi-polynomial time since this is how many large monomial concepts must be considered. Despite these computational difficulties however, Scov remains a very *data*-efficient learning procedure for this space.

Theorem 3.39 *For $\epsilon < 1/2$ and any target concept $c \in$ monomials$\{0,1\}^n$,*

$$\mathrm{E}\, T_{\mathsf{Scov}}(\text{monomials, uniform}, \epsilon) \;\le\; \frac{2}{\epsilon}\left(\ln\frac{1}{\epsilon}\right)\log_2(en).$$

Overall, these case studies show how Procedure Scov can be easily implemented for various concept spaces once the metric structure is understood, but finding a polynomial time implementation can be difficult in general. Regardless of the computational difficulties however, these case studies demonstrate that Scov is an extremely data-efficient learning procedure for a wide variety of concept spaces.

3.5.6 Assessment

This section showed how it is possible to learn with certainty under the d.s. model, following a necessarily sequential learning strategy. The problem of learning with certainty turns out to be solvable for any concept space that is ϵ-reducible at the desired error level ϵ. This is a condition that is satisfied by all concept spaces normally encountered in practice; *e.g.*, whenever $\text{VC}(C) < \infty$ (but not necessarily whenever (C, P) is finitely-coverable). The main result of this section was to identify a universal learning procedure, `Scov`, that cac-learns with optimal expected data-efficiency for any cac-learnable concept space. This procedure is easy to implement for simple spaces, and appears to be extremely data-efficient in practice.

Analogous to the previous *pac*-learning theory, it would be useful to obtain a tight characterization of the inherent data-complexity of cac-learning based on some simple measure of the complexity of the concept space. However, this turned out to be difficult to achieve. One idea is to exploit the fact that we know `Scov` has optimal data-efficiency, and reduce the problem to determining bounds on `Scov`'s expected training sample size. Thus, the challenge is to identify a simple structural parameter of concept spaces that determines tight (linear) bounds on `Scov`'s data-efficiency. The ϵ-reduction parameters R_ϵ and P_ϵ are obvious candidates for such a measure, since a concept space is cac-learnable if and only if it is ϵ-reducible. However, the analyses of `Scov`'s data-efficiency in terms of R_ϵ and P_ϵ appear to be too loose to be useful in practice. Improving these bounds remains an important open problem for future research. Perhaps a tight characterization could be based on some other measure of concept space complexity; for example, effective VCdimension, or some other measure.[8]

Notice however that such a characterization is not as important here as it was for pac-learning: here we already have an optimal procedure, `Scov`, and we know that we cannot improve upon `Scov`'s efficiency, regardless of how carefully we can predict its performance *a priori*. Therefore, an improved analysis in this case can only help predict how well `Scov` will perform on a particular problem, but it cannot lead to an improved procedure. Anyways, we saw in Section 3.5.5 that it is usually possible to obtain a reasonable characterization of `Scov`'s data-efficiency in specific case studies by a special case analysis.

3.6 Applications of *certain* to pac learning

The previous section studied the problem of learning with certainty, more or less as an aside to our investigation of pac-learning. However, here we observe that the optimally data-efficient cac-learning procedure `Scov` can also be applied to *pac*-learning problems. This is because producing an ϵ-accurate hypothesis with certainty automatically satisfies the pac(ϵ, δ)-criterion for any $\delta > 0$. Surprisingly, `Scov` turns out to be far more data-efficient than any of the standard pac-learning procedures (`BI`, `Sbi`, and `Sfoc`) on many natural problems, even though `Scov` attains a strictly higher level of reliability. Although counterintuitive, this suggests that `Scov` might form the basis for more data-efficient pac-learning procedures than previous approaches.

Of course, from Proposition 3.32 we know that not every pac-learnable concept space is cac-learnable. A simple example of this is the space $(\{\text{finite sets}\} \cup \{[0,1]\}, \textsf{uniform})$ on $[0,1]$, which we have seen is trivially pac-learnable, but cannot be cac-learned for any $\epsilon < 1$. Therefore, Procedure `Scov` cannot be a fully general pac-learning procedure like the previous cover-based techniques, `BI`, `Sbi`, and `Sfoc`. However, we do know that `Scov` can cac-learn any concept space (C, P) for which $\text{VC}(C) < \infty$ by Proposition 3.31, and hence can pac-learn most concept spaces normally encountered in practice.

The fact that `Scov` is far more data-efficient than the cover-based approaches in some cases raises the question of determining the range of concept spaces where `Scov` obtains an advantage, and quantifying `Scov`'s advantage when it does. Unfortunately, this proves to be difficult without a tight characterization of `Scov`'s general data-efficiency. However, we can still investigate the various ways that `Scov` might be applied to improve the data-efficiency of existing pac-learning procedures.

3.6.1 Direct application

The most obvious idea here is to *directly* apply `Scov` to pac-learning problems, exploiting the fact that most natural pac-learnable concept spaces are also cac-learnable. Clearly, if `Scov` cac-learns a concept space then it pac-learns it. Surprisingly, `Scov` is far more data-efficient than standard pac-learning approaches in many natural cases, even though it solves a harder learning problem.

[8]Notice that $\text{VC}(C)$ and $N_\epsilon(C, \text{P})$ cannot be appropriate candidates for such a measure, since neither coincides with cac-learnability; *cf.* Propositions 3.31 and 3.32.

To illustrate this, consider the concept space (initials, uniform) on $[0,1]$. We can directly compare `Scov` with the cover-based learning procedures, `BI`, `Sbi`, and `Sfoc`, on this space (using the covers constructed in Proposition 3.18).

Observation 3.40 *For the pac-learning problem* (initials, uniform, $\epsilon = 0.01, \delta = 0.05$)*:*

$$\begin{aligned} T_{\texttt{BI}} &= \Theta\left(\frac{1}{\epsilon}\left(\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta}\right)\right) &= 24{,}323 \\ \mathrm{E}\,T_{\texttt{Sfoc}} &= O\left(\frac{1}{\epsilon}\left(\ln\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta}\right)\right) &\leq 9{,}528 \\ \mathrm{E}\,T_{\texttt{Scov}} &\leq \frac{1}{\epsilon} &= 100 \end{aligned}$$

Notice that `Scov`'s data-efficiency scales fundamentally better than standard pac-learning procedures. In fact, `Scov` requires orders of magnitude fewer training examples in this case, even though it returns an ϵ-accurate hypothesis with certainty.

Similar results are also obtained for the more complicated space (d-π-initials, uniform) on $[0,1]$. Here, using the α-cover size $(2\alpha)^{-d}$ from Proposition 3.20, we can again see that `Scov` uses many times fewer training examples than any of the cover-based learning procedures in addition to achieving better scaling behavior.

Observation 3.41 *For the problem* (d-π-initials, uniform, $\epsilon = 0.01, \delta = 0.05$) *with* $d = 10$*:* [9]

$$\begin{aligned} T_{\texttt{BI}} &= \Theta\left(\frac{1}{\epsilon}\left(d\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta}\right)\right) &= 156{,}952 \\ \mathrm{E}\,T_{\texttt{Sfoc}} &= O\left(\frac{1}{\epsilon}\left(d+\ln\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta}\right)\right) &\leq 44{,}552 \\ \mathrm{E}\,T_{\texttt{Scov}} &\leq \frac{d}{\epsilon} &= 1{,}000 \end{aligned}$$

Finally, for the space (monomials, uniform) on $\{0,1\}^n$, we again see that `Scov` is substantially more data-efficient than the cover-based learning approaches (using the covers constructed in Propositions 3.23 and 3.24).

Observation 3.42 *For* (monomials, uniform, $\epsilon = 2^{-7}, \delta = 0.05$) *on* $\{0,1\}^n$*, with* $n = 10$*:* [10]

$$\begin{aligned} T_{\texttt{BI}} &= \Theta\left(\frac{1}{\epsilon}\left((\ln n)\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta}\right)\right) &= 24{,}907 \\ \mathrm{E}\,T_{\texttt{Scov}} &\leq O\left(\frac{1}{\epsilon}(\ln n)\ln\frac{1}{\epsilon}\right) &\leq 958 \end{aligned}$$

Although these results are anecdotal, they demonstrate how `Scov` achieves *far* better data-efficiency in a variety of cases. This seems surprising since, at first glance, one would expect `Scov` to be less data-efficient than the cover-based procedures given its greater reliability. However, we see that `Scov` can actually use orders of magnitude less training data in many cases, even while achieving these higher reliability levels.

The primary reason for this overwhelming advantage appears to be the fact that `Scov` is not based on *estimation*. That is, all of the cover-based learning techniques use training examples to estimate the error rates of various hypotheses to identify an accurate candidate. `Scov`, on the other hand, avoids estimation

[9] Interestingly, the straightforward implementations of `BI` and `Sbi` require exponential time for this problem, while `Sfoc` and `Scov` can be easily implemented to run in polynomial (expected) time.

[10] The obvious implementations of procedures `BI`, `Sbi`, `Sfoc`, and `Scov` all require quasi-polynomial time for this problem. However, `Scov` can be easily modified to run in polynomial time for this problem if we give up the guarantees of certainty.

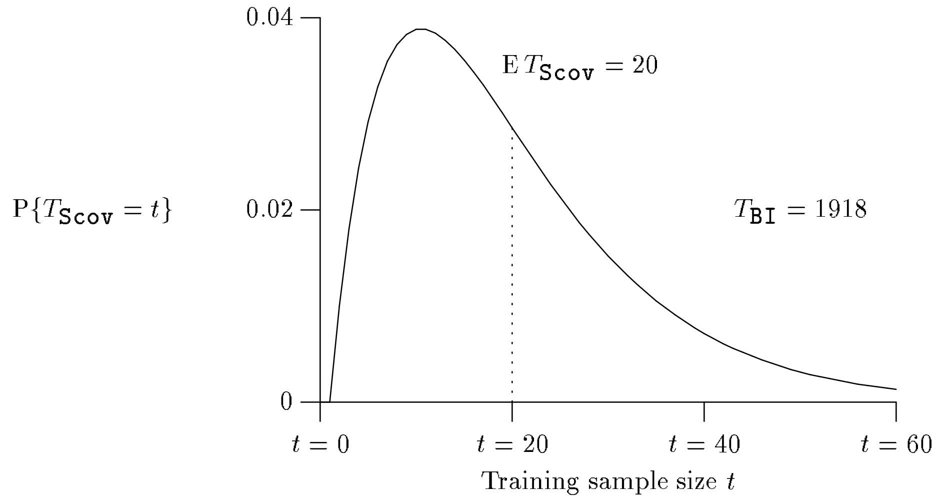

Figure 3.11: Comparing $T_{\texttt{Scov}}$ and $T_{\texttt{BI}}$ on $(\mathsf{initials}, \mathsf{uniform}, \epsilon = 0.05, \delta = 0.05)$; noting that $T_{\texttt{Scov}}(\mathsf{initials}, \mathsf{uniform}, \epsilon, \delta) \sim \mathsf{negative\text{-}binomial}(p = 2\epsilon, k = 2)$.

altogether. There is an intuitive sense in which estimation seems inherently inefficient under the d.s. model: if the entire metric structure of the space is known *a priori*, why use training data to estimate inter-concept distances? Scov appears to gain an inherent data-efficiency advantage over the cover-based learning procedures by not using training examples to gain information that is already known *a priori.*[11] Of course, we also know that not every cac-learnable concept space can also be pac-learned. Therefore, Scov is not applicable to every pac-learnable concept space, and hence some form of estimation is necessary in general; *i.e.*, whenever the space is finitely ϵ-coverable but not ϵ-reducible. Of course, non–ϵ-reducible spaces are not typical in practice, and this leaves open the question of whether estimation is necessary for d.s. pac-learning under any practical circumstances.

3.6.2 Tail truncation

Interestingly, Scov's performance can still be slightly improved for pac-learning problems. To see this, note that Scov not only observes a small expected number of training examples, it is also unlikely that Scov ever observes a large training sample.

To see this, consider the space $(\mathsf{initials}, \mathsf{uniform})$ on $[0, 1]$. From Proposition 3.36 we know that $T_{\texttt{Scov}}$ has a negative-binomial distribution with parameters $p = 2\epsilon$ and $k = 2d$, and Figure 3.11 shows that this distribution has most of its mass concentrated below the mean $1/\epsilon$. This means that Scov observes large training samples only with exceedingly small probability. In fact, for this example there is only a minuscule probability that Scov ever observes more training examples than BI.

Proposition 3.43 *For* $(\mathsf{initials}, \mathsf{uniform})$*:* $\mathrm{P}_{X^\infty}\{T_{\texttt{Scov}} > t\} < e^{-t\epsilon\left(1-\frac{1}{t\epsilon}\right)^2}$ *for* $t \geq 1/\epsilon$*. Thus, for* $\epsilon \leq e^{-3/2}$, $\delta \leq e^{-1/2}$*, and using the cover constructed in Proposition 3.18, we get*

$$\mathrm{P}_{X^\infty}\{T_{\texttt{Scov}} > T_{\texttt{BI}}\} < (e\epsilon\delta)^{30}.$$

The same is also true for the more complicated space $(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform})$.

Proposition 3.44 *For* $(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform})$*:* $\mathrm{P}_{X^\infty}\{T_{\texttt{Scov}} > t\} < e^{-t\epsilon\left(1-\frac{d}{t\epsilon}\right)^2}$ *for* $t \geq d/\epsilon$*. Thus, for* $\epsilon \leq e^{-3/2}$, $\delta \leq e^{-d/2}$*, and using the cover constructed in Proposition 3.20, we get*

$$\mathrm{P}_{X^\infty}\{T_{\texttt{Scov}} > T_{\texttt{BI}}\} < (e\epsilon)^{30d}\delta^{30}.$$

[11]Note that estimation also prevents these cover-based procedures from achieving certainty within any finite number of training examples.

Procedure `Scut` $(C, \mathrm{P}, \epsilon, \delta)$

- Let δ-tail$_{\texttt{Scov}}$ be a fixed training sample size such that $\mathrm{P}_{X^\infty}\{T_{\texttt{Scov}} > \delta\text{-tail}_{\texttt{Scov}}\} < \delta$ for every $c \in C$.
- Run Procedure `Scov` and return any hypothesis it produces.
- If `Scov` has not terminated within $t = \delta$-tail$_{\texttt{Scov}}$ training examples, then halt and return an arbitrary hypothesis.

Figure 3.12: Procedure `Scut`

This shows that even though there is no strict upper bound on the number of training examples `Scov` might observe, the probability that `Scov` observes a large training sample is exceedingly small; in fact, `Scov` almost never observes more training examples than the fixed-sample-size procedure `BI`. However, by insisting that the learner stop within some maximum number of training examples we obtain another idea for data-efficient pac-learning: instead of letting `Scov` run until termination, we only let `Scov` run until the "δ-tail" of its sample size distribution.

Definition 3.45 (δ-tail) *The* δ-tail *of a sequential learning procedure L is the smallest training sample size, δ-tail$_L$, such that $\mathrm{P}_{X^\infty}\{T_L > \delta\text{-tail}_L\} < \delta$ for any target concept $c \in C$. That is, L is guaranteed to halt within δ-tail$_L$ training examples, with probability at least $1-\delta$ for any $c \in C$.*

Thus, we can obtain pac-learning by running `Scov` as is, returning any hypothesis it produces, until we reach the δ-tail of its sample size distribution—at which point we halt and return an arbitrary hypothesis. I refer to this truncated version of `Scov` as Procedure `Scut` (Figure 3.12). Obviously `Scut` is a correct pac-learner whenever the concept space is ϵ-reducible at the desired scale ϵ: if `Scov` halts it is guaranteed to return an ϵ-approximation by construction, and `Scut` truncates `Scov` with probability at most δ.

The benefits of this truncation procedure are twofold: First, `Scut` improves `Scov`'s expected stopping time, since it always halts at or before before `Scov`. Second, `Scut` provides a strict upper bound on training sample size. Thus, if one cannot tolerate even a small probability that the training sample size exceeds some pre-set bound, we can use `Scut` instead of `Scov` (and obtain a slight data-efficiency improvement in the bargain). Of course, in so doing we must give up the guarantees of certainty and settle instead for the weaker pac-criterion.

To illustrate the advantages of `Scut` we briefly reconsider the previous case studies. From Propositions 3.43 and 3.44 we can determine an upper bound on `Scov`'s δ-tail for the concept spaces $(\mathsf{initials}, \mathsf{uniform})$ and $(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform})$, and hence identify appropriate sample size cutoffs for Procedure `Scut`; see Figure 3.13.

Proposition 3.46

$$T_{\texttt{Scut}}(\mathsf{initials}, \mathsf{uniform}, \epsilon, \delta) \;\le\; \frac{2}{\epsilon}\left(1 + \frac{1}{2}\ln\frac{1}{\delta}\right)$$

$$T_{\texttt{Scut}}(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}, \epsilon, \delta) \;\le\; \frac{2}{\epsilon}\left(d + \frac{1}{2}\ln\frac{1}{\delta}\right)$$

Thus, not only does `Scut` pac-learn these spaces with a smaller expected sample size than `Scov` (and hence smaller than `BI`, `Sbi`, and `Sfoc`, *cf.* Observations 3.40 3.41, and 3.42), `Scut` also pac-learns with a small upper bound on total training sample size. In fact, `Scut` observes a maximum number of training examples that is an order of magnitude smaller than `BI` for these problems.

Observation 3.47 *Using the $\epsilon/2$-covers constructed in Propositions 3.18 and 3.20,*

$$T_{\texttt{BI}}(\mathsf{initials}, \mathsf{uniform}, \epsilon, del) \;=\; \frac{32}{\epsilon}\left(\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)$$

$$T_{\texttt{BI}}(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}, \epsilon, del) \;=\; \frac{32}{\epsilon}\left(d\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)$$

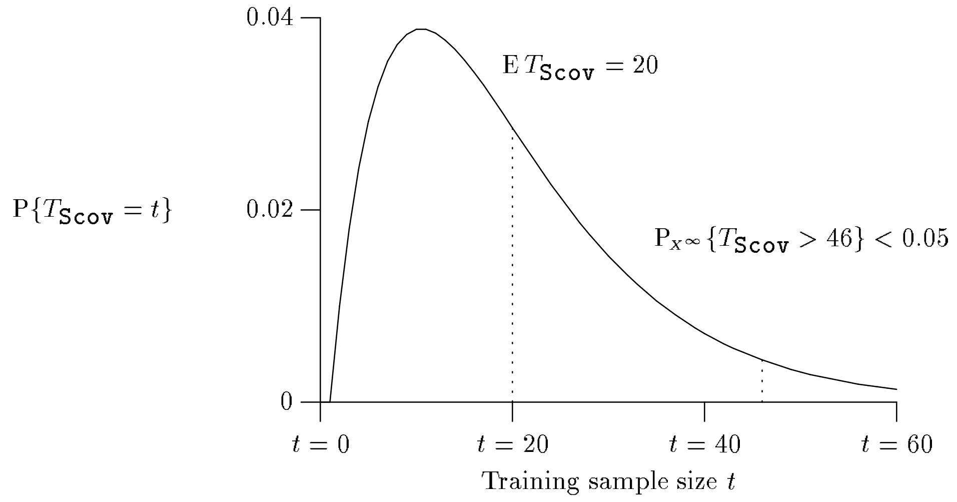

Figure 3.13: Illustration of a δ-tail cutoff. Here we see that `Scov`'s 0.05-tail for the cac-learning problem $(\mathsf{initials}, \mathsf{uniform}, \epsilon = 0.05)$ is 46. *I.e.*, Procedure `Scut` can solve this problem by truncating `Scov` after 46 training examples.

Thus, the upper bound on `Scut`'s training sample size is over 16 times smaller than `BI`'s fixed-sample-size for these problems. Interestingly, this means that these concept spaces can be pac-learned with a much smaller fixed-sample-size than previous bounds. That is, analyzing the sample size distribution of a sequential learning procedure like `Scov` actually provides a new technique for deriving improved fixed-sample-size pac-learning bounds.

3.6.3 Strict domination

Finally, we note that the data-efficiency of any fixed-sample-size learner L can be strictly dominated by a sequential learner `Scov`$\wedge L$ that simply runs L and `Scov` in parallel and returns any hypothesis proposed by either; see Figure 3.14. Since `Scov` never returns an ϵ-bad hypothesis, `Scov`$\wedge L$ only makes a mistake whenever L does, and hence is a correct pac-learning procedure whenever L is. Obviously `Scov`$\wedge L$ never observes more examples than L, and is guaranteed to observe fewer for ϵ-reducible spaces.

3.6.4 Assessment

The results of this section demonstrate how further improvements to the data-efficiency of d.s. pac-learning can be obtained by considering alternative learning techniques beyond the cover-based strategies considered in Sections 3.2–3.4. We saw how the reduction-based techniques developed in Section 3.5 for learning with certainty can be more data-efficient than previous cover-based approaches for many natural concept spaces.

However, not every finitely ϵ-coverable (*i.e.*, pac-learnable) concept space is ϵ-reducible, so these reduction-based techniques can only offer an improvement for a strict subset of the pac-learnable concept spaces. So these reduction-based learning procedures are not universal pac-learners in the same sense as the cover-based procedures `BI` and `Sbi`. However, most concept spaces encountered in practice are ϵ-reducible (*cf.* Proposition 3.31), and for many of these spaces we can dramatically improve on the performance of `BI` and `Sbi`; not only requiring fewer training examples on average, but also achieving significantly smaller bounds on maximum training sample size. This raises the important question of determining the generality of this advantage: for what class of concept spaces does `Scut` obtains a significant advantage over the cover-based learning procedures, and how much of an improvement is possible in principle? It appears that these improvements can be obtained for any ϵ-reducible concept space. However, this has yet to be proved, and much work remains to be done in generalizing these results.

Procedure `Scov`$\wedge L$ $(C, \mathrm{P}, \epsilon, \delta)$

- Run `Scov` and L in parallel.
- Return the first hypothesis returned by either procedure.

Figure 3.14: Procedure `Scov`$\wedge L$

3.7 Conclusion

This chapter introduced the idea of using sequential learning procedures to improve the data-efficiency of distribution-specific (d.s.) pac-learning. In the d.s. setting, a concept class over a known domain distribution forms a metric space in a natural way. Previous (general) approaches to d.s. pac-learning adopt this view and exploit the existence of a small covers of the concept space to pac-learn. The idea is to first find a small collection of concepts that accurately approximate every other concept in the space, and then observe a large training sample that is sufficient to accurately estimate the errors of these cover-concepts. In this chapter, we considered a number of ways to improve the data-efficiency of this basic learning strategy.

The first technique we considered was to apply a sequential rather than fixed-sample-size technique to estimating hypothesis errors. Here we observed that a sequential probability ratio test (`sprt`) could estimate the errors of cover-concepts much more efficiently than the previous fixed-sample-size procedure, leading to a significant reduction in the expected number of training examples needed to pac-learn. This improvement was obtained for any concept space, error level ϵ, and failure level δ.

The second idea we explored was to adopt a multiresolution rather than global-cover based strategy to search the concept space. The basic approach is to begin with a coarse search of the concept space and then gradually refine this search to consider smaller neighborhoods near the target concept. This strategy obtains fundamental improvements in data-efficiency over the global-cover based approach for any concept space that is uniformly dense across local neighborhoods.

After exploring these direct techniques for improving the data-efficiency of d.s. pac-learning, we then noted as an aside that sequential procedures can learn with certainty under the d.s. model rather than just high probability. We first demonstrated this in a simple case study and then formalized the general problem of certainly approximately correct (cac) learning. Here I proposed a simple generic learning procedure, `Scov`, that correctly cac-learns a wide range of concept spaces. Surprisingly, this simple procedure cac-learns with optimal data-efficiency, in that any other procedure that observes fewer training examples than `Scov` cannot meet the cac-criterion for every target concept in the space. In addition to optimality, it was also shown that `Scov` is a universal cac-learning procedure in the sense that it successfully cac-learns any concept space for which this is possible in principle.

Surprisingly, this "certain" learning procedure `Scov` uses dramatically fewer training examples than any of the previous *pac*-learning strategies in many case studies, even though it obtains a strictly higher level of reliability. In fact, we saw that `Scov` could even be modified to learn with a much smaller fixed training sample size than current fixed-sample-size learning procedures. This counterintuitive result reveals how the learning with certainty model can provide an alternative technique for deriving data-efficient *pac*-learning procedures. Of course, the scope of this technique is somewhat limited as `Scov` is not applicable to every pac-learnable concept space. However, `Scov` cac-learns (and hence pac-learns) most natural concept spaces.

Contributions

The research reported in this chapter constitutes the first step towards developing learning procedures that achieve the same accuracy and reliability guarantees as current fixed-sample-size pac-learning procedures, but use far fewer training examples in practice. As in the previous chapter, we observed that sequential learning strategies can substantially reduce the number of training examples needed to pac-learn. In fact, we were able to prove these advantages analytically, rather than just establish them empirically. Interestingly, the techniques developed in this chapter also lead to significant computational-efficiency improvements in some cases.

The data-efficiency improvements we achieve range from constant factors to significant improvements in scaling behavior, depending on the concept space and learning technique used. These substantial improvements in data-efficiency enhance the practical applicability of pac-learning to real learning situations. In practice, even a constant reduction in training sample size can be the difference between a useful and an irrelevant theory.

3.7.1 Research directions

Although specific directions for future research were discussed throughout the chapter, here I briefly summarize some of the more important areas for future work. The work on learning with certainty (Section 3.5) raises important questions concerning the computational feasibility of implementing procedures such as Scov, as well as characterizing the complexity of concept spaces with respect to cac-learning. Other research directions involve investigating alternative learning criteria and models not covered in this thesis; in particular, exact learning, and pac-learning with respect to classes of domain distributions. Finally, I describe how the reduction-based learning techniques developed here can be scaled-up to deal with the distribution-free and noisy learning models.

Computational feasibility: Although the primary goal of this chapter was to improve the data-efficiency of pac-learning, it is also important to develop computationally-efficient procedures. The techniques developed in this chapter lead to computationally-efficient procedures for learning simple concept spaces like (initials, uniform) and (d-π-initials, *uniform*) on $[0,1]$, and (monomials, uniform) on $\{0,1\}^n$. However, for these simple spaces polynomial time pac-learning procedures were already known. It would be interesting to see if computationally-efficient learning procedures could be found for more interesting spaces like (halfspaces, uniform) on $[-1,1]^n$, or spaces involving multilayer-perceptrons, *etc.* Perhaps also some of these ideas could lead to a polynomial time procedure for pac-learning (dnf, uniform), a long standing open problem in computational learning theory.

Characterizing complexity: For the cac-learning model, the most important remaining challenge is to develop a characterization of concept space complexity that determines tight bounds on achievable data-efficiency. This characterization is important because it helps identify those spaces where the reduction-based learning strategies Scov and Scut can yield significant advantages over previous cover-based approaches. The current results show that ϵ-reducibility is both necessary and sufficient for cac-learnability, but fail to yield a tight characterization of Scov's data-efficiency. Perhaps, some refinement of this reducibility notion is needed to achieve an appropriate characterization. Another approach is to generalize the tight analyses of special cases like (d-π-initials, uniform) to handle a wider variety of concept space structures.

Exact learning: In Section 3.5 we observed how sequential learning procedures can learn with certainty under the d.s. model. That is we can achieve pac$(\epsilon, 0)$-learning under the d.s. model, not just pac(ϵ, δ)-learning. Similarly, one can consider the problem of *exact* learning; *i.e.*, return a hypothesis with zero error, not just error $\epsilon > 0$. In fact, we immediately obtain two variants of this problem: probably exact learning (pac$(0, \delta)$) and certainly exact learning (pac$(0,0)$). Clearly, each of these two criteria can be met under the d.s. model for some concept spaces. For example, it is easy to see that any finite space can be pac$(0,0)$-learned under the d.s. model, simply by observing training examples until the target concept is identified up to zero-error variants. This raises the question of determining the range of concept spaces that can be pac$(0, \delta)$ and pac$(0,0)$ learned, and determining the distributional assumptions necessary to achieve these criteria.

Classes of distributions: Returning to the standard pac(ϵ, δ)-criterion, note that it is also possible to consider the problem of pac-learning a concept class C with respect to a *class* of domain distributions $\mathcal{P}$, rather than just learning C with respect to a single distribution P, or every possible domain distribution. This problem has been considered in the literature by Kulkarni [1991]. It has been conjectured that pac-learnability of a concept class C with respect to a class of distributions $\mathcal{P}$ is determined by the existence of a uniform bound on metric entropy, $N_\epsilon(C, \mathrm{P})$, over all distributions $\mathrm{P} \in \mathcal{P}$ [Benedek and Itai, 1991]. However, Dudley *et al.* [1994] have recently shown that this conjecture is false. Therefore, the challenge of finding a complexity measure that determines the pac-learnability of general $(C, \mathcal{P})$ remains open. In spite of this fact, it would still be interesting to consider the effectiveness of sequential learners for this task; *i.e.*, identifying effective sequential learning procedures for this problem, quantifying their data-efficiency, and comparing this to fixed-sample-size approaches.

Detecting training convergence: From a practical perspective the most interesting direction for future research is based on the observation that any d.f. pac-learning problem can be turned into a d.s. problem simply by exploiting *unlabelled* training examples. That is, we can use a large collection of unlabelled training examples to accurately estimate all inter-concept distances (without observing a single labelled training example) and then apply the d.s. learning strategies developed in this section to significantly reduce the number of labelled training examples needed to pac-learn. Since unlabelled examples are presumably cheaper, why use labelled examples to estimate inter-concept distances? Although this strategy is likely to be computationally expensive and require large amounts of unlabelled training data, it can be extremely advantageous in situations where labelled training data is expensive to obtain; for example, in medical diagnosis or geological applications. Another interesting aspect of this strategy is that is generalizes more readily to handle noise than the d.f. strategy developed in Chapter 2. In fact, these metric-based learning strategies seem applicable to general function approximation problems, not just classification learning.

Chapter 4

Characterizing rational versus exponential learning curves

4.1 Introduction

When learning a concept from random examples we naturally expect the accuracy of a learner's hypotheses to improve as it sees more training examples. In some sense, it is the rate of this improvement that best describes the learning performance of the system. Given i.i.d. random examples, we define a hypothesizer's *learning curve* by the expected error of its hypotheses as a function of training sample size. Intuitively, this is what one measures by repeatedly training a hypothesizer on a fixed problem and plotting the average error of its hypotheses as a function of training sample size. Since we expect a learner's hypotheses to improve with increasing training sample size, we measure the quality of a learning curve by the rate at which the average hypothesis error converges to zero.[1]

Thus, here we are considering an alternative aspect of learning performance to pac-learning: the idea is to examine the expected error of a learner's hypotheses, rather than the probability the learner produces a hypothesis with error less than ϵ. Also, we examine a hypothesizer's performance for all training sample sizes, rather than just determining a minimum sample size sufficient to meet the pac-criterion.

Worst case learning curves

Obviously the quality of a hypothesizer's learning curve is determined by its prior knowledge about the underlying target concept and domain distribution. For example, if the exact target concept were known *a priori* then zero error can be trivially achieved. Obtaining rapid convergence to zero error is more interesting if we know less about the target concept and domain distribution beforehand. Here we continue to consider the model of prior knowledge popularized by Valiant [1984]: we assume the target concept c belongs to some known class C, but nothing is known about the distribution of domain objects P, which could be arbitrary. Given this model, we naturally consider what can be achieved in the "worst case, distribution-free" sense. Specifically, for a concept class C we are interested in determining the best learning curve that can be obtained in the worst case over all target concepts in C and domain distributions P.

An analysis of this form has been carried out by Haussler, Littlestone and Warmuth [1988; 1994], who investigate the smallest expected error a hypothesizer can achieve after t training examples in the worst case over all target concepts in C and domain distributions P. In particular, they develop a special hypothesis guessing strategy, HLW, that obtains an expected error of at most $O(\mathrm{VC}(C)/t)$ for any target concept in C and domain distribution. Moreover, they show that *no* hypothesis guessing strategy can do significantly better than this: given t training examples, any hypothesizer must obtain an expected error of at least $\Omega(\mathrm{VC}(C)/t)$ for some target concept c_t in C and some domain distribution P_t. This shows that, overall, the best achievable worst case expected error always behaves as a "rational" function of t (*i.e.*, $\Theta(t^{-1})$) for any reasonable class of concepts C.

[1]Note that we are assuming zero error is achievable here; *i.e.*, that classification is noise-free.

Issue

These results would seem to suggest that we should always expect to observe rational learning curves in practice, at least in the worst case. However, it turns out that rational learning curves are *not* always observed in practice: a recent empirical study by Cohn and Tesauro [1990; 1992] shows that *exponential* learning curves can often be observed in natural situations. They demonstrate this by testing a specific hypothesis generation algorithm, BP (backpropagation), on concept classes defined by fixed neural network architectures over $\{0,1\}^n$. By repeatedly training BP at various sample sizes t and plotting the average error of its hypotheses, Cohn and Tesauro observe learning curves that converge exponentially to zero error (*i.e.*, average error behaving as $e^{\Theta(-t)}$ for training sample size t).

These experimental findings demonstrate a clear limitation of the worst case theory of Haussler, Littlestone and Warmuth for predicting even the shape of the learning curves observed in practice. Of course, these empirical results do not directly contradict the worst case theory, since the exponential behavior was only observed for specific target concepts and domain distributions and this may not reflect the true worst case behavior of the given situation. That is, the discrepancy between the theoretical and empirical results could simply be because the worst case results are unrepresentative of the typical performance observed in practice. However, this simplistic conclusion does not explain all of the results obtained by Cohn and Tesauro: Aside from testing BP on neural networks on $\{0,1\}^n$, Cohn and Tesauro also tested BP on the *same* network architectures defined over $[0,1]^n$; *i.e.*, they considered the same neural network architectures with discrete and real-valued inputs. Although defined on different input domains, these paired concept classes defined by the same network architecture have identical VCdimension (in one case considered), and hence are isomorphic under the previous theory. In spite of this, Cohn and Tesauro observe radically different learning curves in each case: By training BP on the same set of target weights and uniform domain distributions for both the discrete and real-valued cases, they invariably observe:

1. *Finite* concept classes (networks with $\{0,1\}$ inputs) yield exponential learning curves.
2. *Continuous* concept classes (networks with $[0,1]$ inputs) yield rational learning curves.

Moreover, the rational curves they observe in the continuous case closely match the worst case predictions of Haussler, Littlestone and Warmuth. Thus, the worst case theory *can* provide an accurate prediction of learning curve behavior in some cases, so merely stating that the worst case theory does not reflect the "typical" learning curves observed in practice is incorrect—sometimes it does. An adequate theory of empirical learning curve behavior must explain how the worst case theoretical results *are* typical in some situations (continuous concept classes) but not others (finite concept classes).

Observation and approach

These experimental results illustrate a striking dichotomy between rational and exponential convergence that is completely missed by the current worst case theory. In this chapter we provide a simple theoretical explanation of this dichotomy based on making a simple observation about the previous theory: A close inspection of [Haussler, Littlestone and Warmuth, 1988] reveals that their analysis is *non-uniform* in training sample size t. In particular, the lower bound result (that forces an $\Omega(t^{-1})$ worst case expected error) chooses a *different* domain distribution and target concept for each training sample size t. Clearly, this does not reflect the situation normally encountered in practice, nor that investigated by Cohn and Tesauro, where these are held fixed. This raises the question of whether, in a model where the domain distribution and target concept are held fixed, there are situations where the best achievable worst case learning curve is exponential and other situations where it is rational. It turns out the answer to these questions is yes.

Results

By carrying out an analysis of worst case learning curves where the domain distribution and target concept are held fixed, this chapter shows how the dichotomy between rational and exponential learning curves can be explained in the worst case distribution-free (d.f.) setting. In particular, we show that there are concept classes that permit exponential learning curves, even in the worst case; but there are other concept classes that force any hypothesizer to produce rational learning curves, even for fixed domain distributions and target concepts.

We first establish the basic dichotomy between exponential and rational convergence for the simple case of finite versus continuous concept classes. Here it is not hard to show that for a finite class C, any hypothesizer that guesses consistent concepts from C will obtain exponential convergence in the worst case. We also prove that it is impossible to achieve better than exponential worst case convergence for a non-trivial class, so this must be the optimal form of worst case learning curve. However, we then show that exponential convergence cannot be achieved for every possible concept class. In particular, we show that continuous classes force any hypothesizer to obtain rational learning curves for some target concepts and domain distributions, even if these are held fixed. Note that this is stronger than the lower bound result of Haussler, Littlestone and Warmuth [1988], in that it is easier to force bad behavior by choosing a different target concept and domain distribution for each training sample size, than it is to show bad behavior results even when these are held fixed.

These results nicely corroborate the experimental findings of Cohn and Tesauro. The fact that they observe exponential learning curves for finite concept classes is no accident, since this is achieved by *any* consistent hypothesizer. Continuous concept classes, on the other hand, always force rational learning curves to be exhibited, at least in the worst case.

Of course, there is a significant gap between finite and continuous concept classes. Therefore, these results are far from comprehensive; for example, they say nothing about what happens for countably infinite concept classes. This leaves open the question of identifying the precise conditions that dictate between rational and exponential learning curves, and whether other intermediate forms of convergence are possible (*e.g.*, $\Theta(t^{-n})$ or some other form). We obtain exact answers to these questions for the special case of concept *chains*—*i.e.*, classes that are totally-ordered under set-inclusion.[2] It turns out that for a concept chain C, the precise condition that determines the worst case convergence form is the presence or absence of any *dense* subchains in C (*i.e.*, a subchain $D \subset C$ such that between any two concepts $d_1 \subset d_2$ in D there is a third $d_1 \subset d_3 \subset d_2$). We show that somewhere-dense chains force rational worst case convergence, but exponential convergence can always be obtained for nowhere-dense (*i.e.*, "scattered") chains, even in the worst case. Not only does this characterize the precise boundary between rational and exponential learning curves in the d.f. model, it also shows that no other form of worst case learning curve is possible (for concept chains) in this setting.

Finally, we observe that every concept class Cohn and Tesauro considered in their computer simulations was represented with limited precision, and hence fundamentally finite. This means that every learning curve they observed must have been asymptotically exponential—apparently contradicting the fact that they observed rational curves in some situations. This discrepancy is resolved by noting how, at the scale of training sample sizes examined relative to the inter-concept distances, convergence can appear rational even when it is fundamentally exponential.

Overview

This chapter is organized as follows. Before presenting the main results, Section 4.2 first introduces the basic modelling assumptions and briefly surveys the existing theory of worst case learning curves.

Section 4.3 then demonstrates the dichotomy between exponential and rational learning curves in its most fundamental form by examining finite versus continuous concept classes. Once this basic dichotomy is established, Section 4.4 then investigates the exact boundary between rational and exponential worst case learning curves for the special case of concept chains. Here it is shown that the existence of a dense subchain in the original chain C is the precise condition that dictates between worst case convergence modes. This also establishes that only two types of worst case learning curves are possible in the d.f. setting (for concept chains).

Section 4.5 then briefly notes how exponential learning curves can appear rational at small training sample sizes, even when convergence remains asymptotically exponential; thus explaining how the dichotomy between rational and exponential learning curves can be observed in finite precision computer experiments.

[2]Note that in this chapter it will be convenient to think of concepts as *subsets* of the domain, $c \subset X$, rather than indicator functions, $c : X \to \{0, 1\}$.

Next, Sections 4.6 and 4.7 on some preliminary results for extending these analyses to the general case. Here we consider the two issues of *(i)* drawing a general boundary between rational and exponential worst case learning curves, and *(ii)* determining tight bounds on the specific rates of worst case convergence. These questions turn out to be quite difficult however, and only minor progress is reported. The barriers to developing a general theory are identified for both the d.f. and d.s. cases.

Finally, Section 4.8 considers the relationship between learning curve theory and the theory of pac-learning, and concludes the chapter with some suggestions for future research. Overall, these results show how the dichotomy between rational and exponential learning curves can be recovered in the distribution-free setting.[3]

4.2 Background: learning curve theory

Before investigating the specific conditions where rational versus exponential learning curves are obtained, this section briefly introduces the mathematical model used in this analysis and surveys the relevant existing research.

4.2.1 Model

We continue to consider the standard problem of learning an accurate concept definition from (noise free) random examples. Recall that in this chapter it will be convenient to think of concepts as *subsets* of the domain, $c \subset X$, rather than indicator functions $c : X \to \{0,1\}$. Thus, an *example* is a pair $\langle x, 1_c(x)\rangle$ that specifies a domain object x and gives the value of c's indicator function at x. To simplify notation we let $c\mathbf{x}^t \triangleq \langle\langle x_1, 1_c(x_1)\rangle, ..., \langle x_t, 1_c(x_t)\rangle\rangle$ denote the sequence of training examples generate by a target concept c and object sequence $\mathbf{x} = \langle x_1, ..., x_t\rangle$.

Given these definitions, we consider the same *batch* training protocol as before: the learner is given a finite sequence of training examples $c\mathbf{x}^t$ from which it must produce a hypothesis $h \subset X$ that is then tested *ad infinitum* on subsequent test examples. But rather than investigate learning procedures that must choose an appropriate training sample size as well as produce an accurate hypothesis, we focus only on the hypothesis guessing strategies themselves. In particular, we consider how the quality of a hypothesizer's guesses improve with increased training sample size. Formally, a *hypothesizer* H is just a mapping from finite sequences of training examples to hypotheses $H : (X \times \{0,1\})^* \to 2^X$.

As usual, we adopt the usual i.i.d. random examples model, which assumes domain objects are independently generated according to some fixed domain distribution P and classified according to some fixed target concept c. Here, the *error* of a hypothesis h with respect to target concept c and domain distribution P is given by $\mathrm{P}(h \vartriangle c)$, where $\vartriangle$ denotes symmetric set-difference. Thus P defines a natural (pseudo)metric, $d_{\mathrm{P}}(h, c) \triangleq \mathrm{P}(h \vartriangle c)$, over the space of concepts on X. For a given training sequence $c\mathbf{x}^t$, we denote the learner's hypothesis by $H[c\mathbf{x}^t]$, and the error of this hypothesis by $err(H, \mathrm{P}, c, \mathbf{x}^t) \triangleq d_{\mathrm{P}}(H[c\mathbf{x}^t], c)$. Given this model, the hypothesizer's goal is to produce accurate hypotheses given as few training examples as possible.

Under the i.i.d. random examples model we can characterize the overall performance of a hypothesizer H, as follows: First, consider a fixed training sample size t, and notice that a fixed target concept c and domain distribution P induce a distribution training sequences of length t. From this distribution, H maps each training sequence $c\mathbf{x}^t$ to a particular hypothesis, $H[c\mathbf{x}^t]$, and hence induces a distribution over hypotheses. Since each hypothesis $H[c\mathbf{x}^t]$ has a particular error value with respect to c and P, $err(H, \mathrm{P}, c, \mathbf{x}^t)$, H in fact induces a distribution over hypothesis errors. This distribution, referred to as the "t^{th} error distribution of H with respect to c and P" completely characterizes H's learning performance for a given training sample size t. To characterize H's overall learning performance then, we consider the sequence of error distributions H produces as a function of training sample size. "Good" learning performance is characterized by error distributions that quickly become skewed towards zero after few training examples; as illustrated in Figure 4.1.

[3]The key results from this chapter were reported in [Schuurmans, 1995], and a full paper is currently under review [Schuurmans, 1996a]. Permission has been obtained from Springer-Verlag for inclusion of this material here.

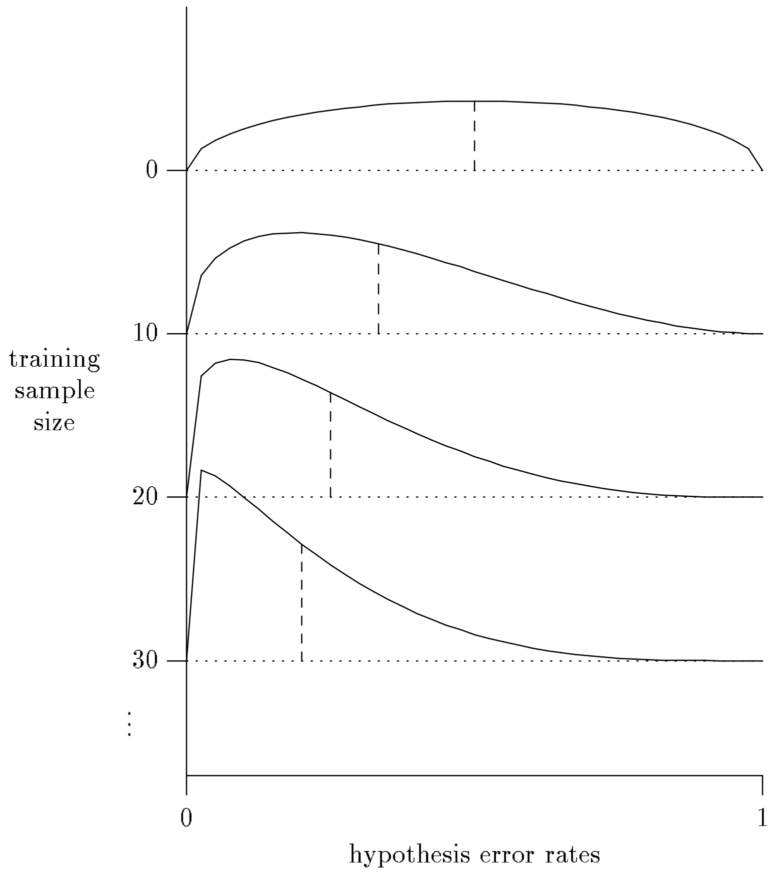

Figure 4.1: Evolving density of hypothesis error rates as a function of training sample size.

Although the error distribution completely describes a hypothesizer's learning performance at a given training sample size, it is usually not possible to consider such a comprehensive characterization in practice. Useful but practical characterizations of learning performance can still be obtained by focusing on some crude aspect of the error distribution's shape; for example, its expected value or tail probabilities. In this chapter we focus on the expected value of a hypothesizer's error distribution and consider the average error attained by H's hypotheses after t training examples, $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t)$.[4] In particular, we define H's **learning curve** (with respect to a target concept c and domain distribution P) by the *expected error* of its hypotheses as a function of training sample size t. So although the sequence of error distributions fully describes H's learning performance, we are only focusing on the average error of H's hypotheses for each training sample size.[5]

Obviously the quality of a hypothesizer's learning curve depends on whatever prior knowledge it has about the target concept and domain distribution beforehand. Most research on learning curves adopts one of two models of prior knowledge: the d.f. model (studied in Chapter 2) where we assume the target concept belongs to some known class C but nothing is known about the domain distribution; and the d.s. model (considered in Chapter 3) where we assume in addition that the precise domain distribution is known *a priori.* In either case, the idea is to characterize the best learning curve that can be achieved in the worst case over all possible target concepts in C and all possible domain distributions P, given whatever prior assumptions we make about P. Obviously the rate at which this best achievable learning curve converges to zero error is determined by the complexity of the concept class C or concept space (C, P), as the case may be. The primary focus in this chapter will be on the d.f. model studied by Valiant.

4.2.2 Distribution-free theory

In Valiant's d.f. model we assume the target concept c belongs to some known class C, but nothing is known about the domain distribution P beforehand. Given this model, we naturally consider what can be achieved in the "worst case, distribution-free" sense. Specifically, for a concept class C we are interested in determining the best learning curve that can be obtained in the worst case over all possible target concepts in C and domain distributions P.

Upper bound

An analysis of worst case learning curves under this model has been carried out by Haussler, Littlestone and Warmuth [1988] who investigate the smallest expected error a hypothesizer can guarantee after t training examples for a concept class C. They develop a special learning strategy HLW which, given t training examples, always produces hypotheses with a small expected error, regardless of the target concept in C and domain distribution P.

Theorem 4.1 [Haussler, Littlestone and Warmuth, 1988, Theorem 5.1] *For any concept class C with $\mathrm{VC}(C) < \infty$: given t random training examples, guessing strategy* HLW *produces hypotheses with an expected error of at most*

$$\mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{HLW}, \mathrm{P}, c, \mathbf{x}^t) \quad \leq \quad \frac{2\mathrm{VC}(C)}{t+1} \tag{4.1}$$

for any target concept $c \in C$, and any domain distribution P.

[4] Interestingly, $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t)$ corresponds to the probability that H's hypothesis after t training examples misclassifies the $t+1$st example [Haussler, Littlestone and Warmuth, 1988, Lemma 6.1]. *I.e.,*

$$\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) \quad = \quad \mathrm{P}^{t+1}\left\{\mathbf{x}^{t+1} : 1_{H[c\mathbf{x}^t]}(x_{t+1}) \neq 1_c(x_{t+1})\right\}.$$

[5] Of course other aspects of an error distribution's shape might be important in particular applications. For example, in pac-learning theory (as studied in the previous two chapters) we consider the *tail probabilities* of the hypothesizer's error distribution: demanding that the hypothesizer produce an error distribution with at most δ of its probability mass concentrated above ϵ. Here however, we simply consider how the *expected* error of a learner's hypotheses evolves with increased training sample size. These two aspects of an error distribution's shape are clearly related: a small expected value implies a small probability in the upper tail, and *vice versa.*

So, for any concept class C with finite VCdimension, the hypothesis strategy HLW achieves a worst case expected error that converges rationally (*i.e.*, $O(t^{-1})$) to zero error, with a specific rate of decrease that is proportional to the VCdimension of C. In later work, Haussler, Littlestone and Warmuth [1994] develop a randomized version of this guessing strategy, HLW_R, and obtain a slightly better bound $\mathrm{E}_{\mathbf{x}^t}\ err(\mathsf{HLW}_R, \mathrm{P}, c, \mathbf{x}^t) \leq \mathrm{vc}(C)/(t+1)$.[6]

Lower bound

In addition to proving a rational upper bound on the expected error obtained by HLW, Haussler, Littlestone and Warmuth also show that *no* hypothesizer can do significantly better than this: for any guessing strategy H there is a sequence of domain distributions $\mathrm{P}_1, \mathrm{P}_2, \ldots$, and a sequence of target concepts $c_1, c_2, \ldots$, that forces H to obtain an expected error of at least $\Omega(t^{-1})$ for training sample sizes $t = 1, 2, \ldots$, *etc.*

Theorem 4.2 [Haussler, Littlestone and Warmuth, 1988, Theorem 5.2] *For any concept class C with $\mathrm{vc}(C) \geq 2$: given $t > d$ training examples,* any *hypothesizer H must obtain an expected error of at least*

$$\mathrm{E}_{\mathbf{x}^t}\ err(H, \mathrm{P}_t, c_t, \mathbf{x}^t) \ \geq \ \frac{\mathrm{vc}(C) - 1}{2e(t+1)} \tag{4.2}$$

for some target concept $c_t \in C$ and some domain distribution P_t.

Therefore, no hypothesis guessing strategy can achieve a worst case expected error that converges faster than rationally to zero error. This shows that the best achievable worst case expected error for each training sample size t must behave as a rational function in t (*i.e.*, $\Theta(\mathrm{vc}(C)/t)$) for any concept class C with finite VCdimension.

Assessment

These results appear to provide a comprehensive characterization of worst case learning curves in the d.f. setting. From Theorems 4.1 and 4.2, it is tempting to conclude that for any concept class C:

1. rational convergence is the best that is achievable in the worst case, and
2. hypothesis strategy HLW always attains (near) optimal worst case learning curves.

However, neither of these conclusions is entirely accurate: It turns out that substantially better than rational convergence can be achieved for many non-trivial concept classes, even in the worst case; and there are alternative guessing strategies that do significantly better than HLW in some situations. The main reason for this discrepancy is that the analysis of Haussler, Littlestone and Warmuth is *non-uniform* in training sample size t. In particular, the lower bound result, Theorem 4.2, forces large expected error rates by choosing a different domain distribution and target concept for each training sample size t. This analysis clearly does not reflect the situation encountered by a learning system in practical applications, nor in empirical learning curve investigations. Consequently, the results of this theory do not always match the learning curve phenomena observed in practice.

For example, the first conclusion is inaccurate, since forcing bad behavior by choosing a series of target concepts and domain distributions can easily overlook situations where sub-rational convergence could be obtained for any fixed choices, even in the worst case. Evidence for this is provided by the experimental results of Cohn and Tesauro which demonstrate that exponential learning curves can be obtained for many finite concept classes. (Granted, these results could simply be missing the worst case behavior, but we will see below that this is not the case.) The second conclusion could be inaccurate since, in a uniform setting

[6]Note that a complete description of Strategy HLW is somewhat involved. In fact, HLW will generally produce hypotheses that do not directly belong to C. The precise details of this strategy will not concern us here; a complete description is given in [Haussler, Littlestone and Warmuth, 1988], where this strategy is referred to as the "1-inclusion graph prediction strategy." The reason they devise this strategy HLW is that the obvious approach of simply guessing arbitrary consistent concepts from C does not work in general: such a strategy can be forced to have expected errors of $\Omega((\ln t)/t)$ (which is not quite rational in t) for certain concept classes [Haussler, Littlestone and Warmuth, 1988, Theorem 6.2]. Therefore, in order to obtain guaranteed rational convergence, the more sophisticated guessing strategy HLW is required.

where the target concept and domain distribution are held fixed, further refinements to the HLW strategy might be necessary to ensure near-optimality. For example, if sub-rational worst case convergence is possible for some concept classes, then Theorem 4.1 no longer guarantees that HLW necessarily achieves near-optimal learning curves.

The main point of this chapter is to consider a *uniform* analysis of worst case learning curves that keeps the domain distribution and target concept fixed for all training sample sizes. It turns out that this more natural model allows us to prove that exponential worst case convergence is indeed possible for some concept classes, but also that there are other classes that force any hypothesizer to obtain rational convergence for fixed domain distributions and target concepts.

4.2.3 Distribution-specific theory

Given the comprehensive nature of the d.f. theory developed by Haussler, Littlestone and Warmuth [1988], most recent research has concentrated on developing a d.s. theory of learning curves. Part of the motivation for this is that stronger assumptions should yield better predictions of the actual learning curves observed in practice. A d.s. analysis (as investigated in Chapter 3) adopts a stronger model of prior knowledge where we assume the domain distribution P is known *a priori*, but the target concept c is known only to belong to some class C. So here we are interested in determining the best learning curve that can be obtained in the worst case over all possible target concepts in some class C given prior knowledge of a concept space (C, P).

A comprehensive theory of d.s. worst case learning curves has yet to be developed to the same extent as the d.f. theory presented above. Most research to date has considered specific examples; determining the convergence rates obtained by particular hypothesis algorithms on specific concept spaces. Given the stronger assumptions of this model, many researchers have demonstrated that exponential convergence can be achieved in specific situations.

For example, many researchers have analyzed the behavior of specific hypothesizers on particular concept spaces and shown that exponential convergence is possible. Much recent work in machine learning, for example, has considered the concept space (monomials, uniform) on $\{0,1\}^n$. For this space, Pazzani and Sarrett [1990] have analyzed the performance of a hypothesis strategy that simply chooses the (unique) smallest monomial concept consistent with the training examples. Similarly, Langley *et al.* [1992] study a hypothesis strategy that guesses the most likely class assuming independent features. In both cases, the authors show that these hypothesizers achieve exponential convergence to any target concept (although this is not explicitly stated as such). In a similar study, Golea and Marchand [1993] investigate the performance of a "clipped Hebb rule" hypothesizer for the concept space (halfspace, uniform) on $\{+1,-1\}^n$, and explicitly prove that it achieves exponential convergence in the worst case.

Some researchers have shown that even *general* hypothesis guessing strategies can obtain exponential learning curves for particular concept spaces. For example, Baum and Lyuu [1991] and Lyuu and Rivin [1992] analyze the same concept space (halfspace, uniform) on $\{+1,-1\}^n$ considered by Golea and Marchand above, but show that in fact *any* hypothesizer that guesses consistent halfspace concepts achieves exponential convergence to any target in this space (for large n). Haussler *et al.* [1994] achieve a similar result for the related concept space (halfspace*, spherical) defined on $\mathbb{R}^n$, where halfspace* is the class of concepts defined by perceptrons with $\{+1,-1\}$ weights, and spherical is any distribution in $\mathbb{R}^n$ that is radially symmetric about the origin. Again, they show that any consistent halfspace hypothesizer achieves exponential learning curves for this space.

Bayesian analyses

Another large body of work concerning the analysis of learning curves is the Bayesian statistical-mechanical approach, which not only assumes the hypothesizer knows the domain distribution P *a priori*, but also has access to a prior distribution Q over possible target concepts. This type of analysis tends to consider the *average case* learning curves obtained by the Bayes and Gibbs hypothesizers in particular. For example, Seung *et al.* [1991] consider the concept space (halfspace*, spherical) defined on $\mathbb{R}^n$, and argue that the Bayes and Gibbs guessing strategies obtain exponential average case learning curves for this space. Similarly, Opper and Haussler [1991] argue that the Bayes and Gibbs strategies obtain rational average case learning curves for the space (halfspace, spherical) on $\mathbb{R}^n$. General analyses [Amari, Fujita and Shinomoto, 1992; Haussler, Kearns and Schapire, 1991] suggest that rational convergence is a natural form of learning curve for this model.

Given the much stronger assumptions of this model it is not too surprising that many researchers have indicated such a dichotomy between rational and exponential average case learning curves; *e.g.*, [Schwartz et al., 1990; Seung, Sompolinsky and Tishby, 1991; Barnard, 1994]. The common suggestion is that this dichotomy is determined by the existence of "gaps" between target concepts; *i.e.*, whether target concepts are isolated under the distribution metric d_{P} [Schwartz et al., 1990; Barnard, 1994]. However, it is easy to refute this suggestion in general [Dudley et al., 1994] (see also the counterexample in Section 4.7), so this really must be an informal observation. Our purpose here is to (rigorously) establish that this dichotomy already exists under the much weaker distribution-free model.[7]

Assessment

The distribution-specific theory, given its much stronger assumptions, appears able to predict a dichotomy between rational and exponential learning curves, both under a worst case and average case analysis. Unfortunately, current analyses only address specific case studies, and therefore do not provide a general characterization of the concept space properties that permit or prohibit exponential convergence. (A preliminary attempt at a general (rigorous) characterization is given by Haussler *et al.* [1994], but only finite concept spaces are addressed in any detail.)

Even though distribution-specific analyses can clearly provide tighter characterizations of the learning curves observed in practice, these analyses require much more problem specific information than a d.f. analysis; in fact, much more than is typically available in practice. This chapter shows how, in a general way, this dichotomy can still be revealed under much weaker assumptions. A benefit of the d.f. approach is the simplicity of analysis and its wider range of applicability. This allows us to draw a clean boundary between convergence types based solely on a simple structural property of concept classes. Any insights obtained here will aid in our understanding of the distribution-specific case also.

4.3 Dichotomy between rational and exponential learning curves

This section investigates the uniform analysis of worst case learning curves under the d.f. model. Here we assume and target concept are held fixed throughout the training process. We establish the dichotomy between rational and exponential worst case convergence in its most basic form by considering finite versus continuous concept classes; showing that the dichotomy between rational and exponential learning curves does indeed exist and can be recovered in the d.f. setting.

Since we are primarily concerned with the distinction between rational and exponential learning curves, we focus on the worst case *asymptotic* form of a hypothesizer's learning curve. In particular, for a concept class C we are interested in determining the worst case asymptotic form of learning curve a hypothesizer can obtain for fixed target concepts in C and fixed domain distributions P.

Definition 4.3 (Worst case learning curve) *We say that a concept class C has a $\Theta'(g(t))$ worst case learning curve, written* $\mathrm{LC}(C) = \Theta'(g(t))$, *if*

1. *There* exists *a hypothesizer H that achieves $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) = O(g(t))$ for every target concept $c \in C$ and domain distribution* P.
2. *For* every *hypothesizer H, there is a target concept $c \in C$ and a domain distribution* P *that forces H to obtain $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) = \Omega'(g(t))$.*

Thus, we say $\mathrm{LC}(C) = \Theta'(g(t))$ if some hypothesizer achieves $O(g(t))$ worst case convergence, but every hypothesizer can be forced to have an expected error of at least $\Omega'(g(t))$ for some fixed domain distribution

[7] It has been suggested that all intermediate forms of convergence between rational and exponential (*e.g.*, $\Theta(t^{-2})$) are possible under this average-case distribution-specific model [Haussler et al., 1994]. However, Haussler *et al.* [1994] also point out that these statistical-mechanical analyses often employ non-rigorous asymptotic arguments, and many of these conclusions may not apply to real situations.

P and target concept in C. (The notation $f(t) = \Omega'(g(t))$ means there exists a constant α such that $f(t) \geq \alpha g(t)$ for infinitely many $t > 0$. We use this weaker definition instead of the standard "for all but finitely many $t > 0$" because there is no way to prevent a hypothesizer from periodically "guessing right" on large training sample sizes. That is, we want to rule out the case of a hypothesizer that systematically cycles through a finite (or countable) concept class, ignoring the training data, and yet periodically achieving zero error for any target concept.)

4.3.1 Intuitive illustration

Before developing the general theory we first illustrate the intuitive source of the dichotomy between rational and exponential convergence in the simplest possible way. First, note that a training sequence $c\mathbf{x}^t$ produced by a concept c and object sequence $\mathbf{x}^t$ reduces a concept class C to a *consistent neighborhood* $C[c\mathbf{x}^t]$ which contains every hypothesis $h \in C$ that correctly classifies every training example in $c\mathbf{x}^t$. Then for any domain distribution P this consistent neighborhood will have a diameter, $dia(C, c, \mathbf{x}^t)$, given by the maximum distance between any two concepts in the consistent neighborhood under the distribution metric d_{P}. Now consider an arbitrary hypothesizer H that is consistent for C (*i.e.*, produces a concept from C that correctly classifies every training example). Clearly, the expected error of H's hypotheses must be bounded by the expected diameter of the consistent neighborhood:

$$\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) \;\leq\; \mathrm{E}_{\mathbf{x}^t} dia(C, c, \mathbf{x}^t). \tag{4.3}$$

So we need only consider the rate at which the diameter of this consistent neighborhood converges to zero as a function of training sample size, as this will give an immediate upper bound on H's learning curve.

To simplify things, we will consider a simple chain of concepts C.

Definition 4.4 (Concept chain) *A* concept chain *is a class C that is totally-ordered under set-inclusion;* i.e., *for every distinct c_1 and c_2 in C, either $c_1 \subset c_2$ or $c_1 \supset c_2$.*

Given a concept chain C, assume that the target c is a lower bound for the chain; *i.e.*, $c = \varnothing$. Then for a training sequence $c\mathbf{x}^t$, the consistent neighborhood $C[c\mathbf{x}^t]$ will simply consist of all concepts between $\varnothing$ and the largest consistent concept ℓ in C. Thus, the diameter of the consistent neighborhood will just be the distance between $\varnothing$ and ℓ, $d_{\mathrm{P}}(\varnothing, \ell)$. Here we are interested in how rapidly the expected diameter of this consistent neighborhood converges to zero as a function of training sample size t.

To measure the width of this consistent neighborhood, define a random variable $Z : X \to [0,1]$ that gives the distance between $\varnothing$ and the largest consistent concept ℓ in C. Then for a training sequence $c\mathbf{x}^t$, the distance between $\varnothing$ and the largest consistent concept in C is given by the minimum of the t observations of Z: $\underline{Z}_t(\mathbf{x}^t) = \min_{x_i \in \mathbf{x}^t}\{Z(x_1), ..., Z(x_t)\}$ (the first order statistic). We now ask: how rapidly does $\underline{Z}_t$ converge to zero as a function of t? It turns out that the specific form of convergence depends on the structure of (C, P). In fact, it is not hard to demonstrate situations where $\underline{Z}_t$ converges exponentially to zero error, and other situations where it converges only rationally.

Case 1: Assume that (C, P) is constructed so that the smallest non-empty concept in C is a non-zero distance away from $\varnothing$ (*i.e.*, assume $\mathrm{P}\{Z = 0\} > 0$). Then it is not hard to see that the width of the consistent neighborhood of C converges exponentially to zero.

Proposition 4.5 *For any bounded random variable $Z : X \to I\!R^+$ with $0 < \mathrm{P}\{Z = 0\} < 1$, $\mathrm{E}\,\underline{Z}_t = e^{\Theta(-t)}$.*[8]

Case 2: Assume instead that (C, P) is constructed so that C contains some concept at every distance $0 < z < 1$ from $\varnothing$. This implies that $Z \sim \mathsf{uniform}(0,1)$ (since clearly $\mathrm{P}\{Z < z\} = z$). This means that the width of the consistent neighborhood fo C will only converge rationally to zero.

Proposition 4.6 *For any random variable $Z \sim \mathsf{uniform}(0,1)$, $\quad \mathrm{E}\,\underline{Z}_t = \Theta(t^{-1})$.*

[8] Proofs of all (original) results stated in this chapter are given in Appendix C.

This illustrate the fundamental dichotomy between rational and exponential convergence in its most basic form: Whenever there is a non-zero probability of eliminating all but the target concept, the expected distance to the furthest consistent hypothesis converges exponentially to zero. However, if there is a chain of hypotheses at every distance $0 < z < \bar{z}$, then the expected distance to the furthest consistent concept converges only rationally to zero.

The remainder of this section extends these simple observations to establish the fundamental dichotomy between concept classes that permit exponential worst case learning curves, and classes that force rational learning curves in the worst case. Specifically, we establish this dichotomy for finite versus continuous concept classes.

4.3.2 Finite concept classes

It is actually not hard to see that exponential convergence can always be achieved for any finite concept class. In fact, *any* consistent hypothesizer H for C obtains exponential convergence in this case, even in the worst case over all target concepts in C and domain distributions P.

Proposition 4.7 (Finite UB) *For any* finite *concept class C: Any consistent hypothesizer H for C obtains an exponential learning curve (i.e., $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) = e^{O(-t)}$) for every target concept $c \in C$, regardless of the domain distribution* P.

This result follows from the simple fact that all non-identical concepts (under P) are eliminated with non-zero probability after a finite number of training examples, and hence Proposition 4.5 applies.

Not only do we obtain exponential convergence for finite concept classes, this is in fact the *optimal* form of convergence that can be obtained for any non-trivial concept class. We say a class C is *non-trivial* if it contains at least two concepts c_1 and c_2 such that $(c_1 \,\Delta\, c_2) \neq \varnothing$ and $(c_1 \equiv c_2) \neq \varnothing$. For a non-trivial class C we can always find a domain distribution P that forces any hypothesizer to exhibit (at least) exponential convergence for some fixed target concept in C.

Proposition 4.8 (Universal LB) *For any* non-trivial *concept class C: There is a domain distribution* P *that forces any hypothesizer H to obtain an exponential learning curve (i.e., $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c', \mathbf{x}^t) = e^{\Omega'(-t)}$) for some target concept $c' \in C$.*

The basic idea behind the proof is to construct a domain distribution that forces any hypothesizer H to obtain an expected error of at least $e^{\Omega(-t)}$ *on average* between a pair of non–mutually-exclusive concepts c_1 and c_2. This can then be used to show that H must produce an exponential learning curve for at least one of the two (for infinitely many training sample sizes).

Thus, we have shown that exponential convergence is the optimal achievable form of worst case learning curve in the d.f. setting for any non-trivial concept class. Overall, Propositions 4.7 and 4.8 show that any non-trivial, finite concept class C has exactly an exponential worst case learning curve, and this is achieved by any consistent hypothesizer H for C.

Corollary 4.9 *Any non-trivial, finite concept class C has an exponential worst case learning curve: $\mathrm{LC}(C) = e^{\Theta'(-t)}$.*

It is interesting to observe how these results compare to the non-uniform theory of Haussler, Littlestone and Warmuth [1988]. Although we obtain an exponential learning curve for any fixed domain distribution, there is no single "worst case" distribution that maximizes the expected error for all training sample sizes t. That is, we have a different worst case distribution for each training sample size t. Moreover, although each worst case curve is exponential, any universal upper bound over all of these curves (that simultaneously takes every domain distribution into account) turns out to have a rational form; see Figure 4.2. It is this rational envelope over a family of exponential curves that is characterized by the non-uniform theory of Haussler, Littlestone and Warmuth. Figure 4.2 also illustrates the discrepancy between the non-uniform bounds of Haussler, Littlestone and Warmuth, and the empirical observations of Cohn and Tesauro: Any empirical study (which considers a single P for all t) will follow one of the exponential curves; whereas the worst case non-uniform bounds (which considers the upper envelop over all worst case learning curves) has a rational form that quickly diverges from any empirical curve.

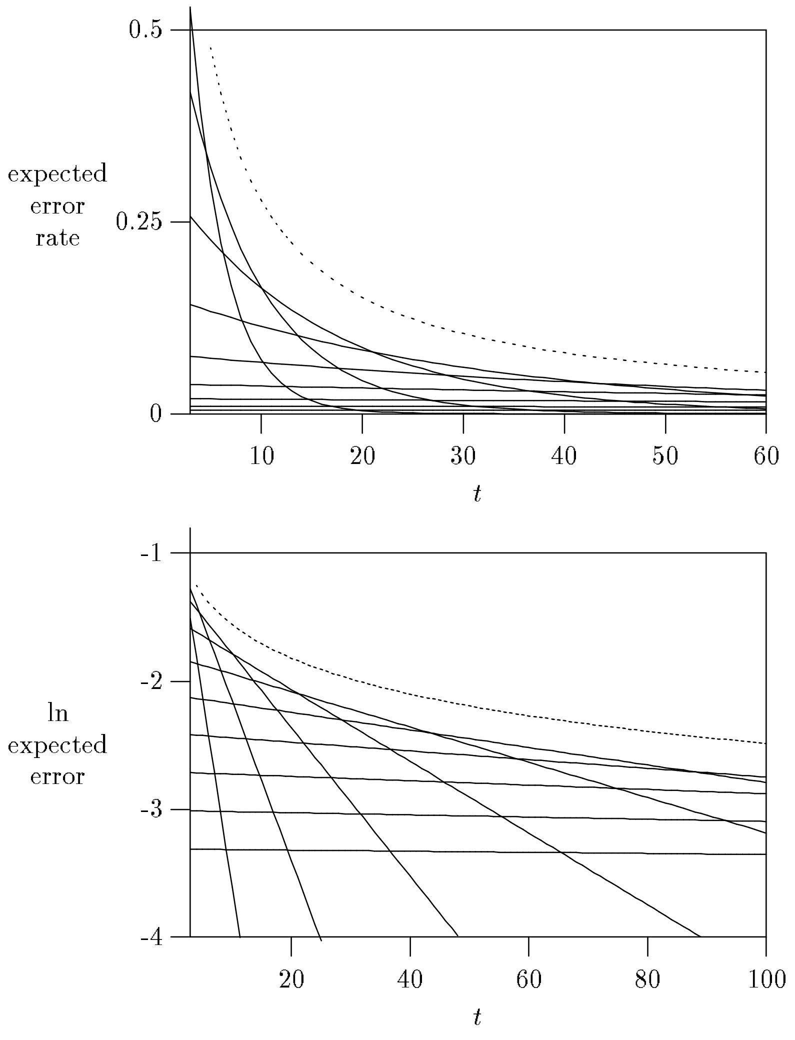

Figure 4.2: Comparing uniform versus non-uniform bounds. This demonstrate how a series of exponential learning curves can have a rational upper envelope.

4.3.3 Continuous concept chains

We have shown that a uniform analysis can yield exponential learning curves for certain concept classes, even though the non-uniform theory of Haussler, Littlestone and Warmuth [1988] predicts only rational forms. This raises the question of whether every concept class permits exponential convergence under a uniform analysis, or whether there are concept classes that can still force rational learning curves even for fixed domain distributions and target concepts. Here we show that there are indeed concept classes that force rational convergence in this case. Evidence for this is given by the experimental results of Cohn and Tesauro that show BP obtains rational convergence whenever it learns from a continuous concept class defined by a neural network architecture on $\mathbb{R}^n$. This suggests that, unless the backpropagation hypothesis guessing strategy is fundamentally flawed in some way, we should not be able to obtain substantially better learning curves in these cases.

To establish conditions where rational convergence is forced, we formalize the notion of a continuous concept chain.

Definition 4.10 (Continuous chain) *A* continuous chain *is a any concept chain C that is order-isomorphic to $\mathbb{R}$;* i.e., *a chain that can be indexed $C = \{c_y : y \in \mathbb{R}\}$ such that $c_y \subset c_z$ for $y \leq z$. (A simple example of a continuous chain is the class of* initial *segment concepts on* $[0, 1]$ *discussed in Chapter 3.)*

Below we show that any continuous concept class forces at least rational convergence in the worst case, even under a uniform analysis. However, before proving this, we first observe that any consistent hypothesizer already achieves rational worst case convergence for any concept chain.

Proposition 4.11 (Chain UB) *For any concept* chain *C: Any consistent hypothesizer H for C obtains a rational learning curve (i.e., $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) = O(t^{-1})$) for every target concept $c \in C$, regardless of the domain distribution* P.

Therefore, the worst case learning curve for any concept chain can never be worse than rational. It remains to show that rational convergence is the best any hypothesizer can achieve in this case.

Theorem 4.12 (Continuous LB) *For any* continuous *concept chain C: There is a domain distribution* P *that forces any hypothesizer H to obtain a rational learning curve (i.e., $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c', \mathbf{x}^t) = \Omega'(t^{-1})$) for some target concept $c' \in C$.*

The proof of this theorem is somewhat involved, but in broad outline follows the same steps as Proposition 4.8: First, we fix a uniform domain distribution P on X, and then fix a uniform prior Q over the class of concepts C. Then we argue that any hypothesizer H must obtain an *average* expected error of at least $\Omega(t^{-1})$ over the set of concepts in C. This is then used to show that there must *be* some target concept $c' \in C$ that forces H to exhibit $\Omega'(t^{-1})$ expected error for infinitely many training sample sizes t. See Appendix C for details.

This result shows that, for any continuous chain C there is a *single* domain distribution that forces any hypothesizer H to exhibit a rational learning curve for some target concept c' in C. Notice that this is a stronger result than Theorem 4.2 as it shows how rational convergence can be forced by a single domain distribution and target concept, as opposed to choosing different distributions and target concepts for each training sample size.

Combining Proposition 4.11 and Theorem 4.12, we see that the worst case learning curve for a continuous concept chain must be exactly rational.

Corollary 4.13 *Any continuous concept class C has a rational worst case learning curve: $\textsc{lc}(C) = \Theta'(t^{-1})$.*

Together, Corollaries 4.9 and 4.13 establish the existence of the fundamental dichotomy between rational and exponential worst case learning curves in the distribution-free model.

4.3.4 Assessment

Overall, these results establish the existence of a fundamental dichotomy between rational and exponential learning curves under the d.f. model. This is contrary to the common suggestion that distributional assumptions are needed to reveal this dichotomy [Seung, Sompolinsky and Tishby, 1991; Haussler et al., 1994].

Our results corroborate the experimental findings of Cohn and Tesauro, in that observing exponential learning curves for finite concept classes is no accident since this is achieved by any consistent hypothesizer, and any continuous class of concepts forces rational convergence in the worst case. Our results also explain why the non-uniform theory of Haussler, Littlestone and Warmuth matches Cohn and Tesauros' empirical results for continuous concept classes, but not for finite classes. In the continuous case, there is a single domain distribution that can force rational convergence as per the theory of Haussler, Littlestone and Warmuth, and this is exactly what Cohn and Tesauro observe in this case. However, for finite classes there is no single domain distribution that can force rational convergence, and here the non-uniform theory no longer applies, as the experimental results demonstrate.

4.4 Boundary between rational and exponential curves

The previous section revealed the basic dichotomy between rational and exponential learning curves by demonstrating that continuous concept chains have rational worst case learning curves and finite concept classes have exponential such curves. Of course, this characterization is far from complete as there is a significant gap between finite and continuous concept classes. In this section we draw an exact boundary between rational and exponential learning curves for the special case of simple concept chains. Specifically, this boundary is determined by the presence or absence of a dense subchain in the original class C.

Definition 4.14 (Dense versus scattered chains) *A concept chain C is* dense *if between any two concepts $c_1 \subset c_2$ in C there is a third $c_3 \in C$ such that $c_1 \subset c_3 \subset c_2$. (Note that we require a dense chain to contain at least two, and hence infinitely many concepts.) We say that an arbitrary chain is* somewhere-dense *if it contains a dense subchain (not necessarily a subinterval), and* nowhere-dense *if it contains no such subchain. Nowhere-dense chains are also referred to as* scattered.

The main contribution of this section is to show that somewhere-dense chains have rational worst case learning curves, whereas nowhere-dense chains have exponential worst case curves. This gives an exact and complete characterization of worst case convergence (for concept chains) under the d.f. model.

4.4.1 Dense concept chains

From Corollary 4.13 above we know that continuous concept chains have rational worst case learning curves. Since any continuous chain is obviously dense it is natural to consider whether this is the *key* property that forces rational convergence in the worst case. Here we show that this is indeed the case: the presence of a dense subchain is sufficient to force any learner to exhibit rational learning curves for some fixed target concepts and domain distributions.

Theorem 4.15 (Dense LB) *For any dense concept chain C: There is a domain distribution P that forces any hypothesizer H to obtain a rational learning curve (i.e., $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c', \mathbf{x}^t) = \Omega^*(t^{-1})$) for some target concept $c' \in C$.*

(Note that we establish the slightly weakened proposition that $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) = \Omega'(t^{-1-\epsilon})$ for any $\epsilon > 0$; hence the notation Ω^*.) This theorem generalizes the previous result concerning continuous concept chains, Theorem 4.12, to arbitrary (*i.e.*, countable) dense concept chains. The basic argument is still the same: We first construct a domain distribution that forces the expected diameter of the consistent neighborhood to converge rationally to zero for any target concept $c \in C$. We then fix a prior distribution Q over C and argue that any hypothesizer H must obtain an *average* expected error of at least $\Omega^*(t^{-1})$ over the concepts in C. Finally, we use this result to show that there must *be* a target concept $c \in C$ that forces H to exhibit $\Omega^*(t^{-1})$ expected error for infinitely many training sample sizes t. Details are given in Appendix C.

Combining Theorem 4.15 and Proposition 4.11, we see that the worst case learning curve for a somewhere-dense concept chain must be rational.

Corollary 4.16 *Any somewhere-dense concept chain C has a rational worst case learning curve:* $\textsc{lc}(C) = \Theta^*(t^{-1})$.

4.4.2 Scattered concept chains

We now consider the complementary class of scattered (nowhere-dense) concept chains. Here we wish to show that exponential learning curves can always be obtained for any fixed target concept and domain distribution. From Proposition 4.8 above we know that it is impossible to achieve better than exponential convergence, so it remains only to show that exponential convergence can always be achieved in this case. This turns out to be somewhat hard to prove: we must actually demonstrate a learning strategy that achieves exponential convergence for any scattered concept chain (or least prove that such a strategy must exist).

Recall from the finite case (Proposition 4.7) that exponential convergence results whenever the hypothesizer can identify the target concept (up to P-equivalence) with non-zero probability after a finite number of training examples. Now consider applying this idea to scattered concept chains; fixing an arbitrary scattered chain C and domain distribution P. First, we observe that any consistent hypothesizer H for C must obtain exponential convergence to an *isolated* target; *i.e.*, a concept that has a least larger neighbor $\ell \supset c$ and a greatest smaller neighbor $s \subset c$ in (C, P). This is because an isolated concept is clearly identified with non-zero probability after just two training examples (eliminating both s and ℓ), and H will guess the target exactly in such cases. Therefore, the only catch must be with *limit* concepts; *i.e.*, concepts that are the limits of infinite ascending $(\bigcup_1^\infty c_i)$ or descending $(\bigcap_1^\infty c_i)$ sequences of concepts in C. (Note that this can easily happen without the chain being dense; see Figure 4.3a.) The problem with limit concepts is that they permit a consistent hypothesizer H to guess an infinite sequence of hypotheses that converge to, but never reach the target. For example, in Figure 4.3a, if the rightmost concept is the target, then a hypothesizer that always guesses the smallest consistent concept in the chain will never reach it. The point is that this guessing behavior can easily lead to rational convergence for certain domain distributions; *e.g.*, as demonstrated in the proof of Theorem 4.15 (Lemma C.8). So the trick to achieving exponential convergence must be to avoid guessing infinite sequences of hypotheses that slowly converge to limit concepts without ever reaching them.[9]

Perhaps the most obvious way to avoid this difficulty is to always guess limit concepts *before* isolated concepts. Then, provided the limit concepts are isolated from one another, we will still achieve exponential convergence, since any target will be guessed with non-zero probability after just two training examples. Of course, it also possible to have *limits* of limit concepts in a scattered chain (*i.e.*, second order limits; see Figure 4.3b). In fact, limit concepts of each order $1, 2, 3...$ are certainly possible; and in fact one can even have limits of these (*i.e.*, concepts of infinite order!); see Figure 4.3c. However, no matter how large these limits become, the simple strategy of guessing the highest order limit concept that is consistent with the training examples (almost) works in general. I refer to this guessing strategy as HOLC; see Figure 4.4. Intuitively, HOLC will obtain exponential convergence as long as the limit concepts are isolated from concepts of the same or higher order, since this will then guarantee that target concept is identified (up to P-equivalence) with non-zero probability after just two training examples. So this isolation property appears to be the key aspect of scattered chains that we need to establish in order to ensure exponential convergence. Fortunately, this turns out to be true. A proof of this fact follows as a corollary to Hausdorff's Theorem, which provides a suitably constructive characterization of the class of scattered linear orderings [Rosenstein, 1982, Chapter 5].

Lemma 4.17 (Corollary to Hausdorff's Theorem) *For a scattered concept chain C, there is some least ordinal γ such that* (i) *every concept in C has order $\beta \leq \gamma$, and* (ii) *all limit concepts of a particular order β are* isolated *in concepts of the same or higher order.*

Given this fact, we expect HOLC to achieve exponential convergence for any scattered concept chain, since if the target concept is isolated in the class of concepts with the same or higher order, HOLC will be guaranteed to guess a zero-error hypothesis with non-zero probability after just two training examples.

[9]Notice that the special learning strategy HLW developed by Haussler, Littlestone and Warmuth [1988; 1994] does not automatically do this, and hence, might produce a rational learning curve when in fact exponential convergence is possible.

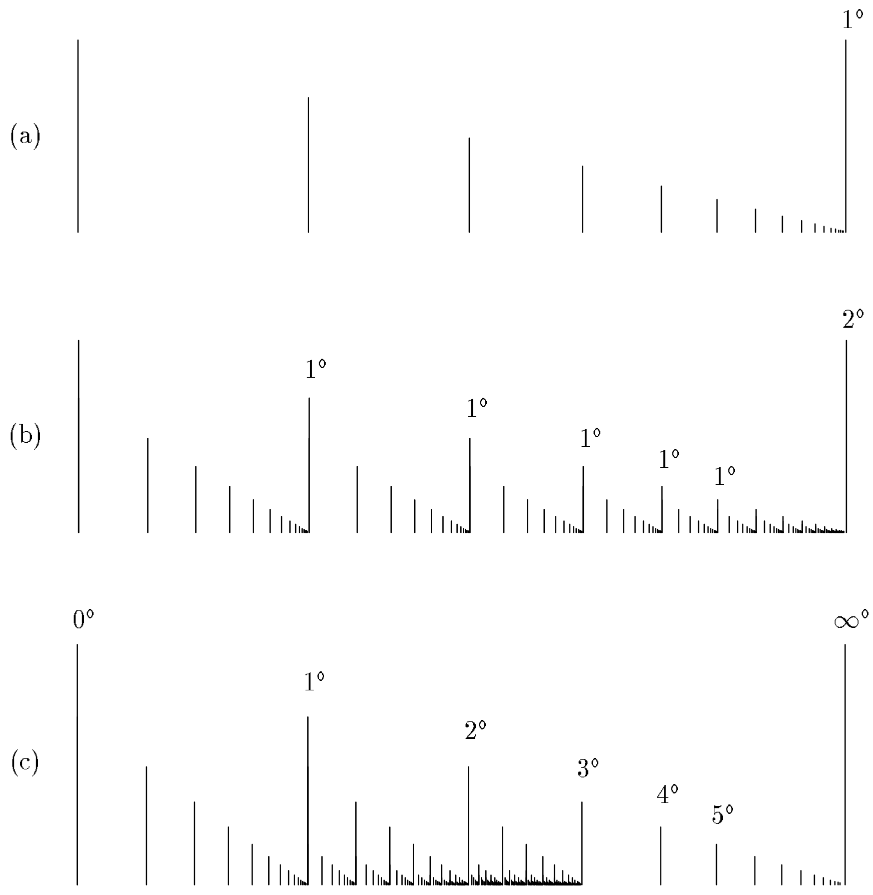

Figure 4.3: This figure depicts three scattered concept chains defined on $X = [0, 1]$. Here, each line y indicates a concept c_y containing all points $x \leq y$. Notice that each of these chains is nowhere-dense since every concept (except the last) has a *least* larger concept. *(a)* A scattered chain with a first order limit concept at the end (*i.e.*, a chain of order type $\omega + 1$). *(b)* A scattered chain with a series of first order limit concepts and a finally a second order limit at the end (*i.e.*, a chain of order type $\omega^2 + 1$). *(c)* A scattered chain with a series of limit concepts of progressively higher order, followed by an infinite order limit (*i.e.*, a chain of order type $\omega^\omega + 1$).

Strategy HOLC $(C, c\mathbf{x}^t)$

INPUT: a *scattered* concept chain C,
a training sequence $c\mathbf{x}^t$ labelled by some unknown target concept $c \in C$.

PROCEDURE:

- Guess the highest order limit concept $h \in C$ consistent with $c\mathbf{x}^t$.

Figure 4.4: Strategy HOLC

Strategy CHOLC $(C, c\mathbf{x}^t)$

INPUT: a *scattered* concept chain C,
a training sequence $c\mathbf{x}^t$ labelled by some unknown target concept $c \in C$.

PROCEDURE:

- Close C under $\cup$, $\cap$ to obtain $\mathcal{C}(C)$.
- Guess the highest order limit concept $h \in \mathcal{C}(C)$ consistent with $c\mathbf{x}^t$.

Figure 4.5: Strategy CHOLC

However, there is one final problem: for HOLC to be applicable, we require that there *exist* a limit concept of maximal order consistent with any sequence of training examples. Unfortunately, this need not be true in general; *e.g.*, consider removing the last concept from the chain depicted in Figure 4.3c. Therefore, HOLC is not always well-defined. However, there is a way to circumvent this difficulty: we first *compactify* the chain C in a natural way, by closing it under $\cap$ and $\cup$ to yield a larger chain $\mathcal{C}(C)$. We call such a chain $\mathcal{C}(C)$ compact for the following natural reason.

Lemma 4.18 *Any chain C that is closed under $\cup$, $\cap$ is also complete and bounded, and satisfies a natural version of the Bolzano-Weierstrass property.*

The key property of this closure $\mathcal{C}(C)$ is that it cannot make a scattered chain dense.

Lemma 4.19 *If C is a scattered chain, then the chain $\mathcal{C}(C)$ formed by closing C under $\cup$, $\cap$ is still scattered.*

Moreover, because of compactness, $\mathcal{C}(C)$ provides a maximum order limit concept for any sequence of training examples.

Lemma 4.20 *For a scattered chain C that is closed under $\cap$, $\cup$, there is always a concept $c \in C$ of* maximal *order consistent with any finite sequence of training examples.*

This leads to our final proposal for a hypothesis guessing strategy that achieves exponential convergence for any scattered concept chain: First compactify the chain, and then guess the highest order limit concept consistent with all the training examples; see Strategy CHOLC in Figure 4.5.

Theorem 4.21 (Scattered UB) *For any* scattered *concept chain C: The hypothesis guessing strategy* CHOLC *obtains an exponential learning curve (i.e., $\mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{CHOLC}, \mathrm{P}, c, \mathbf{x}^t) = e^{O(-t)}$) for every target concept $c \in C$, regardless of the domain distribution* P.

Obviously this hypothesis guessing strategy CHOLC has little direct practical use, but the issues it addresses do shed light on the fundamental nature of worst case learning curves. Theorem 4.21, along with Proposition 4.8, shows that any non-trivial, nowhere-dense concept chain must have exactly an exponential worst case learning curve.

Corollary 4.22 *Any non-trivial, scattered concept chain C has an exponential worst case learning curve:* $\mathrm{LC}(C) = e^{\Theta'(-t)}$.

4.4.3 Assessment

The results of this section provide a complete characterization of the asymptotic form of worst case learning curves for concept chains under the d.f. model. Combining Corollaries 4.16 and 4.22, we have shown the following.

Corollary 4.23 (Exact boundary) *Any concept chain C must have either a rational or exponential worst case learning curve: rational if and only if the chain is somewhere-dense; exponential if and only if the chain is nowhere-dense.*

Not only does this precisely characterize the boundary between rational and exponential worst case learning curves, it also shows that no other form of worst case learning curve is possible for concept chains in the d.f. setting. These results also corroborate the previous dichotomy of Section 4.3: Clearly, finite chains are scattered, and hence yield exponential convergence; whereas continuous chains are intrinsically dense, and hence force rational convergence in the worst case. Moreover, the results of this section subsume the earlier theory by showing how exponential convergence can be obtained for certain infinite concept classes, and how rational convergence can be forced for certain countable concept classes.

4.5 Scaling effects

Although the previous results draw a precise boundary between the conditions when rational and exponential learning curves are obtained, in one sense they miss the point demonstrated by the experimental results of Cohn and Tesauro [1990; 1992]: Since their computer simulations were conducted with finite precision, every concept class Cohn and Tesauro considered must have been fundamentally finite and hence every learning curve they obtained must have been asymptotically exponential. The fact that they observe rational learning curves seems to contradict the results of this chapter. However, the real source of the dichotomy in Cohn and Tesauros' experiments is a scaling effect. That is, every learning curve they observed must have been *asymptotically* exponential, its just that at the scale of training sample sizes they considered, relative to the size of the inter-concept distances, convergence appeared rational. This effect is easily demonstrated by a simple example.

Consider a finite concept chain C_n consisting of $n+1$ concepts $c_0 \subset c_1 \subset \ldots \subset c_n$, and fix a domain distribution P_n that imposes a distance of $1/n$ between adjacent concepts.

Proposition 4.24 *For any $c \in (C_n, \mathrm{P}_n)$,*

$$\left(1-\frac{1}{n}\right)^t \frac{1}{4(t+1)} \quad < \quad \mathrm{E}_{\mathbf{x}^t}\, wid(C_n, \mathrm{P}_n, c, \mathbf{x}^t) \quad < \quad \left(1-\frac{2}{n}\right)^t \frac{2}{t+1}.$$

Notice that this bound is exponential in t and well approximated by $e^{-at/n}b(t+1)^{-1}$ for large n. Now, if we focus on training sample sizes t that are small relative to n, the $e^{-t/n}$ factor will behave like a constant near 1 and the $(t+1)^{-1}$ factor dominates. Here we would observe apparently rational convergence, even though convergence remains asymptotically exponential. To reveal the underlying exponential convergence, we would have to consider training sample sizes on the order of $t = n, 2n, 3n, \ldots$; which in the "continuous" case considered by Cohn and Tesauro [1990; 1992] is on the order of "computer BIGNUM." This partly explains the observed dichotomy, as they considered the *same* training sample sizes for concept classes with vastly different inter-concept distances; observing exponential convergence when the gaps were large, and rational convergence when the gaps were small. (Barnard [1994] makes a similar observation.)

This concludes the main contributions of this chapter. The next two sections discuss preliminary results concerning the general boundary between rational and exponential convergence for arbitrary concept classes (spaces), and general characterizations of specific convergence rates.

4.6 Towards a general distribution-*free* theory

Although we have established a precise characterization of the dichotomy between rational and exponential learning curves for simple concept chains, from a practical perspective these results are still limited. The primary shortcomings are the restriction to simple concept chains, and the lack of any reasonable quantitative predictions. In general, we require a characterization of the boundary between rational and exponential convergence for arbitrary concept classes, not just concept chains; and tight bounds on the specific rates of (worst case) convergence. Unfortunately, both of these problems present serious difficulties, and only preliminary results can be reported here. Below I discuss the challenge of extending the theory in each of these two directions—pointing out specific barriers to developing a general theory, and presenting the prospects for progress in each direction.

General boundary

Theorem 4.2 shows that finite VCdimension is necessary to ensure worst case convergence for a concept class in the d.f. setting. Given the results of Section 4.4, it is tempting to conjecture that we can always achieve exponential convergence for any class C (with finite VCdimension) that does not contain a dense chain. However, we have to be specific about what we mean by containment here: It is possible for a concept class not to directly contain any dense chain and yet still form a dense chain over some restricted subset of the domain. For example, the concept class $C = \{[0, y] \cup \{1 + y\} : y \in [0, 1]\}$ defined on $X = [0, 2]$ contains no chain longer than one, and yet defines a continuous chain on the subdomain $[0, 1]$. In this case, a domain distribution concentrated on $[0, 1]$ could force rational convergence, even though the original concept class contains no dense concept chains *per se*. So obviously by containment we mean C forms a dense subchain over some (measurable) subset of the domain.

Conjecture 4.25 *For any concept class C with* $\mathrm{VC}(C) < \infty$*: If C contains a dense chain, then $LC(C) = \Theta'(t^{-1})$. If C does not contain a dense chain, then $LC(C) = e^{\Theta'(-t)}$.*

The first assertion is obviously true and follows directly from Theorems 4.1 and 4.15. The second assertion however is far from obvious, and would be extremely hard to prove. In effect, one would have to demonstrate a hypothesis guessing strategy that always achieves exponential convergence for any concept class (with finite VCdimension) that does not contain a dense chain. This would involve generalizing Strategy CHOLC to somehow deal with arbitrary nowhere-dense concept classes, which appears to be an extremely difficult task. Limited progress can be made however. For example, the previous results can easily be extended to handle products of chains.

Definition 4.26 (Product chain) *Consider a partition $X_1, ..., X_d$ of some domain X, and a collection of concept chains $C_1, ..., C_d$ defined on each element of the partition.* I.e., *each concept $c \in C_i$ has its extension contained in X_i, and has empty intersection with X_j for $j \neq i$. Then the d-*fold product chain*, C^d, is formed by taking arbitrary unions of concepts selected from each basis chain.*

Intuitively, a concept in a product chain is defined by independently choosing an endpoint from each subdomain X_i. Clearly, the VCdimension of a d-fold product chain is d. (An example of a product chain is the class of d-π-initial concepts examined in Chapter 3.) The results of Section 4.4 can be easily generalized to provide the exact boundary between exponential and rational convergence for product chains.

Observation 4.27 *For any product chain C^d with $d < \infty$: If some base chain of C^d is dense, then $LC(C) = \Theta'(t^{-1})$. If every base chain of C^d is scattered, then $LC(C) = e^{\Theta'(-t)}$.*

The first part follows directly from Theorems 4.1 and 4.15. The second assertion can be proved by considering the obvious extension of Strategy CHOLC: simply apply CHOLC to each basis-chain independently and return the union of its hypotheses as the final conjecture. Since each basis-chain is scattered, we obtain exponential convergence for each basis-chain, and hence exponential convergence overall.

Convergence rates

From a practical perspective, it is important to predict the specific rates of convergence in addition to the functional forms of learning curves. For example, a characterization of convergence rates could be used to choose the complexity of a hypothesis class relative to the amount of training data available, or determining whether sufficient data was available to achieve desired error levels, *etc.* Of course, a necessary prerequisite for any predictive theory of convergence rates is to be able to predict whether rational versus exponential convergence will take place. Obviously we cannot hope to predict specific convergence rates without first being able to predict the underlying functional form of the learning curve.

For dense product chains, given Proposition 4.27 above, it is easy to obtain tight bounds on the precise rates of worst case convergence.

Observation 4.28 *For any product chain C^d, $d < \infty$, that contains $d' \geq 1$ somewhere-dense basis chains: $LC(C^d) = \Theta'(d'/t)$.*[10]

Unfortunately, one cannot give a correspondingly tight characterization of exponential convergence rates for scattered product chains. The problem is that there is no single "worst case" domain distribution (Figure 4.2), and the envelope over all worst case learning curves only has a rational form. Therefore, any tight characterization of convergence rates will have to take into account the specific domain distribution P. This supports the view that some form of d.s. analysis is needed to achieve accurate predictions of empirical learning curves in general.

4.7 Towards a general distribution-*specific* theory

As discussed in Section 4.2, it is also possible to consider a d.s. rather than d.f. analysis of worst case learning curves. Here we assume the domain distribution P is known *a priori* but the target concept c is known only to belong to some class C. The motivation for a d.s. analysis is that incorporating strong distributional assumptions should enable us to more accurately predict the learning curves observed in practice.

Convergence rates

As mentioned in Section 4.2, a general characterization of worst case learning curves has yet to be developed in the distribution-specific setting. Haussler *et al.* [1994] report a preliminary attempt for finite concept spaces, based on characterizing the convergence rates of the expected diameters of consistent neighborhoods. Recall from the discussion in Section 4.3.1 that this will immediately give an upper bound on the learning curve for any consistent hypothesizer H. Unfortunately, this approach cannot work in general: The problem is that many concept spaces have consistent neighborhoods that never converge, and yet special hypothesis guessing strategies can still achieve worst case convergence to zero error for these spaces.

A simple example of this is the space $(\{\text{finite sets}\} \cup \{[0,1]\}, \mathsf{uniform})$ defined on $[0,1]$. Notice that the class $\{\text{finite sets}\} \cup \{[0,1]\}$ has infinite VCdimension, and hence no finite number of training examples can ever eliminate every ϵ-bad concept for $\epsilon > 0$; meaning that the consistent neighborhoods never converge. On the other hand, a trivial guessing strategy achieves worst case convergence for this space: simply guess $\varnothing$ or $[0,1]$, whichever correctly classifies the most training examples. This strategy achieves immediate convergence to zero error wp1 after just one training example.

This example shows that a general characterization of minimax learning curves for concept spaces cannot be based on the convergence of consistent neighborhoods. In Chapter 3 we saw that the general pac-leanability of concept spaces was determined by the finiteness of their metric entropy; *i.e.*, whether the space can be finitely ϵ-covered at various scales ϵ. Here we can exploit the same cover-based approach discussed in Chapter 3 to obtain a hypothesis guessing strategy that always achieves worst case convergence to zero error for any concept space, where this is possible. The idea is to construct an α_t-cover of the space at some appropriate scale α_t for the training sample size t, and then simply guess the hypothesis from this cover that correctly classifies the most training examples; see Strategy BC in Figure 4.6. It turns out that this strategy obtains a worst case learning curve that is determined by the effective dimensionality of the concept space (*cf.* Section 3.4.2); *i.e.*, how fast the space's α-covers grow as a function of $\alpha \to 0$.

Theorem 4.29 *For any space (C, P) such that $N_\alpha(C,\mathrm{P}) = \Theta(1/\alpha)^d$ for some d as $\alpha \to 0$:*

1. *Strategy BC achieves $\mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{BC}, \mathrm{P}, c, \mathbf{x}^t) = O((d/t)\ln(t/d))$ for any target $c \in C$.*
2. *Any hypothesizer H obtains $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c', \mathbf{x}^t) = e^{\Omega'(-t/d)}$ for some target $c' \in C$.*

[10] For the upper bound, use a product version of CHOLC to obtain exponential convergence on the $d-d'$ scattered basis-chains and then apply HLW to obtain $O(d'/t)$ convergence on the remaining d' dense chains. The lower bound requires a product version of Theorem 4.15.

Strategy BC $(C, \mathrm{P}, c\mathbf{x}^t)$

INPUT: target concept space (C, P),
a training sequence $c\mathbf{x}^t$ labelled by some unknown target concept $c \in C$.

PROCEDURE:

- Let a and d be constants such that $N_\alpha(C, \mathrm{P}) \leq (a/\alpha)^d$.
- Let $\alpha_t = \frac{32d}{t} \ln \frac{at}{32d}$ (applicable whenever t is sufficiently large to ensure $\alpha_t < 1$).
- Find an α_t-cover of (C, P), V_t, with size $|V_t| = N_{\alpha_t}(C, \mathrm{P})$.
- Return the hypothesis in the cover $h \in V_t$ with minimum observed error on $c\mathbf{x}^t$.

Figure 4.6: Strategy BC

Part 1 shows that BC achieves worst case convergence to zero error for any space with finite effective-dimension. Part 2 shows that finite effective-dimension is not only sufficient but necessary for any hypothesis strategy to obtain worst case convergence to zero error. Thus, BC is a universal guessing strategy in the sense that it achieves worst case convergence whenever this is possible in principle. Unfortunately, this theorem only provides loose bounds on BC's rate of worst case convergence, admitting both super-rational and exponential convergence. It remains an open question whether these bounds can be improved and when. Perhaps useful bounds could be obtained for the special case of ϵ-reducible concept spaces as studied in Chapter 3.

General boundary

Surprisingly, just characterizing the boundary between rational and exponential convergence is harder in the d.s. case than the d.f. model. In fact, it is not even obvious what property distinguishes rational from exponential worst case learning curves for simple chains in this case. It has often been suggested in the literature [Barnard, 1994; Schwartz et al., 1990; Seung, Sompolinsky and Tishby, 1991] that density in the distribution metric d_P should determine whether rational versus exponential learning curves are obtained. However, this suggestion is easily shown to be false.

Consider a concept chain (C, P) where $C \triangleq \{\mathbf{q} = [0, q] : q \in \mathbb{Q}[0, 1]\}$ contains the initial segments of $[0, 1]$ with rational endpoints, and P is any distribution on $[0, 1]$ such that $\mathrm{P}\{q\} > 0$ for all (and only) rational points of $[0, 1]$. Clearly, C is dense under d_P (since for every $\mathbf{q} \in C$ and $\epsilon > 0$ there is an $\mathbf{r} \in C$ such that $d_\mathrm{P}(\mathbf{q}, \mathbf{r}) < \epsilon$) and yet a simple learning procedure always achieves exponential convergence for this space: simply guess the smallest rational initial segment $\mathbf{r}$ consistent with the training sequence. Since each $\mathbf{q} \in C$ has a *gap* between it and all smaller concepts (because $q \in \mathbf{q}$ but $q \notin \mathbf{r}$ for all $\mathbf{r} \subset \mathbf{q}$) we will guess $\mathbf{q}$ with non-zero probability after one training example, and hence achieve exponential convergence.[11]

This example shows that density in the distribution metric d_P is not sufficient to force rational convergence in general. Instead we need some property of "density from all sides." Unfortunately, extending this notion to arbitrary concept spaces appears intractable, and the prospects for a general theory seem remote. As long as we cannot distinguish between rational and exponential convergence, it is unlikely that we can develop a tight quantitative characterization of empirical learning curves. So this difficulty has serious implications for any theory (d.s. or otherwise) that purports to provide a general and tight characterization of the learning curves observed in practice.

[11]Note that this does not contradict Theorem 4.15, as we have only shown that exponential convergence can be achieved for some, but not every domain distribution.

4.8 Conclusion

This chapter provided a theoretical explanation of the dichotomy between exponential and rational learning curves observed in practice. This theoretical explanation is based on a uniform analysis of worst case learning curves where we keep the domain distribution and target concept fixed for all training sample sizes. This model allowed us to prove that exponential worst case convergence is possible for certain concept classes, but other concept classes force any hypothesizer to obtain rational worst case learning curves.

First, we showed that exponential learning curves can always be obtained for finite concept classes, whereas classes that contain continuous chains always force rational learning curves for some fixed target concepts and domain distributions. Next, we generalized this result to obtain an exact boundary between exponential and rational worst case learning curves for concept *chains*. Here we found that exponential convergence could always be obtained for nowhere-dense chains, but somewhere-dense chains force rational convergence in the worst case. This shows that, for concept chains at least, rational and exponential convergence are the only two types of worst case learning curves in the d.f. setting.

Next, we observed how the original dichotomy between rational and exponential convergence observed by Cohn and Tesauro was actually due to a scaling effect: if the training sample sizes are small relative to the inter-concept distances, then convergence can appear rational even when it is asymptotically exponential.

Finally, I presented some preliminary results towards generalizing the theory to arbitrary concept classes. In particular, the previous theory was extended to handle product chains. We also briefly considered the d.s. case, and introduced a universal hypothesis guessing strategy that always obtains worst case convergence whenever this is possible. Unfortunately, it proved difficult to achieve a completely general theory in either the d.f. or d.s. case, and various specific reasons for this were discussed in detail.

Contributions

The analysis considered in this chapter differs from the preceding work of Haussler, Littlestone and Warmuth [1988; 1994] in that we considered the worst case learning curves that are obtained when the target concept and domain distribution are held fixed throughout the training process. Consequently, our results go further towards explaining many salient learning phenomena observed in practice. For example, we are able to predict whether exponential versus rational learning curves will be observed, in exactly the same circumstances as observed by Cohn and Tesauro [1990; 1992].

One possible objection to the analysis presented in this chapter is that it only characterizes the (best achievable) *worst case* learning curves, which might not be representative of the learning curves observed in practice. For example, even though a concept class might force rational learning curves in the worst case, it still might be possible for a hypothesizer to exhibit exponential convergence to particular target concepts. However, this is not what Cohn and Tesauro observe: whenever the worst case theory predicts even the possibility of rational convergence, that is exactly what their experiments show. So in terms of the mode of convergence, their experimental results coincide with our worst case theory. That is, if the worst case theoretical results are unrepresentative in practice (as far as the type of convergence is concerned), then the experimental results have yet to demonstrate this.

Relationship to pac-learning

Given the striking dichotomy between rational and exponential learning curves, it seems natural to try and exploit these results to achieve better pac-learning performance. Unfortunately, it turns out that exponential convergence does not confer an immediate advantage for pac-learning. In fact, the following example demonstrates that two concept spaces can have identical pac-properties for a given ϵ, while one space permits exponential and the other rational learning curves.

Consider two concept chains C_f and C_c. Let C_f to have two concepts, $\varnothing$ and $\ell \neq X$. Fix a domain distribution P_f so that the distance between $\varnothing$ and ℓ is exactly ϵ. Next, define C_c to be a continuous concept chain $C_c = \{c_y : y \in [0,1]\}$ and fix a uniform domain distribution P_c so that the distance between two concepts c_i and c_j is given by $|i-j|$. Clearly, C_f has an exponential worst case learning curve, whereas C_c has a rational worst case learning curve. However, the number of training examples needed to achieve pac(ϵ,δ)-learning can be the same in both cases! To see this, fix a target concept $\varnothing$ in both cases, and consider the probability that all ϵ far concepts are eliminated after t training examples. Here we see that this probability is the same in each case, and so the minimum number of training examples needed to achieve the pac-criterion is also the same.

This example shows that an isolated target concept may have nothing to do with the probability of eliminating all ϵ-bad concepts from the space (if the gap size is less than ϵ). A difference in pac-learning finite and continuous concept classes would only be revealed by considering progressively smaller error levels $\epsilon \to 0$.

Research directions

The main direction for future research is to characterize the boundary between exponential and rational worst case learning curves for *general* concept classes and spaces. However, Sections 4.6 and 4.7 show that this is quite difficult. The other main research question is to develop a precise (d.s.) characterization of specific convergence rates, if not for the general case, then for useful and interesting subclasses of concept spaces.

Chapter 5

Conclusions

5.1 Synopsis

This thesis addressed the problem of learning a classification function from examples: given a sequence of training examples $\langle\langle x_1, c(x_1)\rangle, \langle x_2, c(x_2)\rangle, ..., \langle x_t, c(x_t)\rangle\rangle$ generated by some unknown classification scheme $c : X \to Y$, how can we produce a global classification function $h : X \to Y$ that agrees with c over as much of X as possible? This simple abstract task is by far the most studied problem in machine learning research. Most work, however, empirically evaluates specific *ad hoc* strategies for mapping training sequences to hypotheses given particular representations for domain objects x and classification functions h. In this thesis we pursued a general theoretical analysis of this problem.

A theoretical analysis of classification learning requires one to adopt a mathematical model of the learner's environment. The specific model we considered was the i.i.d. random examples model, which assumes there is a fixed distribution of examples from which all training and test examples are generated randomly and independently. This is a natural model of many learning situations where successive training examples are not correlated and the example generating mechanism remains stable over time.[1] Given this model, we addressed the standard batch training protocol where in an initial training phase the learner is given access to a (finite) sequence of training examples, from which it must produce a classification function $h : X \to \{0, 1\}$ that is then tested *ad infinitum* in a subsequent testing phase. Here the learner's goal is to produce an accurate hypothesis as quickly and reliably as possible.

Of course, the learner's success depends strongly on the prior knowledge it has about the underlying target concept and domain distribution. A theoretical analysis of classification learning requires us to explicitly model the prior knowledge/constraints the learner has about the target concept and domain distribution before training. In this thesis we addressed a model of prior knowledge popularized by Valiant [1984]: we assume the learner knows only that the target concept c belongs to some class C, but nothing is known about the domain distribution P_X beforehand. Given this form of prior knowledge it is natural to consider the worst case performance guarantees we can make over all possible situations permitted by our prior constraints.

Sequential pac-learning

The first part of this thesis (Chapter 2) considered the problem of probably approximately correct (pac) learning; *i.e.*, returning a hypothesis with guaranteed accuracy and reliability after observing some finite number of training examples. Here, given a prior concept class C, a specified accuracy level $1 - \epsilon$, and a specified reliability level $1 - \delta$, we demand that the learner produce a hypothesis of accuracy at least $1 - \epsilon$ with probability at least $1 - \delta$, regardless of the particular target concept in C used to label the training examples and the domain distribution used to generate domain objects.

Previous research has investigated the minimum data and computational resources needed to solve this problem as a function of C, ϵ, and δ. Most of these investigations consider a simplistic fixed-sample-

[1] We also restricted our attention to two-class classification problems (*i.e.*, learning a concept $c : X \to \{0,1\}$) and made the simplifying assumption that all generated examples were consistent with some underlying target concept (*i.e.*, noise-free classification).

size learning strategy, Procedure F, that first observes a large collection of training examples and returns an arbitrary concept from C that correctly classifies every training example. The data-efficiency of this procedure scales-up near-optimally in terms of ϵ, δ, and the Vapnik-Chervonenkis dimension of C. Despite this optimal scaling however, the actual training sample sizes this procedure demands are far too large to be practical in real applications.

To overcome this problem we considered an alternative approach—sequential pac-learning—where we observe training examples one-at-a-time and decide on-line whether to halt and return a hypothesis or continue training. Specifically, we introduced a novel pac-learning procedure S that observes a sequence of hypotheses $h_0, h_1, ...$ produced by a consistent hypothesizer for C, and uses a sequential probability ratio test to detect any accurate hypotheses in this list. We noted that this sequential procedure observes a random rather than fixed number of training examples, so we compared S's maximum expected sample size to the fixed sample size required by F to solve the same pac-learning problem.

A theoretical analysis showed that S's expected sample size scales the same as F's up to constant factors, and beats the previous bounds for small values of δ. But we also saw that there are inherent limits to the data-efficiency of sequential learning: a lower bound result showed that no sequential learner can beat the fixed-sample-size lower bound by more than a constant factor. However, training data is often the critical resource in practical applications, so even constant improvements can be important. In this regard we observed that Procedure S actually uses many times fewer training examples than F in practice even while obtaining the exact same worst case pac-guarantees. Moreover, this empirical advantage appears robust to changes in target concept, domain distribution, and even concept class (for a given VCdimension); thus, countering any claim that S's advantage is solely due to exploiting easy learning situations.

Finally, as an aside, we showed how Procedure S could also convert mistake bounded hypothesizers to data-efficient pac-learners; simplifying and improving on an earlier result of Littlestone [1989]. We also showed how S could be applied to arbitrary hypothesizers, to obtain pac-learning under a wider variety of situations than fixed-sample-size learning.

Distribution-specific sequential pac-learning

Next, in Chapter 3 we considered a different model of prior knowledge where the learner knows the domain distribution that is used to generate domain objects, but does not know which target concept from a prior class C is being used to label the examples. Here we considered the same pac-learning problem as before: demanding that the learner return a hypothesis with error less than ϵ with probability at least $1 - \delta$ for any target concept in C. Part of the motivation for considering this distribution-specific (d.s.) model is the presumption that the distribution-free (d.f.) pac-learning bounds are perceived as too pessimistic in that they must account for every possible domain distribution. As in the d.f. case, previous work on d.s. pac-learning has only considered fixed-sample-size learning procedures, but clearly there is nothing to prevent us from taking a sequential approach here. In fact, in Chapter 3 we saw that much stronger theoretical improvements could be obtained in this case.

The first thing we observed about d.s. pac-learning was that a fixed domain distribution automatically imposes a natural metric over the concept class C. From this perspective, Benedek and Itai [1988a] derive a universal d.s. pac-learning strategy, Procedure BI, that first constructs a finite cover of the concept space, and then identifies a near-optimal concept in this cover—using a fixed-sample-size approach to estimate inter-concept distances. In this chapter we showed how using a *sequential* test to identify accurate cover-concepts could substantially improve the data-efficiency of BI (by a factor of five).

We then investigated a more sophisticated multiresolution learning strategy, Sfoc, that first constructs a coarse cover of the concept space, and then searches the local neighborhood of the best concept in this cover, *etc.*; gradually refining the search to small local neighborhoods of the target concept. This procedure improves the data-efficiency of BI by more than constant factors, for concept spaces that are uniformly dense across local neighborhoods.

Next, as an aside, we observed how sequential procedures are able to learn with certainty in the d.s. setting (*i.e.*, return an ϵ-accurate hypothesis with probability 1), not just high probability. We referred to this form of learning as certainly approximately correct (cac) learning, and showed that cac-learning was impossible to achieve with fixed-sample-size learning and under the d.f. model. We then derived a sequential learning strategy for the d.s. case, Scov, that cac-learns a wide variety of concept spaces by first, observing the set of consistent concepts remaining in C, and then halting as soon as this set can be covered by a single hypothesis. We showed that this procedure `Scov` is a universal learner in the sense that it cac-learns every concept space where this is possible in principle. Moreover, we showed that `Scov` cac-learns with *optimal* data-efficiency. Although we were unable to derive a tight characterization of `Scov`'s data-efficiency in terms of any simple parameter of concept space complexity, we were able to demonstrate its extreme data-efficiency in a series of concrete examples.

Finally, returning to the standard *pac*-learning model, we pointed out how `Scov` could be applied to learning with high probability rather than certainty. Surprisingly, `Scov` turned out to be far more data-efficient than any of the previous pac-learning strategies (`BI`, `Sbi`, and `Sfoc`), even though `Scov` attains a higher level of reliability. Although `Scov` is not a universal pac-learning strategy, it appears to be strongly advantageous for natural spaces. The primary drawback of this procedure is that it is only computationally feasible for very simple spaces.

Learning curves

Finally, in the last part of this thesis (Chapter 4) we turned our attention to a different aspect of learning performance: investigating the average error of a learner's hypotheses as a function of training sample size t, rather than demand the learner to return an accurate concept with some pre-specified reliability. We referred to this as the *learning curve* of the hypothesizer. Clearly, the rate at which a learner's curve converges to zero error is determined by the prior knowledge it has about the target concept and domain distribution. We analyzed this question under the d.f. model considered in Chapter 2. Here Haussler, Littlestone and Warmuth [1988; 1994] have shown that the smallest expected error any hypothesizer can obtain after t training examples, in the worst case over all possible target concepts in C and all possible domain distributions, behaves as a rational $\Theta(t^{-1})$ function of t. However, in a series of experiments Cohn and Tesauro [1990; 1992] showed that *exponential* learning curves could be observed in many practical situations.

In Chapter 4 we observed that the worst case analysis of Haussler, Littlestone and Warmuth is *non-uniform* in training sample size. The point of this chapter was to show that this non-uniformity accounts for the discrepancy between the theoretical and experimental results. Specifically, we undertook a uniform analysis that keeps the domain distribution and target concept fixed for all training sample sizes, and investigated the best asymptotic form of convergence that can be achieved in the worst case over all possible target concepts in a class C and all possible domain distributions.

Our first results established the basic dichotomy between rational and exponential convergence by showing that finite concept classes always have exponential worst case learning curves, whereas continuous concept classes always force rational convergence in the worst case. These results corroborated the experimental findings of Cohn and Tesauro; predicting rational and exponential (worst case) learning curves in exactly the same circumstances. We then investigated the exact boundary between these convergence modes and established that rational versus exponential worst case learning curves (for simple concept chains) is determined by the presence or absence of any *dense* subchains in the original class. This showed that rational and exponential learning curves are the only two types of worst case convergence possible in the d.f. setting (for concept chains).

Next, we observed that every concept class considered by Cohn and Tesauro was fundamentally finite, and hence every learning curve they observed was asymptotically exponential. The reason they observed the dichotomy between rational and exponential learning curves is that they considered different relative scales between the training sample size and inter-concept distances.

Finally, we discussed the prospects for a general theory of worst case learning curves. The two important questions are *(i)* identifying the boundary between rational and exponential convergence, and *(ii)* deriving tight bounds on the specific convergence rates in terms of some simple parameter of the concept class (space) structure. Preliminary results were obtained for both the d.f. and d.s. models, however the prospects for a completely general theory remain distant.

5.2 Contributions

The overall point of this thesis was to demonstrate how the worst case theory of classification learning (developed by Valiant, Haussler, and others) could be made more relevant to practice. We did this by:

1. Deriving new learning strategies that improve the practical efficiency of previous techniques, while maintaining the same theoretical guarantees of correctness.
2. Deriving new theoretical explanations of widely observed empirical phenomena left unaccounted for by the previous theory.

Specifically, for the problem of d.f. pac-learning we presented a new learning technique that significantly improves the data-efficiency of previous approaches, while maintaining the exact same accuracy and reliability guarantees. This new procedure, S, embodies a generic hypothesis testing strategy that can be applied to arbitrary concept classes with little computational overhead. The practical implication of this technique is that it makes many new d.f. pac-learning problems realistically solvable for the first time.

For the related problem of d.s. pac-learning, we presented a number of new learning techniques that obtain significantly better data-efficiency than previous approaches, again maintaining the exact same reliability and accuracy guarantees as before. These procedures obtain better than constant reductions in special cases, and are orders of magnitude more efficient in natural problems.

Although the d.s. model may not seem very relevant to practice at first, these d.s. results might ultimately have a greater impact than the previous d.f. results. This is because we can always turn a d.f. learning problem into a d.s. problem simply by using *un*labelled training examples to estimate the inter-concept distances. So we should be able to use the extremely data-efficient d.s. learning procedures developed in Chapter 3 to solve d.f. pac-learning problems using very few *labelled* training examples. This appears to be an interesting area for future research. Overall, the results of this thesis show that sequential learning is a far more effective technique for minimizing training resources than fixed-sample-size learning; both theoretically (in terms of the ultimate efficiency levels that can be achieved), and pragmatically (in terms of the ease designing efficient learning procedures).

Finally, in this thesis we derived a precise theoretical explanation of the dichotomy between rational and exponential learning curves, which has often been observed in practice. The practical significance of this form of analysis is that it delineates intrinsically easy from hard learning situations.

5.3 Research directions

A number of specific directions for future research were identified throughout this thesis, including:

- achieving further improvements to the data-efficiency of pac-learning;
- pursuing a more serious investigation of the computational issues faced by a sequential learner;
- improving the data-efficiency (and learning curve) analyses to achieve tighter performance bounds, perhaps based on other structural parameters of concept class (space) complexity besides VCdimension, metric-entropy, or ϵ-reduction numbers, *etc.*;
- generalizing the d.s. pac (and learning curve) results to handle a wider range of natural problems.

Each of these items was discussed in detail, but rather than reiterate those discussions, I reconsider what a theory of classification learning should be doing in the first place.

The main thrust of this thesis was to show how the current (worst case) theory of classification learning could be made more relevant to practice. However, what we really want is effective classification learning in practice. That is, we want practice to benefit directly from theory, not the other way around. Although pac-learning theory has providing a deeper understanding of the factors that affect learning performance, this has yet to result in more useful tools for applied machine learning. Most of the techniques currently used in applied machine learning are *ad hoc*, and there are few examples of theoretically derived techniques in widespread practical use. The security of mathematically proven guarantees is nice, but practitioners want tools that actually *work* for them.

Desirable mathematical properties

This suggests that the key for making the theory more useful and relevant is to first ask what mathematical properties of a learning system would yield desirable behavior in practice. For example, pac-learning (where we fix an accuracy and reliability level, and then determine a training sample size sufficient to achieve these levels) might not be the most natural model of applied learning situations. It might be more natural to consider a fixed training sample size, and perhaps even a fixed confidence level, and then determine the smallest error level that can be achieved.

Another natural question is (d.s.) model selection: given a domain distribution P, a nested sequence of hypothesis classes $C_1 \subset C_2 \subset \ldots$, and a fixed training sample size t, how do we decide which class to choose a hypothesis from? Perhaps such a selection procedure could be based on a principle of minimizing the worst case expected error of the procedure.

Models of prior knowledge

It also makes sense to investigate other models of prior knowledge. This thesis concentrated on a worst case model where the target concept and domain distribution were assumed only to belong to general classes, and the goal was to account for every possibility by minimizing the worst case outcome. Another useful model of prior knowledge is to encode *quantitative* preferences or vague knowledge by placing a probability distribution over possible target concepts (and domain distributions); *e.g.*, a Bayesian approach. The benefits of a prior is that it permits us to express quantitative preferences in a simple and well-motivated manner, and automatically implements a principled trade-off between the "complexity" of the hypothesis and its "fit" to the training data. In fact, a Bayesian approach seems to give the best results in data-limited practical learning problems [Hinton, 1995]. The main drawbacks of the Bayesian approach are *(i)* that it demands complete prior knowledge (in the sense that one always has to specify a single, complete distribution over all possibilities), and *(ii)* implementing Bayesian inference is computationally demanding, even in simple cases.

A less demanding way to express prior preferences is to specify a qualitative preference hierarchy $C_1 \subset C_2 \subset \ldots$, *etc.* Of course, this representation introduces the classical "complexity versus fit" problem (also known as the bias/variance dilemma [Geman, Bienenstock and Doursat, 1992]). Since the preferences are incommensurable with empirical data-fit, there is no obvious way to trade-off these two aspects. This was the motivation for the second proposal above: to base any such trade-off, not on prior "universal" principles, but on desirable, quantitative performance characteristics for any model selection strategy.

The overall point here is that there is little need to be dogmatic about the type of prior information or constraints we can express about a learning problem. We just need languages that allow us to express naturally all prior information/constraints we have, without forcing us to express more than we know (or are willing to state) *a priori*. The best approach to any applied learning problem is to state all and only what one knows about the domain beforehand, and let the data do the rest. Different forms of prior knowledge will likely be available in different applications. It is important to have a sound mathematical understanding of the consequences on learning performance of adopting each such form.

Appendix A

Technical details: Chapter 2

This appendix addresses the technical issues raised in Chapter 2. First, since the learning techniques investigated in Chapter 2 were based on the sequential probability ratio test (`sprt`) [Wald, 1947], we first discuss this procedure in some detail in Section A.1; describing the key properties that will be needed in the subsequent analysis. After these preliminaries Section A.2 then presents complete proofs of all (original) results stated in Chapter 2 and Section A.3 summarizes some algebraic bounds used in these proofs.

A.1 Sequential probability ratio testing

The sequential learning strategies developed in Chapter 2 made use of a sequential probability ratio test (`sprt`) [Wald, 1947] to test the error rates of their hypotheses. This procedure (Figure 2.4) tests the probability that a boolean random variable $\phi : X \to \{0,1\}$ takes on the value 1 with a probability less than a or greater than r, $a < r$. In particular, `sprt` tests

$$H_{acc} : \mathrm{P}_X\{\phi(x) = 1\} \leq a \qquad \text{versus} \qquad H_{acc} : \mathrm{P}_X\{\phi(x) = 1\} \geq r,$$

with a probability of incorrectly deciding H_{acc} when H_{rej} is true bounded by δ_{acc}, and a probability of incorrectly deciding H_{rej} when H_{acc} is true bounded by δ_{rej} (we do not care what happens when the probability is between a and r). For the analyses in Section A.2 below we first need to establish a number of key properties of this `sprt` procedure.

Definitions and notation: Notice that from a fixed distribution P_X on X, a boolean random variable $\phi : X \to \{0,1\}$ induces a distribution $\mathrm{P}_{\{0,1\}}$ on $\{0,1\}$. To analyze `sprt` we need to consider the sample space that consists of all observation sequences of unbounded length; *i.e.*, $\langle \phi_1, \phi_2, ...\rangle \in \{0,1\}^\infty$. Since we are assuming i.i.d. observations, from a fixed distribution $\mathrm{P}_{\{0,1\}}$ on $\{0,1\}$, the probability of an event $A \subset \{0,1\}^\infty$ is determined by the product distribution $\mathrm{P}_{\{0,1\}^\infty} = \mathrm{P}_{\{0,1\}} \times \mathrm{P}_{\{0,1\}} \times ...$ on $\{0,1\}^\infty$. To simplify the presentation, we drop the subscript $\{0,1\}$ from $\mathrm{P}_{\{0,1\}}$ and let P^t denote $\mathrm{P}_{\{0,1\}^t}$, and P^∞ denote $\mathrm{P}_{\{0,1\}^\infty}$. We also let P_p stand for the probability measure for which $\mathrm{P}\{1\} = p$, and let E_p stand for the expectation operator with respect to this distribution. The notation $\boldsymbol{\phi}^t$ will refer to the initial t-segment of some unbounded observation sequence $\boldsymbol{\phi} \in \{0,1\}^\infty$.

Here, a stopping rule $T : \{0,1\}^\infty \to I\!N$ is defined as a mapping from unbounded observation sequences to stopping times, such that the event $\{\boldsymbol{\phi} \in \{0,1\}^\infty : T(\boldsymbol{\phi}) = t\}$ depends only on the first t training examples. For a stopping rule T, we let $\boldsymbol{\phi}^T$ refer to the initial $T(\boldsymbol{\phi})$-segment of $\boldsymbol{\phi} \in \{0,1\}^\infty$.

Analysis of Procedure `sprt`

From Figure 2.4 we can see that Procedure `sprt` monitors the likelihood ratio $R_t(\boldsymbol{\phi}^t) = \mathrm{P}_a^t\{\boldsymbol{\phi}^t\}/\mathrm{P}_r^t\{\boldsymbol{\phi}^t\}$ and decides "H_{acc}" whenever $\mathrm{P}_r\{\boldsymbol{\phi}^t\} \leq \delta_{acc}\mathrm{P}_a\{\boldsymbol{\phi}^t\}$, and "$H_{rej}$" whenever $\mathrm{P}_a\{\boldsymbol{\phi}^t\} \leq \delta_{rej}\mathrm{P}_r\{\boldsymbol{\phi}^t\}$. The basic idea is to decide "H_{acc}" as soon as the probability of the observed sequence under H_{rej} drops below δ_{acc} times its probability under H_{acc}. It is not hard to show that this strategy meets the stated reliability criteria.

Lemma A.1 [Wald, 1947] *For any boolean random variable $\phi : X \to \{0,1\}$, any $0 \le a < r \le 1$, and any $\delta_{acc} \ge 0$ and $\delta_{rej} \ge 0$: The probability that* $\texttt{sprt}(\phi(x), a, r, \delta_{acc}, \delta_{rej})$ *incorrectly decides H_{acc} when in fact H_{rej} is true, is bounded by δ_{acc}. (Similarly, the probability that* `sprt` *incorrectly decides H_{rej} when H_{acc} is true is bounded by δ_{rej}.)*

Proof (This proof is due to Wald [1947]. I include it here because it clearly illustrates the behavior of `sprt`.) We first want to show that, assuming H_{rej} is true, the probability that `sprt` decides H_{acc} is bounded by δ_{acc}; *i.e.*, $\mathrm{P}_r^\infty\{\texttt{sprt} \text{ decides } H_{acc}\} \le \delta_{acc}$. To this end, let A^* be the set of all finite observation sequences $\boldsymbol{\alpha}^t \in \{0,1\}^*$ such that `sprt` decides H_{acc} at the end of the sequence, without making any decision beforehand. Then

$$\{\boldsymbol{\phi} \in \{0,1\}^\infty : \texttt{sprt} \text{ decides } H_{acc}\} \;=\; \bigcup_{\boldsymbol{\alpha}^t \in A^*} \{\boldsymbol{\phi} : \boldsymbol{\phi}^t = \boldsymbol{\alpha}^t\}\,. \tag{A.1}$$

By construction, for each $\boldsymbol{\alpha}^t \in A^*$ we have $\mathrm{P}_r^t\{\boldsymbol{\alpha}^t\} \le \delta_{acc}\,\mathrm{P}_a^t\{\boldsymbol{\alpha}^t\}$, or equivalently

$$\mathrm{P}_r^\infty\left\{\boldsymbol{\phi} : \boldsymbol{\phi}^t = \boldsymbol{\alpha}^t\right\} \;\le\; \delta_{acc}\,\mathrm{P}_a^\infty\left\{\boldsymbol{\phi} : \boldsymbol{\phi}^t = \boldsymbol{\alpha}^t\right\}.$$

Since this inequality holds for each $\boldsymbol{\alpha}^t \in A^*$, it must also hold for the (disjoint) union:

$$\mathrm{P}_r^\infty \bigcup_{\boldsymbol{\alpha}^t \in A^*} \left\{\boldsymbol{\phi} : \boldsymbol{\phi}^t = \boldsymbol{\alpha}^t\right\} \;\le\; \delta_{acc}\,\mathrm{P}_a^\infty \bigcup_{\boldsymbol{\alpha}^t \in A^*} \left\{\boldsymbol{\phi} : \boldsymbol{\phi}^t = \boldsymbol{\alpha}^t\right\}. \tag{A.2}$$

Finally, since P_a^∞ is a probability measure over $\{0,1\}^\infty$, from (A.1) and (A.2) we get

$$\begin{aligned}\mathrm{P}_r^\infty\{\boldsymbol{\phi} : \texttt{sprt} \text{ decides } H_{acc}\} &\le \delta_{acc}\,\mathrm{P}_a^\infty\{\boldsymbol{\phi} : \texttt{sprt} \text{ decides } H_{acc}\}\\ &\le \delta_{acc}.\end{aligned}$$

(The proof that $\mathrm{P}_a^\infty\{\texttt{sprt} \text{ decides } H_{rej}\} \le \delta_{rej}$ is symmetrical.) ∎

The sequential learning procedures developed in Chapter 2 call this `sprt` procedure to test whether the error of a hypothesis h was less than ϵ/κ, or greater than ϵ, for some $\kappa > 1$. In particular, they call $\texttt{sprt}(\ h(x) \ne c(x),\ \epsilon/\kappa,\ \epsilon,\ \delta,\ 0\ )$, which accepts an ϵ-bad hypothesis with probability at most δ, but never rejects any hypothesis (since setting $\delta_{rej} = 0$ means `sprt` never decides "H_{rej}"). A key fact is that such a call to `sprt` is guaranteed to accept any ϵ/κ-good hypothesis wp1.

Lemma A.2 *For any $\epsilon > 0$, $\delta > 0$, $\kappa > 1$, and any boolean random variable ϕ such that $\mathrm{P}_X\{\phi(x) = 1\} \le \epsilon/\kappa$: Calling* $\texttt{sprt}(\ \phi(x),\ \epsilon/\kappa,\ \epsilon,\ \delta,\ 0\ )$ *returns "H_{acc}" wp1.*

Proof First notice that since $\delta_{rej} = 0$, `sprt` never rejects a hypothesis. Therefore, it remains only to show that `sprt` terminates wp1 for a boolean random variable ϕ with $\mathrm{P}_X\{\phi(x) = 1\} \le \epsilon/\kappa$. (Notice that this induces a distribution P_p on $\{0,1\}$ where $p \le \epsilon/\kappa$.) We will use the fact that the log-likelihood S_t monitored by `sprt` is actually an i.i.d. sum:

$$S_t(\boldsymbol{\phi}^t) \;=\; \sum_{\phi_i \in \boldsymbol{\phi}^t} Z(\phi_i),$$

where

$$Z(\phi_i) = \begin{cases} \ln \frac{1-\epsilon/\kappa}{1-\epsilon}, & \phi_i = 0,\\ -\ln \kappa, & \phi_i = 1. \end{cases} \tag{A.3}$$

Since S_t is an i.i.d. sum we have $S_t/t \to \mathrm{E}\,Z$ wp1 by the strong law of large numbers [Ash, 1972, Theorem 7.2.5]. Claim A.3 below proves that $\mathrm{E}_p Z > 0$ for any $p \le \epsilon/\kappa$ where $\epsilon > 0$ and $\kappa > 1$, and therefore $S_t \to \infty$ wp1 under these conditions. Finally, since `sprt` terminates whenever $S_t(\boldsymbol{\phi}^t) \ge \ln(1/\delta_{acc})$, this means that `sprt` terminates wp1 for any $\delta_{acc} > 0$. ∎

Claim A.3 *For $\epsilon > 0$, $\kappa > 1$, and $p \leq \epsilon/\kappa$:*

$$\mathrm{E}_p Z \;\geq\; \left(\frac{\kappa - 1 - \ln\kappa}{\kappa}\right)\epsilon.$$

Proof For a distribution P_p on $\{0,1\}$ (*i.e.*, such that $\mathrm{P}\{1\} = p$) we have

$$\mathrm{E}_p Z \;=\; (1-p)\ln\frac{1-\epsilon/\kappa}{1-\epsilon} - p\ln\kappa, \tag{A.4}$$

by the definition of Z given in (A.3). Since this quantity is increasing for decreasing p, it suffices to derive a lower bound on $\mathrm{E}_p Z$ at $p = \epsilon/\kappa$. Fixing κ and thinking of $\mathrm{E}_{\epsilon/\kappa} Z$ as a function of ϵ, we can lower bound this quantity by a linear function of ϵ as follows. Plugging $p = \epsilon/k$ into (A.4) and rearranging gives

$$\mathrm{E}_{\epsilon/\kappa} Z \;=\; \left(1 - \frac{\epsilon}{\kappa}\right)\ln\frac{\kappa-\epsilon}{1-\epsilon} - \ln\kappa.$$

This quantity is 0 at $\epsilon = 0$ (*i.e.*, $\mathrm{E}_0 Z = 0$), so we lower bound it by a linear function $\alpha\epsilon$ for a constant α. To determine this bound we take the derivative of $\mathrm{E}_{\epsilon/\kappa} Z$ with respect to ϵ

$$\frac{d}{d\epsilon}\mathrm{E}_{\epsilon/\kappa} Z \;=\; \frac{1}{\kappa}\left(\frac{\kappa-1}{1-\epsilon} - \ln\frac{\kappa-\epsilon}{1-\epsilon}\right).$$

The constant α is given by the value of this derivative at $\epsilon = 0$,

$$\left.\frac{d}{d\epsilon}\mathrm{E}_{\epsilon/\kappa} Z\right|_{\epsilon=0} \;=\; \frac{\kappa - 1 - \ln\kappa}{\kappa}.$$

Notice that this quantity is strictly positive for $\kappa > 1$, and also that the second derivative

$$\frac{d^2}{d\epsilon^2}\mathrm{E}_{\epsilon/\kappa} Z \;=\; \frac{(\kappa-1)^2}{\kappa(\kappa-\epsilon)(1-\epsilon)^2}$$

is strictly positive for $\kappa > 1 \geq \epsilon$. This means that

$$\mathrm{E}_{\epsilon/\kappa} Z \;\geq\; \left(\frac{\kappa - 1 - \ln\kappa}{\kappa}\right)\epsilon,$$

for all $\epsilon \geq 0$, $\kappa > 1$. ▮

A consequence of these results is that `sprt(` $h(x) \neq c(x)$, ϵ/κ, ϵ, δ, 0 `)` is guaranteed to accept any ϵ/κ-good hypothesis h wp1. However, not only is this call to `sprt` guaranteed to eventually accept any ϵ/κ-good hypothesis, it does so quickly—*i.e.*, with a small expected sample size. In fact, it is well known that `sprt` meets the specified accuracy and reliability criteria with *optimal* expected sample size [Chernoff, 1972].[1] Here we derive a reasonable upper bound on the expected number of observations `sprt` makes before accepting an ϵ/κ-good hypothesis.

Lemma A.4 *For any $0 < \epsilon < 5/8$, $\delta > 0$, $\kappa > 1$, and any boolean random variable ϕ such that $\mathrm{P}_X\{\phi(x) = 1\} \leq \epsilon/\kappa$:*

$$\mathrm{E}\, T_{\mathtt{sprt}(\,\phi(x),\,\epsilon/\kappa,\,\epsilon,\,\delta,\,0\,)} \;\leq\; \left(\frac{\kappa}{\kappa - 1 - \ln\kappa}\right)\frac{1}{\epsilon}\left(\ln\frac{1}{\delta} + 1\right).$$

[1] The general optimality result uses cutoff boundaries that are a bit are a bit smaller than δ_{rej} and $1/\delta_{acc}$, however these boundaries coincide with the ones used here for the special case of $\delta_{rej} = 0$.

Proof Recall from (A.3) that S_t is a sum of i.i.d. random variables $Z_1, ..., Z_t$. Wald's identity then says that $\mathrm{E}\, S_T = \mathrm{E}\, Z \times \mathrm{E}\, T$ for any stopping rule T [Wald, 1947, §3.5; Shiryayev, 1978, pp.175], which obviously means

$$\mathrm{E}\, T \;=\; \frac{\mathrm{E}\, S_T}{\mathrm{E}\, Z}. \tag{A.5}$$

I.e., the expected time to halt equals the expected sum at termination divided by the average step size. Now consider the stopping variable $T = T_{\mathtt{sprt}(\,\phi(x),\, \epsilon/\kappa,\, \epsilon,\, \delta,\, 0\,)}$. By construction `sprt` terminates whenever $S_t \geq \ln(1/\delta)$, so we know the sum at termination can be at most

$$S_T \;<\; \ln\frac{1}{\delta} + \ln\frac{1-\epsilon/\kappa}{1-\epsilon}.$$

I.e., the final sum must be less than the threshold plus one positive increment. Combined with (A.5) this gives

$$\mathrm{E}\, T \;<\; \frac{1}{\mathrm{E}\, Z}\left(\ln\frac{1}{\delta} + 1\right) \tag{A.6}$$

for $0 \leq \epsilon \leq 5/8$ (since $\ln[(1-\epsilon/\kappa)/(1-\epsilon)] < 1$ for all $0 \leq \epsilon \leq 1 - e^{-1}$ and $\kappa > 1$).

Notice that inequality (A.6) holds for an arbitrary P_p defined on $\{0,1\}$. We now derive an upper bound on $\mathrm{E}_p T$ under the assumption that $\mathrm{P}_X\{\phi(x) = 1\} \leq \epsilon/\kappa$. Claim A.3 above shows that if $p \leq \epsilon/\kappa$ then

$$\mathrm{E}_p Z \;\geq\; \left(\frac{\kappa - 1 - \ln\kappa}{\kappa}\right)\epsilon$$

for all $\epsilon > 0$, $\kappa > 1$. Therefore, plugging this inequality into (A.6) we obtain

$$\mathrm{E}_p T \;<\; \left(\frac{\kappa}{\kappa - 1 - \ln\kappa}\right)\frac{1}{\epsilon}\left(\ln\frac{1}{\delta} + 1\right)$$

for $p \leq \epsilon/\kappa$, $\epsilon > 0$, $\kappa > 1$. ∎

The next section applies these results to prove the correctness and efficiency of the sequential learning procedures developed in Chapter 2.

A.2 Proofs of results

Definitions and notation: To simplify the notation, for a target concept c and object sequence $\mathbf{x}^t = \langle x_1, x_2, ..., x_t\rangle$, we denote the resulting training sequence by $c\mathbf{x}^t \triangleq \langle\langle x_1, c(x_1)\rangle, \langle x_2, c(x_2)\rangle, ..., \langle x_t, c(x_t)\rangle\rangle$. The analyses presented in this section consider the behavior of sequential learning procedures on training sequences of unbounded length. Therefore, we will generally be considering the sample space $(X \times \{0,1\})^\infty$ consisting of all unbounded training sequences $c\mathbf{x} = \langle\langle x_1, c(x_1)\rangle, \langle x_2, c(x_2)\rangle, ...\rangle$. In particular, for a target concept $c : X \to \{0,1\}$, we will be concerned with events $A \subset X^\infty$. Since we are assuming i.i.d. observations, the probability of an event $A \subset X^\infty$ will be determined by the product distribution $\mathrm{P}_{X^\infty} = \mathrm{P}_X \times \mathrm{P}_X \times ...$ defined on X^∞.

Recall that a stopping rule $T : (X \times \{0,1\})^\infty \to I\!\!N$ is a mapping from unbounded training sequences to stopping times, such that the event $\{c\mathbf{x} : T(c\mathbf{x}) = t\}$ depends only on the first t training examples. For a fixed target concept $c : X \to \{0,1\}$, this stopping rule T becomes a mapping from sequences of domain objects to stopping times (*i.e.*, $T : X^\infty \to I\!\!N$), where again the event $\{\mathbf{x} : T(\mathbf{x}) = t\}$ depends only on the first t examples. For a stopping rule T, we let $\mathbf{x}^T$ refer to the initial $T(c\mathbf{x})$-segment of $\mathbf{x} \in X^\infty$ (where c is usually understood from context).

Theorem 2.9 (Correctness) *For any $\epsilon > 0$, $\delta > 0$, and any (well behaved) concept class C with* $\mathrm{VC}(C) < \infty$*: Given i.i.d. examples generated by any distribution* P_X *and target concept $c \in C$, Procedure* `S` *(*`R`*) returns a hypothesis h such that $\mathrm{P}_X\{h(x) \neq c(x)\} \leq \epsilon$ with probability at least $1 - \delta$; using any hypothesizer H that is consistent for C.*

Proof First, we show that if Procedure `S` terminates wp1, then it must satisfy the theorem. Lemma A.1 above shows that `S`'s call to `sprt` accepts an ϵ-bad hypothesis h_i with probability at most δ_i. Therefore, the total probability that `S` accepts some ϵ-bad hypothesis is bounded by $\sum_{i=1}^{\infty} \delta_i = \delta$. Clearly, if `S` returns some hypothesis wp1, but returns an ϵ-bad hypothesis with probability at most δ, then `S` must return an ϵ-*good* hypothesis with probability at least $1-\delta$. (The same argument also works for `R`.)

It remains only to show that `S` terminates wp1. The proof of this relies on two key facts: first, `S`'s call to `sprt` eventually accepts any ϵ/κ-good hypothesis wp1; and second, `S`'s hypothesizer H eventually produces such a hypothesis wp1. Together, these two facts ensure `S` terminates wp1. The first fact follows from by Lemma A.2 above. To establish the second fact we argue as follows: Since C has finite VCdimension, Lemma A.5 below shows that, wp1, there must be some time t by which every ϵ/κ-bad concept in C has misclassified some training example. At this time, either H has already produced a hypothesis that correctly classifies every observed example, or if not, it will be called to do so. In either case, H will have produced an ϵ/κ-good hypothesis.

(Termination wp1 for `R` can be proved by showing that once every ϵ-bad hypothesis has been eliminated from C, `R` must eventually accept some ϵ-good hypothesis wp1.) ▮

Lemma A.5 *For any $\epsilon > 0$, and any concept class C with finite VCdimension: Given a sequence of i.i.d. training examples, every concept in C with error greater than ϵ eventually misclassifies some training example, wp1.*

Proof Fix an arbitrary $\epsilon > 0$ and an arbitrary distribution P over training examples. Let $A_t \subset (X \times \{0,1\})^{\infty}$ be the event that all ϵ-bad concepts have been eliminated from C after the first t training examples. We are interested in the event $A_{\infty} = \bigcup_{t=1}^{\infty} A_t$, that all ϵ-bad concepts have been eliminated from C within some finite number of training examples. Theorem 2.5 [Shawe-Taylor, Anthony and Biggs, 1993] shows that for every $\delta > 0$ there is a t $(= T_{STAB}(C, \epsilon, \delta))$ such that $\mathrm{P}A_s \geq 1-\delta$ for all $s \geq t$. Thus $\mathrm{P}A_t \uparrow 1$, and since $A_t \uparrow A_{\infty}$ we also have $\mathrm{P}A_t \uparrow \mathrm{P}A_{\infty}$ [Ash, 1972, Theorem 1.2.7], and hence $\mathrm{P}A_{\infty} = 1$ as desired. ▮

Theorem 2.11 (Data-efficiency) *For any $\delta > 0$, sufficiently small $\epsilon > 0$, and any (well behaved) concept class C with finite VCdimension: Given i.i.d. examples generated by any distribution P_X and target concept $c \in C$, Procedure `S` observes an average training sample size of at most*

$$\mathrm{E}\, T_{\mathtt{S}}(C, \epsilon, \delta) \;\leq\; \left(\frac{\kappa}{\kappa - 1 - \ln\kappa}\right) \frac{1}{\epsilon} \left([2.12\, \kappa\, \mathrm{vc}(C) + 3] \ln \frac{14\kappa}{\epsilon} + \ln\frac{1}{\delta} \right); \tag{2.2}$$

using any hypothesizer H that produces consistent concepts from C, any constant $\kappa > 1$, and the sequence $\{\delta_i = 6\delta/(\pi^2 i^2)\}_{i=1}^{\infty}$ (which gives $\sum_{i=1}^{\infty} \delta_i = \delta$).

Proof The proof of this result relies on the same two facts used in Theorem 2.9 above: first, the call to `sprt` eventually accepts any ϵ/κ-good hypothesis wp1; and second, the hypothesizer H eventually produces such a hypothesis wp1. Together, these two facts show that we can bound the expected time for `S` to terminate by the time for H to produce an ϵ/κ-good hypothesis plus the time for `sprt` to accept such a hypothesis.

Fix arbitrary $\epsilon > 0$, $\delta > 0$, $\kappa > 1$, and choose an arbitrary target concept $c \in C$ and domain distribution P_X. Let $T_C(\epsilon)$ denote the time at which every ϵ-bad concept has been eliminated from C, and let $T_{\mathtt{sprt}}(\epsilon, \kappa, \delta)$ denote the time for `sprt(`$h(x) \neq c(x)$, ϵ/κ, ϵ, δ, 0`)` to accept an ϵ/κ-good hypothesis. Then, as argued in Theorem 2.9 above, once $T_C(\epsilon/\kappa)$ has been reached, `S`'s current hypothesis h_i must be ϵ/κ-good, since it correctly classifies every observed example by construction. Therefore, we can bound `S`'s stopping time by

$$T_{\mathtt{S}}(\epsilon, \delta) \;\leq\; T_C(\epsilon/\kappa) + T_{\mathtt{sprt}}(\epsilon, \kappa, \delta_i),$$

where i is the index of the first ϵ/κ-good hypothesis produced by H. Notice that i here is a *random variable*, but we can always bound i by $T_C(\epsilon/\kappa)$, since no more than one hypothesis can be generated per training example. This means any sequence of training examples gives

$$T_{\mathtt{S}}(\epsilon, \delta) \;\leq\; T_C(\epsilon/\kappa) + T_{\mathtt{sprt}}(\epsilon, \kappa,\ \delta_{\,T_C(\epsilon/\kappa)}\ ).$$

Now, taking expectations and using $\mathrm{E}\,X = \mathrm{E}\,[\mathrm{E}\,[X|Y]]$, we get

$$\begin{aligned}
\mathrm{E}\,T_{\mathrm{S}}(\epsilon,\delta) &\leq \mathrm{E}\,\left[T_C(\epsilon/\kappa) + T_{\mathtt{sprt}}(\epsilon,\kappa,\ \delta_{T_C(\epsilon/\kappa)}\,)\right] \\
&= \mathrm{E}\,\left[\mathrm{E}\,\left[t + T_{\mathtt{sprt}}(\epsilon,\kappa,\ \delta_t\,)\middle|\,T_C(\epsilon/\kappa) = t\right]\right] \\
&= \mathrm{E}\,T_C(\epsilon/\kappa)\ +\ \mathrm{E}\,\left[\mathrm{E}\,\left[T_{\mathtt{sprt}}(\epsilon,\kappa,\delta_t\,)\middle|\,T_C(\epsilon/\kappa) = t\right]\right]. \qquad \text{(A.8)}
\end{aligned}$$

First we bound the inner expectation: Applying Lemma A.4 above and using the fact that $\delta_i = 6\delta/(\pi^2 i^2)$, we obtain

$$\begin{aligned}
\mathrm{E}\,\left[T_{\mathtt{sprt}}(\epsilon,\kappa,\delta_t)\middle|\,T_C(\epsilon/\kappa) = t\right] &\leq \frac{\kappa'}{\epsilon}\left(\ln\frac{1}{\delta_t} + 1\right) \qquad \text{where } \kappa' = \kappa/(\kappa - 1 - \ln\kappa), \\
&= \frac{\kappa'}{\epsilon}\left(2\ln t + \ln\frac{1}{\delta} + \ln\frac{\pi^2}{6} + 1\right), \qquad \text{(A.9)}
\end{aligned}$$

for $0 < \epsilon \leq 5/8$. So, combining (A.8) and (A.9), and noticing that $\ln(\pi^2/6) < 1/2$, we get

$$\begin{aligned}
\mathrm{E}\,T_{\mathrm{S}}(\epsilon,\delta) &\leq \mathrm{E}\,T_C(\epsilon/\kappa) + \mathrm{E}\,\left[\frac{\kappa'}{\epsilon}\left(2\ln T_C(\epsilon/\kappa) + \ln\frac{1}{\delta} + \frac{3}{2}\right)\right] \\
&= \mathrm{E}\,T_C(\epsilon/\kappa) + \frac{\kappa'}{\epsilon}\left(2\,\mathrm{E}\,[\ln T_C(\epsilon/\kappa)] + \ln\frac{1}{\delta} + \frac{3}{2}\right) \\
&\leq \mathrm{E}\,T_C(\epsilon/\kappa) + \frac{\kappa'}{\epsilon}\left(2\ln\mathrm{E}\,T_C(\epsilon/\kappa) + \ln\frac{1}{\delta} + \frac{3}{2}\right). \qquad \text{(A.10)}
\end{aligned}$$

The last bound follows from Jensen's inequality: since ln is a concave function we have $\mathrm{E}\,\ln X \leq \ln\mathrm{E}\,X$ for any random variable X [Ash, 1972, Chapter 7].

It remains only to find a bound on $\mathrm{E}\,T_C(\epsilon/\kappa)$. Let $d = \mathrm{VC}(C)$. Lemma A.6 below shows

$$\begin{aligned}
\mathrm{E}\,T_C(\epsilon/\kappa) &\leq \frac{\kappa''}{\epsilon}\left(2d\ln\frac{6\kappa}{\epsilon} + \ln 2 + 1\right) \qquad \text{where } \kappa'' = \kappa/(1 - \sqrt{\epsilon/\kappa}), \\
&\leq \frac{2\kappa'' d}{\epsilon}\ln\frac{14\kappa}{\epsilon}, \qquad \text{(A.11)}
\end{aligned}$$

for $d \geq 1$. The rest is just algebraic manipulation: Plugging (A.11) into (A.10) yields

$$\mathrm{E}\,T_{\mathrm{S}}(\epsilon,\delta) \leq \frac{\kappa'}{\epsilon}\left[2\frac{\kappa''}{\kappa'}d\ln\frac{14\kappa}{\epsilon} + 2\ln\left(\frac{2\kappa'' d}{\epsilon}\ln\frac{14\kappa}{\epsilon}\right) + \ln\frac{1}{\delta} + \frac{3}{2}\right].$$

Claim A.12 below shows that $\kappa''/\kappa' \leq 1.06\kappa$, and so we have

$$\begin{aligned}
\mathrm{E}\,T_{\mathrm{S}}(\epsilon,\delta) &\leq \frac{\kappa'}{\epsilon}\left[2.12\kappa d\ln\frac{14\kappa}{\epsilon} + 2\left(\ln\frac{2\kappa''}{\epsilon} + \ln\ln\frac{14\kappa}{\epsilon} + \ln d + \frac{3}{4}\right) + \ln\frac{1}{\delta}\right] \qquad \text{(A.12)} \\
&= O\left(\frac{1}{\epsilon}\left(d\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right).
\end{aligned}$$

This bound holds under the minor restriction that $\epsilon \leq 5/8$ (needed to apply Lemma A.4). If, however, we restrict attention to small values of ϵ this bound can be simplified further. Claim A.13 below shows that

$$\ln\frac{2\kappa''}{\epsilon} + \ln\ln\frac{14\kappa}{\epsilon} + \ln d + \frac{3}{4} \leq \frac{3}{2}\ln\frac{14\kappa}{\epsilon}$$

for $d \geq 2$ and $\epsilon \leq \min\{\,1/4,\ \kappa/[d^2(\ln 1.05d)^2]\,\}$. Plugging this into (A.12) yields

$$\mathrm{E}\,T_{\mathrm{S}}(\epsilon,\delta) \leq \frac{\kappa'}{\epsilon}\left((2.12\kappa d + 3)\ln\frac{14\kappa}{\epsilon} + \ln\frac{1}{\delta}\right). \ \blacksquare$$

Lemma A.6 *For any $\epsilon > 0$, and any concept class C with finite VCdimension: Given a sequence of i.i.d. training examples, the expected number of training examples before every ϵ-bad concept is eliminated from C is bounded by*

$$\mathrm{E}\, T_C(\epsilon) \;\le\; \frac{1}{\epsilon(1-\sqrt{\epsilon})}\left(2\mathrm{vc}(C)\ln\frac{6}{\epsilon} + \ln 2 + 1\right).$$

Proof We prove this using Theorem 2.5 [Shawe-Taylor, Anthony and Biggs, 1993], which showed that for every $\delta > 0$ we have $\mathrm{P}\left\{T_C(\epsilon) > T_{\scriptscriptstyle STAB}(C,\epsilon,\delta)\right\} \le \delta$, where $T_{\scriptscriptstyle STAB}(\epsilon,\delta)$ is a constant that depends on ϵ, δ, and $\mathrm{vc}(C)$. Let us assume, pessimistically, that $T_C(\epsilon)$ is a random variable that makes this an equality; *i.e.*,

$$\mathrm{P}\left\{T_C(\epsilon) > T_{\scriptscriptstyle STAB}(C,\epsilon,\delta)\right\} \;=\; \delta \tag{A.13}$$

for every $\delta > 0$. Now, consider a random variable V defined by the linear transformation

$$V \;=\; \epsilon(1-\sqrt{\epsilon})T_C(\epsilon) - 2\mathrm{vc}(C)\ln\frac{6}{\epsilon} - \ln 2.$$

Notice that $V > \ln(1/\delta)$ holds if and only if $T_C(\epsilon) > T_{\scriptscriptstyle STAB}(C,\epsilon,\delta)$. Therefore, we have $\mathrm{P}\{V > \ln(1/\delta)\} = \delta$ for all $\delta > 0$, and this means V has an $\mathsf{exponential}(1)$ distribution. Now, notice that $T_C(\epsilon)$ is related to V by the inverse linear transformation

$$T_C(\epsilon) = \frac{1}{\epsilon(1-\sqrt{\epsilon})}\left(2\mathrm{vc}(C)\ln\frac{6}{\epsilon} + \ln 2 + V\right).$$

Taking expectations gives the result, since $\mathrm{E}\,V = 1$. ∎

Proposition 2.12 (Comparison)
$\mathrm{E}\,T_{\mathsf{S}}(C,\epsilon,\delta) < T_{\scriptscriptstyle BEHW}(C,\epsilon,\delta)$ *for $\kappa \ge 3.5$ and sufficiently small $\delta = \epsilon^{\Theta(\mathrm{vc}(C))}$.*
$\mathrm{E}\,T_{\mathsf{S}}(C,\epsilon,\delta) < T_{\scriptscriptstyle STAB}(C,\epsilon,\delta)$ *for $\kappa \ge (2/\sqrt{\epsilon})\ln(2/\sqrt{\epsilon})$ and sufficiently small $\delta = \epsilon^{\Theta(\kappa\cdot\mathrm{vc}(C))}$.*

Proof This result simply exploits the fact that $\kappa' = \kappa/(\kappa - 1 - \ln\kappa)$ can be made arbitrarily close to 1 by choosing κ sufficiently large. This is important because reducing the multiplicative factor on the $\ln(1/\delta)$ term in the $\mathrm{E}\,T_{\mathsf{S}}$ bound creates a situation where δ can be chosen sufficiently small to ensure $\mathrm{E}\,T_{\mathsf{S}} < T_{\scriptscriptstyle BEHW}$ and $\mathrm{E}\,T_{\mathsf{S}} < T_{\scriptscriptstyle STAB}$. Let $d = \mathrm{vc}(C)$, and recall the bound (2.2) on $\mathrm{E}\,T_{\mathsf{S}}$

$$\mathrm{E}\,T_{\mathsf{S}}(C,\epsilon,\delta) \;\le\; \frac{\kappa'}{\epsilon}(2.12\,\kappa\,\mathrm{vc}(C)+3)\ln\frac{14\kappa}{\epsilon} \;+\; \frac{\kappa'}{\epsilon}\ln\frac{1}{\delta}.$$

First, for $T_{\scriptscriptstyle BEHW}$, recall

$$\begin{aligned} T_{\scriptscriptstyle BEHW} \;&=\; \max\left\{\frac{8d}{\epsilon}\log_2\frac{13}{\epsilon},\; \frac{4}{\epsilon}\log_2\frac{2}{\delta}\right\} \\ &>\; \frac{5.77d}{\epsilon}\ln\frac{13}{\epsilon} \;+\; \frac{2.88}{\epsilon}\ln\frac{2}{\delta}. \end{aligned}$$

Choosing $\kappa \ge 3.5$ gives $\kappa' < 2.88$, which permits us to choose a value of $\delta = \epsilon^{\Theta(d)}$ so that $\mathrm{E}\,T_{\mathsf{S}} < T_{\scriptscriptstyle BEHW}$. Next, for $T_{\scriptscriptstyle STAB}$, recall

$$T_{\scriptscriptstyle STAB}(C,\epsilon,\delta) \;=\; \frac{1}{\epsilon(1-\sqrt{\epsilon})}2d\ln\frac{6}{\epsilon} \;+\; \frac{1}{\epsilon(1-\sqrt{\epsilon})}\ln\frac{2}{\delta}.$$

Claim A.14 below shows that choosing $\kappa > (2/\sqrt{\epsilon})\ln(2/\sqrt{\epsilon})$ gives $\kappa' < 1/(1-\sqrt{\epsilon})$. This permits us to choose a value of $\delta = \epsilon^{\Theta(\kappa d)}$ for which $\mathrm{E}\,T_{\mathsf{S}} < T_{\scriptscriptstyle STAB}$. ∎

Theorem 2.14 (Data-complexity) *For any $0 < \epsilon \le 1/8$, $0 < \delta \le 1/683$, and any concept class C with $\text{VC}(C) \ge 2$: Any learner that always observes an average training sample size less than*

$$t_{avg}(C, \epsilon, \delta) \;=\; \max\left\{ \frac{\text{VC}(C) - 1}{480\epsilon}, \; \frac{1 - 2\delta}{2\epsilon} \right\}$$

for every fixed $c \in C$ and P_X will fail to meet the pac(ϵ, δ)-criterion for some target concept $c' \in C$ and domain distribution P'_X.

Proof Fix an arbitrary $0 < \epsilon \le 1/8$ and an arbitrary learner $L = (T, H)$. Let $H[c\mathbf{x}^t]$ denote H's hypothesis given training sequence $c\mathbf{x}^t$, and let $H[c\mathbf{x}^T]$ denote $H[c\mathbf{x}^t]$ for $t = T[c\mathbf{x}]$. Also let $err(H, \text{P}_X, c, \mathbf{x}^t)$ denote $\text{P}_X\{H[c\mathbf{x}^t](x) \ne c(x)\}$. We prove this theorem in two parts: first showing $t_{avg} \ge (\text{VC}(C) - 1)/(480\epsilon)$, and then $t_{avg} \ge (1 - 2\delta)/(2\epsilon)$.

Part 1: $t_{avg} \ge (\text{VC}(C) - 1)/(480\epsilon)$. Following Ehrenfeucht *et al.* [1989] we construct a hard domain distribution that forces bad worst case behavior. Let $d = \text{VC}(C) - 1$ and $X' = \{x_0, x_1, ..., x_d\}$ be a set of $d+1$ domain objects shattered by C (such a set must exist by the definition of VCdimension, *cf.* Definition 2.3). Define the distribution P'_X on X' by

$$\begin{aligned} \text{P}'_X\{x_0\} &= 1 - 8\epsilon \\ \text{P}'_X\{x_i\} &= \frac{8\epsilon}{d} \qquad \text{for } 1 \le i \le d. \end{aligned} \tag{A.14}$$

Now, for observation sequences $\mathbf{x} \in X^\infty$, let the random variable $U : X^\infty \to I\!N$ indicate the first time, t, when more than $d/2$ of the objects in $\{x_1, ..., x_d\}$ appear in the initial segment, $\mathbf{x}^t$, of $\mathbf{x}$. The basic idea is to show that if $\text{E}\,T$ is too small then T will often be smaller than U, and this will force H to miss the pac(ϵ, δ)-criterion for some target concept $c' \in C$.

For any H and T, and for any c, P_X, and t we have

$$\begin{aligned} \text{P}_{X^\infty}\{err(H, \text{P}_X, c, \mathbf{x}^T) > \epsilon\} &\ge \text{P}_{X^\infty}\{err(H, \text{P}_X, c, \mathbf{x}^T) > \epsilon,\, T[c\mathbf{x}] < U[\mathbf{x}]\} \\ &= \text{P}_{X^\infty}\left\{err(H, \text{P}_X, c, \mathbf{x}^T) > \epsilon \,\middle|\, Tc < U\right\} \quad \text{P}_{X^\infty}\{Tc < U\} \\ &\ge \text{P}_{X^\infty}\left\{err(H, \text{P}_X, c, \mathbf{x}^T) > \epsilon \,\middle|\, Tc < U\right\} \quad \text{P}_{X^\infty}\{Tc \le t < U\} \\ &\ge \text{P}_{X^\infty}\left\{err(H, \text{P}_X, c, \mathbf{x}^T) > \epsilon \,\middle|\, Tc < U\right\} \\ &\qquad \times [\text{P}_{X^\infty}\{Tc \le t\} + \text{P}_{X^\infty}\{U > t\} - 1] \end{aligned} \tag{A.15}$$

(the last inequality follows since $\text{P}AB = \text{P}A + \text{P}B - \text{P}A \cup B$). Below we find lower bounds for each of these terms.

1. Lemma A.7 shows that for any H there must be some $c' \in C$ for which

$$\text{P}'_{X^\infty}\left\{err(H, \text{P}'_X, c', \mathbf{x}^T) > \epsilon \,\middle|\, Tc' < U\right\} \;\ge\; \frac{1}{7}. \tag{A.16}$$

2. Ehrenfeucht *et al.* [1989, Lemma 3] show that for $t' = d/(32\epsilon)$

$$\text{P}'_{X^\infty}\{U > t'\} \;\ge\; 1 - e^{-1/12} \;>\; \frac{1}{13}. \tag{A.17}$$

3. Finally, by Markov's inequality we have that if $\text{E}\,Tc < t'/k$ for $k \ge 1$, then

$$\text{P}'_{X^\infty}\{Tc \le t'\} \;\ge\; 1 - \frac{1}{k}. \tag{A.18}$$

Combining (A.15), (A.16), (A.17), and (A.18) shows that if $\mathrm{E}\,Tc \le d/(32k\epsilon)$ for all $c \in C$, then for any hypothesizer H there must be some $c' \in C$ for which

$$\begin{aligned} \mathrm{P}'_{X^\infty}\{err(H, \mathrm{P}'_X, c', \mathbf{x}^T) > \epsilon\} &\ge \frac{1}{7}\left[1 - \frac{1}{k} + \frac{1}{13} - 1\right] \\ &= \frac{1}{7}\left[\frac{1}{13} - \frac{1}{k}\right]. \end{aligned}$$

Choosing $k = 15$ yields the result.

Part 2: $t_{avg} \ge (1 - 2\delta)/(2\epsilon)$. Again, we construct a hard domain distribution that forces bad worst case behavior. Since $\mathrm{VC}(C) \ge 2$, there must be two domain objects $\{x_0, x_1\}$ shattered by C. Define the distribution P'_X on $\{x_0, x_1\}$ by

$$\begin{aligned} \mathrm{P}'_X\{x_0\} &= 1 - 2\epsilon \\ \mathrm{P}'_X\{x_1\} &= 2\epsilon. \end{aligned}$$

Now, for observation sequences $\mathbf{x} \in X^\infty$, let $V : X^\infty \to \mathbb{N}$ indicate the first time that x_1 appears in $\mathbf{x}$. As before, we argue that if $\mathrm{E}\,T$ is too small then T will often be smaller than V, and this will force H to miss the pac(ϵ, δ)-criterion for some $c' \in C$. Restrict attention to the class of concepts C_0 that label x_0 as 0; *i.e.*, $C_0 \triangleq \{c \in C : c(x_0) = 0\}$. Here the event $\{Tc < V\}$ is identical for all $c \in C_0$ since it consists of all stopping sequences where only x_0 is observed, so we write this event as $\{T < V\}$.

For any H and T, and any $c \in C_0$ we have

$$\begin{aligned} \mathrm{P}'_{X^\infty}\{err(H, \mathrm{P}'_X, c, \mathbf{x}^T) > \epsilon\} &\ge \mathrm{P}'_{X^\infty}\left\{err(H, \mathrm{P}'_X, c, \mathbf{x}^T) > \epsilon,\ T < V\right\} \\ &= \mathrm{P}'_{X^\infty}\left\{err(H, \mathrm{P}'_X, c, \mathbf{x}^T) > \epsilon \;\middle|\; T < V\right\} \\ &\quad \times\ \mathrm{P}'_{X^\infty}\{T < V\}. \end{aligned} \tag{A.19}$$

First, to lower bound the conditional probability in (A.19), notice that if $\{T < V\}$ has occurred, then C_0 can be partitioned into two subclasses that are 2ϵ apart

$$\begin{aligned} C_{00} &= \{c \in C_0 : c(x_1) = 0\}, \\ C_{01} &= \{c \in C_0 : c(x_1) = 1\}. \end{aligned}$$

This means that for any training sequence $\mathbf{x} \in \{T < V\}$, any hypothesis $H[c\mathbf{x}^T]$ must be at least ϵ away from all of the concepts in one of C_{00} or C_{01}. Thus, for any (randomized) hypothesizer H there must exist some $c' \in C_0$ such that

$$\mathrm{P}'_{X^\infty}\left\{err(H, \mathrm{P}'_X, c', \mathbf{x}^T) > \epsilon \;\middle|\; T < V\right\} \ge \frac{1}{2}. \tag{A.20}$$

Now, to lower bound the probability term in (A.19) notice that $\{T < V\}$ means T halts beforex_1 is observed. *I.e.*, T halts on some initial sequence $\mathbf{x}_0^t = \langle x_0, x_0, \ldots\rangle$ that contains only x_0's.[2] In fact, if $\{T < V\}$ occurs at all, then there must be some *shortest* such initial sequence $\mathbf{x}_o^\tau$, so we write $\mathrm{P}'_{X^\infty}\{T < V\} = \mathrm{P}'_{X^\tau}\{x_o^\tau\}$. Now, if we assume that T stops as soon as x_1 is observed,[3] then this value of τ also determines T's expected stopping time as follows. Let $p_0 = \mathrm{P}'_X\{x_0\}$ and $\mathrm{P}'_X\{x_1\} = 1 - p_0$. Then

$$\begin{aligned} \mathrm{E}\,T &= \sum_{i=1}^{\tau-1} i p_0^{i-1}(1 - p_0) \ +\ \tau p_0^{\tau-1} \\ &= \sum_{i=1}^{\tau-1} i p_0^{i-1} \ -\ \sum_{i=1}^{\tau-1} i p_0^{i} \ +\ \tau p_0^{\tau-1} \end{aligned}$$

[2] Assuming a deterministic stopping rule T. I have yet to generalize this argument to randomized rules.

[3] If not, we can consider a T' that stops as soon as x_1 is observed so that $\mathrm{P}'_{X^\infty}\{T' < V\} \le \mathrm{P}'_{X^\infty}\{T < V\}$.

$$\begin{aligned} &= \sum_{j=0}^{\tau-1}(j+1)p_0^j - \sum_{i=0}^{\tau-1} i p_0^i \\ &= \sum_{j=0}^{\tau-1} p_0^j = \frac{1-p_0^\tau}{1-p_0} = \frac{1-\mathrm{P}'_{X^\infty}\{T<V\}}{2\epsilon}. \end{aligned}$$

This means that if $\mathrm{E}\,T \le (1-2\delta)/(2\epsilon)$ for all $c \in C_0$, then

$$\mathrm{P}'_{X^\infty}\{T<V\} \ \ge\ 2\delta. \tag{A.21}$$

So finally, combining (A.19), (A.20), and (A.21), we have that if $\mathrm{E}\,T \le (1-2\delta)/(2\epsilon)$ for all $c \in C_0$, then for any hypothesizer H there must be some $c' \in C_0$ such that

$$\begin{aligned} \mathrm{P}'_{X^\infty}\{err(H,\mathrm{P}'_X,c,\mathbf{x}^T) > \epsilon\} &\ge \frac{1}{2}\cdot 2\delta \\ &= \delta. \ \blacksquare \end{aligned}$$

Lemma A.7 *For the domain distribution* P'_X *and random variable* $U : X^\infty \to I\!N$ *defined as in (A.14) above: For any learner* $L=(L,H)$, *if* $\mathrm{P}'_{X^\infty}\{Tc < U\} > 0$ *for all* $c \in C$, *then there must be some concept* $c' \in C$ *for which*

$$\mathrm{P}'_{X^\infty}\left\{ err(H,\mathrm{P}'_X,c',\mathbf{x}^T) > \epsilon \,\middle|\, Tc' < U \right\} \ \ge\ \frac{1}{7}.$$

Proof We use the same averaging argument employed by Ehrenfeucht *et al.* [1989, Lemma 2] , however their proof must be reformulated to cope with the fact that the stopping event $\{Tc < U\}$ is no longer independent of the target concept c. Since all that matters is L's behavior on $\{x_1,...,x_d\}$, we focus our attention on the class

$$C_0 \ = \ \{1_S : S \subseteq \{x_1,...,x_d\}\};$$

i.e., the class of concepts that shatters $\{x_1,...,x_d\}$ and yet classifies $c(x_0)=0$ for every $c \in C_0$. Fix a uniform prior Q over C_0. Claim A.8 below shows that any learner $L=(T,H)$ must obtain

$$\mathrm{E}_c\left[\mathrm{E}_\mathbf{x}\left[err(H,\mathrm{P}'_X,c,\mathbf{x}^T) \,\middle|\, T[c\mathbf{x}] < U[\mathbf{x}]\right]\right] \ \ge\ 2\epsilon.$$

This means that there must actually *be* some $c' \in C_0$ that forces this expected error; *i.e.*,

$$\exists c' \in C_0 \quad \mathrm{E}_\mathbf{x}\left[err(H,\mathrm{P}'_X,c',\mathbf{x}^T) \,\middle|\, T[c'\mathbf{x}] < U[\mathbf{x}]\right] \ \ge\ 2\epsilon.$$

Since we have $err(H,\mathrm{P}'_X,c,\mathbf{x}^T) \le 8\epsilon$ for all $c \in C_0$ and $\mathbf{x} \in X^\infty$ by construction,[4] we obtain

$$\mathrm{P}'_{X^\infty}\left\{ err(H,\mathrm{P}'_X,c',\mathbf{x}^T) \,\middle|\, T[c'\mathbf{x}] < U[\mathbf{x}] \right\} \ \ge\ \frac{1}{7}.$$

(This follows because for a random variable X with $\mathrm{E}\,X \ge 2\epsilon$ and $X \le 8\epsilon$, we must have $2\epsilon \le \mathrm{E}\,X \le 8\epsilon\mathrm{P}\{X>\epsilon\} + \epsilon\mathrm{P}\{X \le \epsilon\}$, which implies $\mathrm{P}\{X>\epsilon\} \ge 1/7$.) $\blacksquare$

Claim A.8 *Let* $\mathrm{P}_\times$ *denote the joint distribution on* $C_0 \times X^\infty$ *defined by* $\mathrm{Q}\times\mathrm{P}'_{X^\infty}$. *Then for the random variable* $U : X^\infty \to I\!N$ *defined as above: Any learner* $L=(T,H)$ *such that* $\mathrm{P}_\times\{T[c\mathbf{x}] < U[\mathbf{x}]\} > 0$ *obtains*

$$\mathrm{E}_c\left[\mathrm{E}_\mathbf{x}\left[err(H,\mathrm{P}'_X,c,\mathbf{x}^T) \,\middle|\, T[c\mathbf{x}] < U[\mathbf{x}]\right]\right] \ \ge\ 2\epsilon.$$

[4] Provided H produces hypotheses that classify x_0 as 0 (*i.e.*, $H[c\mathbf{x}^T](x_0)=0$). If not, then we can always construct another hypothesizer H' that does so, and for which $err(H',\mathrm{P}'_X,c,\mathbf{x}^T) \le err(H,\mathrm{P}'_X,c,\mathbf{x}^T)$. The subsequent lower bound will hold for H' and hence for H as well.

Proof Consider the event $A \triangleq \{\langle c, \mathbf{x}\rangle : T[c\mathbf{x}] < U[\mathbf{x}]\}$, which consists of all pairs $\langle c, \mathbf{x}\rangle \in C_0 \times X^\infty$ that cause the stopping rule T to terminate before half of the objects in $\{x_1, ..., x_d\}$ have been observed. We decompose this event as follows. First, write A as

$$\begin{aligned} A &= \sum_M \{\langle c, \mathbf{x}\rangle : T[c\mathbf{x}] < U[\mathbf{x}],\ \mathbf{x}^T \equiv X - M\} \\ &\triangleq \sum_M A_M, \end{aligned}$$

where M ranges over all unobserved subsets of $\{x_1, ...x_d\}$ with size at least $d/2$.[5] That is, A can be decomposed as a union of pair-sets that leave the same set of domain objects unobserved at termination.

Now, consider one of these pair-sets A_M that causes T to halt before a particular subset $M \subseteq \{x_1, ..., x_d\}$ has been observed. This means that the observed objects are a subset of $X - M$. So, given M, we can partition C_0 into $N = 2^{d-|M|}$ equivalence classes $C_{M,1}, ..., C_{M,i}, ..., C_{M,N}$, where each equivalence class shatters M, but the concepts in each class identically label the objects in $X - M$. Then we can write A as a disjoint union

$$\begin{aligned} A &= \sum_M \sum_i \{\langle c, \mathbf{x}\rangle : T[c\mathbf{x}] < U[\mathbf{x}],\ \mathbf{x}^T \equiv X - M,\ c \in C_{M,i}\} \\ &\triangleq \sum_M \sum_i A_{M,i}, \end{aligned}$$

where $A_{M,i}$ consists of all pairs $\langle c, \mathbf{x}\rangle$ that leave a particular set of domain objects M unobserved at termination, and where the concepts $C_{M,i} = \{c : \langle c, \mathbf{x}\rangle \in A_{M,i}\}$ identically label every object in $X - M$ but shatter the unobserved set M.

Finally, consider one of these pair-sets $A_{M,i}$. Notice that if an object sequence $\mathbf{x} \in X^\infty$ causes $\langle c_1, \mathbf{x}\rangle \in A_{M,i}$ for some $c_1 \in C_{M,i}$, then it must cause $\langle c, \mathbf{x}\rangle \in A_{M,i}$ for *every* $c \in C_{M,i}$. This is because $\langle c_1, \mathbf{x}\rangle \in A_{M,i}$ means $T[c_1\mathbf{x}]$ halts before any element of M has been observed, and since every $c \in C_{M,i}$ labels $X - M$ identically, we must have $T[c_1\mathbf{x}] = T[c\mathbf{x}]$ and hence $c_1\mathbf{x}^T = c\mathbf{x}^T$ for every $c \in C_{M,i}$. This shows that $A_{M,i}$ is a cartesian product $A_{M,i} = C_{M,i} \times X^\infty_{M,i}$, where $C_{M,i} = \{c : \langle c, \mathbf{x}\rangle \in A_{M,i}\}$ and $X^\infty_{M,i} = \{\mathbf{x} : \langle c, \mathbf{x}\rangle \in A_{M,i}\}$. That is, A can be decomposed as a finite union of disjoint rectangles

$$A = \sum_M \sum_i C_{M,i} \times X^\infty_{M,i}, \tag{A.22}$$

where each rectangle $C_{M,i} \times X^\infty_{M,i}$ causes a particular (large) set of domain objects M to remain unobserved at termination, and where the concept class $C_{M,i}$ identically labels every observed portion of a sequence $\mathbf{x} \in X^\infty_{M,i}$ but shatters the unobserved set M.

We now prove the claim that $\mathrm{E}_{\langle c,\mathbf{x}\rangle} \left[err(H, \mathrm{P}'_X, c, \mathbf{x}^T) \mid A \right] \geq 2\epsilon$. Note that since $P_X(A) > 0$ by hypothesis, it suffices to prove

$$\mathrm{E}_{\langle c,\mathbf{x}\rangle} \left[err(H, \mathrm{P}'_X, c, \mathbf{x}^T) \mid C_{M,i} \times X^\infty_{M,i} \right] \geq 2\epsilon \tag{A.23}$$

for any rectangle where $\mathrm{P}_X \left(C_{M,i} \times X^\infty_{M,i} \right) > 0$.[6] Consider such a rectangle and let $\mathrm{P}_{M,i}$ denote the conditional probability of P'_{X^∞} given $X^\infty_{M,i}$, and let $\mathrm{Q}_{M,i}$ denote the conditional probability of Q given $C_{M,i}$. The idea is to prove that if T stops before half the objects in $\{x_1, ..., x_d\}$ have been observed, then no hypothesizer H can achieve a small average error over the possible labellings of the unobserved objects.

[5]Here the notation $\sum$ means disjoint union, and *vector* $\equiv$ *set* means *contents*(*vector*) = *set*.

[6]To see that this suffices, note that $P_X(A) > 0$ by hypothesis, so there must be some rectangle $C_{M,i} \times X^\infty_{M,i}$ that gives $\mathrm{P}_X \left(C_{M,i} \times X^\infty_{M,i} \right) > 0$. Now, obviously for any random variable X, if $\mathrm{E}[X|B_j] \geq \alpha$ for disjoint B_j, $\mathrm{P}(B_j) > 0$, then we must also have $\mathrm{E}[X|\sum_j B_j] \geq \alpha$.

Recall that Q is uniform on C_0, and thus $\mathrm{Q}_{M,i}$ is uniform on $C_{M,i}$. Rewriting (A.23) gives

$$\begin{aligned}\mathrm{E}_{\langle c,\mathbf{x}\rangle}\left[err(H,\mathrm{P}'_X,c,\mathbf{x}^T)\mid C_{M,i}\times X^\infty_{M,i}\right] &= \int_{C_{M,i}}\int_{X^\infty_{M,i}} err(H,\mathrm{P}'_X,c,\mathbf{x}^T)\,d\mathrm{P}_{M,i}\,d\mathrm{Q}_{M,i}\\ &= \int_{X^\infty_{M,i}}\int_{C_{M,i}} err(H,\mathrm{P}'_X,c,\mathbf{x}^T)\,d\mathrm{Q}_{M,i}\,d\mathrm{P}_{M,i},\end{aligned}$$

by Fubini's theorem [Ash, 1972, Theorem 2.6.6]. Now, consider an arbitrary $\mathbf{x}\in X^\infty_{M,i}$. Recall from (A.22) that every target concept $c\in C_{M,i}$ yields the same observation sequence $c\mathbf{x}^T$ for $\mathbf{x}$. This means that any hypothesizer H must produce the same hypothesis $h[\mathbf{x}]=H[c\mathbf{x}^T]$ for any $c\in C_{M,i}$. So for this $\mathbf{x}$ we obtain an average error of

$$\begin{aligned}\int_{C_{M,i}} err(H,\mathrm{P}'_X,c,\mathbf{x}^T)\,d\mathrm{Q}_{M,i} &= \int_{C_{M,i}}\frac{8\epsilon}{d}\sum_{x\in M}1_{\{h[\mathbf{x}](x)\neq c(x)\}}\,d\mathrm{Q}_{M,i}\\ &\qquad\text{since each point in } M \text{ has weight } 8\epsilon/d \text{ under } \mathrm{P}'_X,^{7}\\ &= \frac{8\epsilon}{d}\sum_{x\in M}\int_{C_{M,i}}1_{\{h[\mathbf{x}](x)\neq c(x)\}}\,d\mathrm{Q}_{M,i}\\ &= \frac{8\epsilon}{d}\sum_{x\in M}\frac{1}{2}\\ &\qquad\text{since half the concepts in } C_{M,i} \text{ agree with } h[\mathbf{x}] \text{ on } x\in M,\\ &= \frac{4\epsilon}{d}|M|\\ &\geq \frac{4\epsilon}{d}\frac{d}{2}\\ &\qquad\qquad\text{since } |M|\geq d/2 \text{ by construction,}\\ &= 2\epsilon. \blacksquare\end{aligned}$$

Theorem 2.16 *For any $\epsilon>0$, $\delta>0$, and any finite concept class C: Given i.i.d. examples generated by any distribution P_X and target concept $c\in C$, Procedure* `Smb` *observes an average training sample size of at most*

$$\mathrm{E}\,T_{\mathtt{Smb}}(M,\epsilon,\delta)\ \leq\ \frac{\kappa M}{\epsilon}+\left(\frac{\kappa}{\kappa-1-\ln\kappa}\right)\frac{1}{\epsilon}\left(\ln\frac{M}{\delta}+1\right); \tag{A.24}$$

using a hypothesizer H with $M=M(C,H)$ and a constant $\kappa>1$.

Proof We prove this using a simplified version of the argument from Theorem 2.11 above. As in Theorem 2.11 we exploit the two key facts that *(i)* the call to `sprt` eventually accepts any ϵ/κ-good hypothesis wp1, and *(ii)* H must eventually produce such a hypothesis wp1. Thus, we can bound the expected time for `Smb` to terminate by the time for H to produce an ϵ/κ-good hypothesis plus the time for `sprt` to accept such a hypothesis.

Fix arbitrary $\epsilon>0$, $\delta>0$, $\kappa>1$, and choose an arbitrary target concept $c\in C$ and domain distribution P_X. Let $T_H(\epsilon)$ denote the time when H returns an ϵ-good hypothesis, and let $T_{\mathtt{sprt}}(\epsilon,\kappa,\delta)$ denote the time

[7] Note that this assumes $h[\mathbf{x}]$ correctly classifies all observed domain objects $x\in(X-M)$. If not, we can construct an alternative hypothesizer H' that produces consistent hypotheses, and for which $err(H',\mathrm{P}'_X,c,\mathbf{x}^T)\leq err(H,\mathrm{P}'_X,c,\mathbf{x}^T)$, as in Footnote 4.

$\mathtt{sprt}(\,h(x) \neq c(x),\ \epsilon/\kappa,\ \epsilon,\ \delta,\ 0\,)$ takes to accept an ϵ/κ-good hypothesis. Then we can bound `Smb`'s stopping time by

$$T_{\mathtt{Smb}}(\epsilon,\delta) \;\le\; T_H(\epsilon/\kappa) + T_{\mathtt{sprt}}(\epsilon,\kappa,\delta/M).$$

Taking expectations, we get

$$\mathrm{E}\,T_{\mathtt{Smb}}(\epsilon,\delta) \;\le\; \mathrm{E}\,T_H(\epsilon/\kappa) + \mathrm{E}\,T_{\mathtt{sprt}}(\epsilon,\kappa,\delta/M). \tag{A.25}$$

Now, for the T_H term, we know that

$$\mathrm{E}\,T_H(\epsilon/\kappa) \le \frac{\kappa M}{\epsilon}, \tag{A.26}$$

since any ϵ/κ-bad hypothesis is guaranteed to make a mistake in less than κ/ϵ expected time, and there can be at most M such hypotheses. Finally, for the $T_{\mathtt{sprt}}$ term, Lemma A.4 shows that

$$\mathrm{E}\,T_{\mathtt{sprt}}(\epsilon,\,\kappa,\,\delta/M\,) \;\le\; \left(\frac{\kappa}{\kappa-1-\ln\kappa}\right)\frac{1}{\epsilon}\left(\ln\frac{M}{\delta}+1\right). \tag{A.27}$$

So combining (A.25), (A.26), and (A.27) directly yields the stated bound. ▌

Theorem 2.18 *The following are equivalent:*

1. *C can be pac-learned with a bounded expected sample size for each individual c in C.*
2. *C can be decomposed as $C = \bigcup_{i=1}^{\infty} C_i$ where $\mathrm{VC}(C_i) < \infty$.*
3. *Procedure `S` pac-learns C with a hypothesizer H that produces consistent concepts from the earliest possible class in the decomposition.*

Proof (1 ⇒ 2) Following [Benedek and Itai, 1988b, Theorem 2], we assume there is some learner L that pac-learns C with bounded expected sample size for each $c \in C$. Fix $0 < \epsilon \le 1/8$ and $0 < \delta \le 1/683$. For $t = 1, 2, ...,$ let C_t be the set of concepts from C that L pac(ϵ,δ)-learns with an expected training sample size of at most t. By assumption, each $c \in C$ must belong to some C_t. But by Theorem 2.14, each C_t must have finite VCdimension. Thus, $C = \bigcup_{t=1}^{\infty} C_t$ and $\mathrm{VC}(C_t) < \infty$.

(2 ⇒ 3) We are given a decomposition $C = \bigcup_{i=1}^{\infty} C_i$, $\mathrm{VC}(C_i) < \infty$. Fix an arbitrary $c \in C$ and an arbitrary P_X. There must be a first class C_j that contains c. By construction, the hypothesizer H will only produce consistent concepts from the set of concepts $C^j = \bigcup_{i=1}^{j} C_i$. Lemma A.9 below shows that this class C^j has finite VCdimension, and therefore we can directly apply Theorem 2.9 to show `S` correctly pac-learns c.

(3 ⇒ 1) Follows directly. ▌

Lemma A.9 *For two concept classes C_1 and C_2:*

$$\mathrm{VC}(C_1 \cup C_2) \;\le\; 3.1\,\max\{\,\mathrm{VC}(C_1),\ \mathrm{VC}(C_2)\,\} + 2.$$

Proof Recall from Definition 2.3 that $\mathrm{VC}(C) = d$ means there is a set of d domain objects $\{x_1, ..., x_d\}$ that can be independently labelled by choosing concepts from C; *i.e.*, C "picks out" all 2^d distinct subsets of $\{x_1, ..., x_d\}$. Blumer *et al.* [1989, Lemma A2.1] prove that the maximum number of distinct subsets any concept class C with $\mathrm{VC}(C) = d$ can pick out from a set $\{x_1, ..., x_t\}$ of size t is bounded by

$$\Pi_C(t) \;\le\; \left(\frac{et}{d}\right)^d \tag{A.28}$$

for $1 \le d \le t$. We use this fact to derive an upper bound on the VCdimension of $C_1 \cup C_2$: Let $d_1 = \mathrm{vc}(C_1)$ and $d_2 = \mathrm{vc}(C_2)$. By (A.28), the maximum number of distinct subsets of any set $\{x_1, ..., x_t\}$ that can be picked out by concepts from $C_1 \cup C_2$ is bounded by

$$\begin{aligned} \Pi_{C_1 \cup C_2}(t) &\le \Pi_{C_1}(t) + \Pi_{C_2}(t) \\ &\le \left(\frac{et}{d_1}\right)^{d_1} + \left(\frac{et}{d_2}\right)^{d_2} \\ &\le 2\left(\frac{et}{d}\right)^{d} \qquad \text{where } d = \max\{d_1, d_2\}. \end{aligned}$$

Now, if $\mathrm{vc}(C_1 \cup C_2) = t$, then by definition we must have $\Pi_{C_1 \cup C_2}(t) = 2^t$. Therefore, the VCdimension of $C_1 \cup C_2$ can be no larger than the largest t for which $2(et/d)^d \ge 2^t$.

We prove that $t = 3.1d + 2$ implies $2(et/d)^d < 2^t$ and hence $\mathrm{vc}(C_1 \cup C_2) < 3.1d + 2$. To do this, note that $2(et/d)^d < 2^t$ if and only if $(d/\ln 2)\ln(et/d) < t - 1$, and consider

$$\begin{aligned} \frac{d}{\ln 2}\ln\left(\frac{et}{d}\right) &= \frac{d}{\ln 2}\left[\ln\left(3.1 + \frac{2}{d}\right) + 1\right] \\ &\qquad \text{substituting } t = 3.1d + 2, \\ &\le \frac{d}{\ln 2}\left[\left(\frac{3.1}{3} + \frac{2}{3d} + \ln 3 - 1\right) + 1\right] \\ &\qquad \text{by Lemma A.10 below; using } \alpha = 1/3, \\ &< \frac{d}{\ln 2}\left(2.14 + \frac{2}{3d}\right) \\ &< 3.1d + 1 \;=\; t - 1. \ \blacksquare \end{aligned}$$

A.3 Some algebraic bounds

Here we present proofs of the various algebraic bounds used in Section A.2. Most of these bounds are derived from the following two lemmas which provide linear approximations to log functions.

Lemma A.10 [Shawe-Taylor, Anthony and Biggs, 1993] *For any $\alpha > 0$:*

$$\ln x \;\le\; \alpha x + \ln(1/\alpha) - 1.$$

Corollary *For any $a > 0$, $b > 0$, and $0 < \alpha \le b/e^2$: $x \ge a \ln bx$ for*

$$x \;\ge\; \frac{a}{1 - a\alpha}\ln\frac{b}{\alpha e}. \tag{A.29}$$

Proof The lemma is proved in [Shawe-Taylor, Anthony and Biggs, 1993]. For the corollary, we seek a lower bound on x that ensures $x \ge a \ln bx$. It is proposed that (A.29) is sufficient to do this (for $\alpha \le b/e^2$). It suffices to show $x \ge a \ln bx$ holds at the bound (A.29) since x grows faster than $a \ln bx$ for $x > a$, and $\alpha \le b/e^2$ ensures a is smaller than (A.29). To this end, consider

$$a \ln bx \;=\; a \ln\left(b \frac{a}{1 - a\alpha}\ln\frac{b}{\alpha e}\right) \qquad \text{plugging in (A.29),}$$

$$
\begin{aligned}
&= a\ln b + a\ln\left(\frac{a}{1-a\alpha}\ln\frac{b}{\alpha e}\right) \\
&\le a\ln b + a\left(\frac{\alpha a}{1-a\alpha}\ln\frac{b}{\alpha e} + \ln\frac{1}{\alpha e}\right) \qquad \text{by the lemma,} \\
&= a\ln b + \frac{a^2\alpha}{1-a\alpha}\ln b + \left(\frac{a^2\alpha}{1-a\alpha} + a\right)\ln\frac{1}{\alpha e} \\
&= a\left(1+\frac{a\alpha}{1-a\alpha}\right)\ln b + a\left(1+\frac{a\alpha}{1-a\alpha}\right)\ln\frac{1}{\alpha e} \\
&= \frac{a}{1-a\alpha}\ln\frac{b}{\alpha e} \;=\; x. \;\blacksquare
\end{aligned}
$$

Lemma A.11 *For any* $0 < \beta \le 4/e^2$*:* $\quad (\ln x)^2 \;\le\; \beta x + 4\left(\ln\frac{4}{\beta e}\right)^2 \quad$ *for* $\quad x \ge \frac{4}{\beta}\ln\frac{4}{\beta e}$.

Proof We seek a linear function $\beta x + c$ that is an upper bound on $(\ln x)^2$. Consider the difference function $f(x) = \beta x + c - (\ln x)^2$. This difference is increasing for values of x where $f'(x) = \beta - (2\ln x)/x > 0$. In particular, $f'(x) > 0$ for x larger than $(4/\beta)\ln[4/(\beta e)]$. (To see this, note that $x \ge (4/\beta)\ln[4/(\beta e)]$ implies $x \ge (2/\beta)\ln x$, by the corollary to Lemma A.10 above; using $\alpha = \beta/4$ and ensuring $\beta/4 \le 1/e^2$.)

Since $f'(x) > 0$ for $x \ge (4/\beta)\ln[4/(\beta e)]$, we need only find a constant c that ensures $f(x) > 0$ at $x = (4/\beta)\ln[4/(\beta e)]$ and the result will follow automatically for larger x. It turns out that a suitable choice is $c = 4[\ln(4/(\beta e))]^2$. To see that this is sufficient, consider

$$
\begin{aligned}
f(x) &= \beta x + c - (\ln x)^2 \\
&= 4\ln\frac{4}{\beta e} + 4\left(\ln\frac{4}{\beta e}\right)^2 - \left(\ln\frac{4}{\beta}\ln\frac{4}{\beta e}\right)^2 \\
&\qquad \text{for } c = 4[\ln(4/(\beta e))]^2 \text{ and } x = (4/\beta)\ln[4/(\beta e)], \\
&\ge 4\left(\ln\frac{4}{\beta e}\right)\left(\ln\frac{4}{\beta}\right) - \left[\ln\frac{4}{\beta}\left(\frac{4}{\beta e^2}\right)\right]^2 \\
&\qquad \text{since } \ln(4/(\beta e)) \le 4/(\beta e^2) \text{ by Lemma A.10 above (using } \alpha = 1\text{)}, \\
&= 4\left(\ln\frac{4}{\beta e}\right)\left(\ln\frac{4}{\beta}\right) - 4\left(\ln\frac{4}{\beta e}\right)^2 \\
&> 0 \\
&\qquad \text{since } \ln(4/\beta) > \ln(4/(\beta e)) \text{ for } \beta > 0. \;\blacksquare
\end{aligned}
$$

Given these lemmas, we now prove the algebraic bounds referred to in the previous section.

Claim A.12 *For $\kappa > 1$ and $0 \leq \epsilon \leq 1$:* $\dfrac{\kappa - 1 - \ln\kappa}{1 - \sqrt{\epsilon/\kappa}} \leq 1.06\kappa$.

Proof First notice that $\sqrt{\epsilon/\kappa} \leq 1/\sqrt{\kappa}$ for all $0 \leq \epsilon \leq 1$, so it suffices to consider $\epsilon = 1$. We seek a constant α such that

$$\alpha \geq \frac{1}{\kappa}\left(\frac{\kappa - 1 - \ln\kappa}{1 - 1/\sqrt{\kappa}}\right) = \frac{\kappa - 1 - \ln\kappa}{\kappa - \sqrt{\kappa}}$$

for all $\kappa > 1$. It can be numerically verified that $\alpha = 1.06$ is a suitable choice. ∎

Claim A.13 *For $d \geq 2$ and $\epsilon \leq \min\{1/4,\ \kappa/(d\ln 1.05d)^2\}$:*

$$\frac{3}{2}\ln\frac{14\kappa}{\epsilon} \geq \ln\frac{2\kappa''}{\epsilon} + \ln\ln\frac{14\kappa}{\epsilon} + \ln d + \frac{3}{4}.$$

Proof First of all, it can be verified that $2\kappa'' \leq 4\kappa$ for all $\epsilon \leq 1/4$, $\kappa \geq 1$ (recalling that $\kappa'' = \kappa/(1-\sqrt{\epsilon/\kappa})$). So it suffices to show that for $\epsilon \leq \kappa/(d\ln 1.05d)^2$ we have

$$\frac{3}{2}\ln\frac{14\kappa}{\epsilon} \geq \ln\frac{4\kappa}{\epsilon} + \ln\ln\frac{14\kappa}{\epsilon} + \ln d + \frac{3}{4}.$$

Since $\ln(4\kappa/\epsilon) = \ln(14\kappa/\epsilon) + \ln(4/14)$, this is equivalent to

$$\frac{1}{2}\ln\frac{14\kappa}{\epsilon} \geq \ln\frac{4}{14} + \ln\ln\frac{14\kappa}{\epsilon} + \ln d + \frac{3}{4},$$

which in turn is equivalent to

$$\frac{14\kappa}{\epsilon} \geq \left(\frac{4e^{3/4}}{14} d\ln\frac{14\kappa}{\epsilon}\right)^2.$$

Now let $x = 14\kappa/\epsilon$. Noticing that $(4e^{3/4}/14)^2 < 0.37$, we need only establish that

$$x \geq 0.37d^2(\ln x)^2 \tag{A.30}$$

holds for all $x \geq 14d^2(\ln 1.05d)^2$.

Let $f(x) = x - 0.37d^2(\ln x)^2$ and note that $f'(x) = 1 - 0.74d^2(\ln x)/x$. By the corollary to Lemma A.10 above we have that $f'(x) > 0$ for all $x \geq 2d^2\ln(1.05d)$ (using $\alpha = 1/[4 \times 0.74d^2]$, which ensures $\alpha < 1/e^2$ for $d \geq 2$). Clearly, $14d^2(\ln 1.05d)^2 > 2d^2\ln(1.05d)$, so it suffices to establish that the inequality (A.30) holds at $x = 14d^2(\ln 1.05d)^2$. To this end, consider

$$\begin{aligned}
0.37d^2(\ln x)^2 &= 0.37d^2\left[\ln 14d^2(\ln 1.05d)^2\right]^2 \\
&\qquad \text{plugging in } x = 14d^2(\ln 1.05d)^2, \\
&\leq 0.37d^2\left[\frac{7}{0.37}(\ln 1.05d)^2 + 4\left(\ln\frac{8 \times 0.37}{e}d^2\right)^2\right] \\
&\qquad \text{for any } d \geq 1 \text{ by Lemma A.11; using } \beta = 1/[2 \times 0.37d^2] \\
&\qquad \text{(which ensures } \beta \leq 4/e^2 \text{ for } d \geq 2), \\
&< 7d^2(\ln 1.05d)^2 + 16 \times 0.37d^2(\ln 1.05d)^2 \\
&\qquad \text{since } 8 \times 0.37/e < 1.05^2, \\
&< 14d^2(\ln 1.05d)^2 = x.
\end{aligned}$$

∎

Claim A.14 *For all $\epsilon > 0$:* $\dfrac{\kappa}{\kappa - 1 - \ln\kappa} < \dfrac{1}{1-\sqrt{\epsilon}}$ *for* $\kappa > \dfrac{2}{\sqrt{\epsilon}} \ln \dfrac{2}{\sqrt{\epsilon}}$.

Proof Note that $\kappa/(\kappa - 1 - \ln\kappa) < 1/(1-\sqrt{\epsilon})$ is equivalent to $\kappa > \epsilon^{-1/2}(1+\ln\kappa)$. To verify that this inequality holds for $\kappa > (2/\sqrt{\epsilon})\ln(2/\sqrt{\epsilon})$ consider

$$
\begin{aligned}
\frac{1}{\sqrt{\epsilon}}(1+\ln\kappa) \quad &< \quad \frac{1}{\sqrt{\epsilon}}\left[1 + \ln\left(\frac{2}{\sqrt{\epsilon}} \ln \frac{2}{\sqrt{\epsilon}}\right)\right] \\
&\qquad \text{plugging in } \kappa > (2/\sqrt{\epsilon})\ln(2/\sqrt{\epsilon}), \\
&\leq \quad \frac{1}{\sqrt{\epsilon}}\left[1 + \ln\frac{2}{\sqrt{\epsilon}} + \ln\frac{2}{\sqrt{\epsilon}} - 1\right] \\
&\qquad \text{by Lemma A.10 above (using } \alpha = \sqrt{\epsilon}/2\text{)}, \\
&= \quad \frac{2}{\sqrt{\epsilon}} \ln \frac{2}{\sqrt{\epsilon}} \quad = \quad \kappa. \ \blacksquare
\end{aligned}
$$

Appendix B

Technical details: Chapter 3

Definitions and notation: In this appendix we adopt the same definitions and notation used in Appendix A. In particular, we let $c\mathbf{x}^t$ denote the sequence $c\mathbf{x}^t = \langle\langle x_1, c(x_1)\rangle, \langle x_2, c(x_2)\rangle, ..., \langle x_t, c(x_t)\rangle\rangle$ generated by a target concept c and object sequence $\mathbf{x} = \langle x_1, x_2, ..., x_t\rangle$. Thus, for a hypothesizer $H : (X \times \{0,1\})^* \to \{0,1\}^X$ we denote the hypothesis it produces given training sequence $c\mathbf{x}^t$ by $H[c\mathbf{x}^t]$. We also let $H[c\mathbf{x}^T]$ denote $H[c\mathbf{x}^t]$ at $t = T[c\mathbf{x}]$, and let $err(H, c, \mathbf{x}^t) \triangleq d_{\mathrm{P}}(H[c\mathbf{x}^t], c)$ denote the error of H's hypothesis given $c\mathbf{x}^t$, with respect to c and P.

Theorem 3.8 (Correctness) *For any $\epsilon > 0$, $\delta > 0$, and any concept space (C, P) with $N_{\epsilon/2}(C, \mathrm{P}) < \infty$: Given i.i.d. examples generated by* P *and any target concept $c \in C$, Procedure* `Sbi` *returns a hypothesis h such that $\mathrm{P}\{h(x) \neq c(x)\} \leq \epsilon$ with probability at least $1 - \delta$.*

Proof Since `Sbi` constructs an $\epsilon/2$-cover V of (C, P), it must consider at least one $\epsilon/2$-good hypothesis $h \in V$. Lemma A.2 in Appendix A shows that `Sbi`'s call to `sprt` eventually accepts such a hypothesis wp1, and therefore `Sbi` is guaranteed to terminate wp1. On the other hand, each call to `sprt` accepts an ϵ-bad hypothesis with probability at most $\delta/|V|$ by construction, so the total probability that `Sbi` accepts an ϵ-bad hypothesis is bounded by $|V| \times \delta/|V| = \delta$. Therefore, the probability that `Sbi` returns an ϵ-good hypothesis must be at least $1 - \delta$. ▌

Theorem 3.10 (Data-efficiency) *For any $\epsilon > 0$, $\delta > 0$, and any concept space (C, P) with $N_{\epsilon/2}(C, \mathrm{P}) < \infty$: Given i.i.d. examples generated by* P *and any target concept $c \in C$, Procedure* `Sbi` *observes an average training sample size of at most*

$$\mathrm{E}\, T_{\texttt{Sbi}}(C, \mathrm{P}, \epsilon, \delta) \ \leq\ \frac{6.5178}{\epsilon}\left(\ln N_{\epsilon/2}(C, \mathrm{P}) + \ln\frac{1}{\delta} + 1\right). \tag{3.1}$$

Proof As above, since `Sbi` constructs an $\epsilon/2$-cover V of (C, P) it must consider at least one $\epsilon/2$-good hypothesis $h \in V$. Moreover, the call to `sprt(` $h(x) \neq c(x)$, $\epsilon/2$, ϵ, $\delta/|V|$, 0 `)` eventually accepts this hypothesis wp1. So we can bound `Sbi`'s overall stopping time by the time it takes `sprt` to accept this $\epsilon/2$-good hypothesis

$$T_{\texttt{Sbi}}(C, \mathrm{P}, \epsilon, \delta) \ \leq\ T_{\texttt{sprt}(\ h(x)\neq c(x),\ \epsilon/2,\ \epsilon,\ \delta/|V|,\ 0\ )},$$

where $|V| = N_{\epsilon/2}(C, \mathrm{P})$. Applying Lemma A.4 (bounding $\mathrm{E}\, T_{\texttt{sprt}}$) yields the result. ▌

Theorem 3.12 (Data-complexity) *For any $\epsilon > 0$, $\delta > 0$, and any concept space (C, P): A learner that always observes an average training sample size less than*

$$t_{avg}(C, \mathrm{P}, \epsilon, \delta) \ =\ \frac{1}{2}\log_2\left[N_{2\epsilon}(C, \mathrm{P})\left(\frac{1}{2} - \delta\right)\right]$$

for every fixed $c \in C$ will fail to meet the pac(ϵ, δ)-criterion for some fixed target $c' \in C$.

Proof Fix an arbitrary $\epsilon > 0$ and an arbitrary learner $L = (T, H)$. As in the proof of Theorem 2.14, the idea is to show that if L's expected stopping time $\mathrm{E}\,T$ is too small relative to t_{BI}, then H must fail to meet the pac(ϵ, δ)-criterion for some $c' \in C$.

For any T and H, and any c and t we have

$$\begin{aligned} \mathrm{P}_{X^\infty}\{err(H, c, \mathbf{x}^T) > \epsilon\} &= 1 - \mathrm{P}_{X^\infty}\{err(H, c, \mathbf{x}^T) \le \epsilon\} \\ &= 1 - \mathrm{P}_{X^\infty}\{T[c\mathbf{x}] \le t,\ err(H, c, \mathbf{x}^T) \le \epsilon\} \\ &\quad - \mathrm{P}_{X^\infty}\{T[c\mathbf{x}] > t,\ err(H, c, \mathbf{x}^T) \le \epsilon\} \\ &\ge 1 - \mathrm{P}_{X^t}\{err(H, c, \mathbf{x}^t) \le \epsilon\} - \mathrm{P}_{X^\infty}\{T[c\mathbf{x}] > t\}, \end{aligned} \tag{B.2}$$

since $\mathrm{P}_{X^\infty}\{T[c\mathbf{x}] \le t,\ err(H, c, \mathbf{x}^T) \le \epsilon\} \ \ge\ \mathrm{P}_{X^t}\{err(H, c, \mathbf{x}^t) \le \epsilon\}$.[1]

Now we find lower bounds on these terms. First, Lemma B.1 below shows that for any t there must be some $c' \in C$ for which

$$\mathrm{P}_{X^t}\{err(H, c', \mathbf{x}^t) \le \epsilon\} \ \le\ \frac{2^t}{N_{2\epsilon}(C, \mathrm{P})}. \tag{B.3}$$

Then by Markov's inequality we have that, if $\mathrm{E}_{\mathbf{x}}T[c\mathbf{x}] \le t/2$ for all $c \in C$,

$$\mathrm{P}_{X^\infty}\{T[c\mathbf{x}] > t\} \ \le\ \frac{1}{2}. \tag{B.4}$$

So combining (B.2), (B.3), and (B.4) shows that for any t if $\mathrm{E}\,Tc \le t/2$ for all $c \in C$, then there must be some $c' \in C$ for which

$$\mathrm{P}_{X^\infty}\{err(H, c', \mathbf{x}^T) > \epsilon\} \ \ge\ 1 - \frac{2^t}{N_{2\epsilon}(C, \mathrm{P})} - \frac{1}{2}.$$

Choosing $t = \log_2[N_{2\epsilon}(C, \mathrm{P})(\frac{1}{2} - \delta)]$ finishes the proof. ▌

Lemma B.1 *For any concept space (C, P) and any hypothesizer H, there must be some $c' \in C$ for which*

$$\mathrm{P}_{X^t}\{err(H, c', \mathbf{x}^t) \le \epsilon\} \ \le\ \frac{2^t}{N_{2\epsilon}(C, \mathrm{P})}.$$

Proof To prove this lemma we need to consider a "2ϵ-packing" of the space, V, consisting of a maximal collection of pairwise 2ϵ-separated concepts in C. If we let $M_{2\epsilon}(C, \mathrm{P})$ denote the size of the largest 2ϵ-packing of the space, then it is well known that $M_{2\epsilon}(C, \mathrm{P}) \ge N_{2\epsilon}(C, \mathrm{P})$ [Kolmogorov and Tihomirov, 1961], and thus, $|V| \ge N_{2\epsilon}(C, \mathrm{P})$. We now use an averaging argument, originally due to Benedek and Itai [1988, Lemma 5].

Fix a uniform prior Q over V, and consider an arbitrary hypothesizer H. Thinking of concepts as being randomly drawn from V according to Q, consider

$$\begin{aligned} \mathrm{E}_c\left[\mathrm{P}_{X^t}\{err(H, c, \mathbf{x}^t) \le \epsilon\}\right] &= \int_{c \in V} \int_{\mathbf{x}^t \in X^t} 1_{\{err(H,c,\mathbf{x}^t) \le \epsilon\}}\ d\mathrm{P}_{X^t}(\mathbf{x}^t)\, d\mathrm{Q}(c) \\ &= \int_{\mathbf{x}^t \in X^t} \int_{c \in V} 1_{\{err(H,c,\mathbf{x}^t) \le \epsilon\}}\ d\mathrm{Q}(c)\, d\mathrm{P}_{X^t}(\mathbf{x}^t) \end{aligned}$$

by Fubini's theorem,

[1] This assumes H does not change its guess after T has stopped; *i.e.*, $H[c\mathbf{x}^T] = H[c\mathbf{x}^t]$ for $t \ge T[c\mathbf{x}]$. If not, then we can always construct a new hypothesizer H_T that freezes H's hypotheses once the stopping variable T has terminated (and thus is behaviorally equivalent to H with respect to stopping rule T), and still provides the lower bound $\mathrm{P}_{X^\infty}\{T[c\mathbf{x}] \le t,\ err(H, c, \mathbf{x}^T) \le \epsilon\} \ \ge\ \mathrm{P}_{X^t}\{err(H, c, \mathbf{x}^t) \le \epsilon\}$.

$$\leq \int_{\mathbf{x}^t \in X^t} \frac{2^t}{|V|} \, d\mathrm{P}_{X^t}(\mathbf{x}^t)$$

by Claim B.2 below,

$$= \frac{2^t}{|V|}.$$

This means that there must *be* some $c' \in C$ such that

$$\begin{aligned} \mathrm{P}_{X^t}\{err(H, c', \mathbf{x}^t) \leq \epsilon\} &\leq \frac{2^t}{|V|} \\ &\leq \frac{2^t}{N_{2\epsilon}(C, \mathrm{P})}. \ \blacksquare \end{aligned}$$

Claim B.2 *For any set of pairwise 2α-separated concepts V, and any object sequence $\mathbf{x}^t$ of length t:* No *hypothesizer H can produce hypotheses that α-approximate more than a fraction $2^t/|V|$ of the concepts in V.*

Proof Fix an arbitrary 2α-separated subset V of a space (C, P) and an arbitrary object sequence $\mathbf{x}^t$. Let Q denote the uniform prior over V and consider an arbitrary hypothesizer H. Then, considering concepts as being randomly drawn from V according to Q, we have

$$\begin{aligned} \mathrm{Q}\left\{c \in V : err(H, c, \mathbf{x}^t) \leq \alpha\right\} &= \int_{c \in V} 1_{\{d_{\mathrm{P}}(H[c\mathbf{x}^t], c) \leq \alpha\}} \, d\mathrm{Q}(c) \\ &\leq \int_{c \in V} \sum_{\ell^t \in \{0,1\}^t} 1_{\{d_{\mathrm{P}}(H[\ell^t \mathbf{x}^t], c) \leq \alpha\}} \, d\mathrm{Q}(c) \end{aligned}$$

where $\ell^t \mathbf{x}^t$ denotes $\langle\langle x_1, \ell_1\rangle, \ldots, \langle x_t, \ell_t\rangle\rangle$,

$$\begin{aligned} &= \sum_{\ell^t \in \{0,1\}^t} \int_{c \in V} 1_{\{d_{\mathrm{P}}(H[\ell^t \mathbf{x}^t], c) \leq \alpha\}} \, d\mathrm{Q}(c) \\ &\leq \sum_{\ell^t \in \{0,1\}^t} \frac{1}{|V|} \end{aligned}$$

since each $H[\ell^t \mathbf{x}^t]$ can only be within α of one $c \in V$,

$$= \frac{2^t}{|V|}. \ \blacksquare$$

Proposition 3.15 $NB_k(C, \mathrm{P}) \leq (bk)^d$ *implies* $N_\epsilon(C, \mathrm{P}) \leq \left(\frac{k}{\epsilon}\right)^{d\left(\frac{\ln b}{\ln k}+1\right)}$.

Proof Clearly, $N_\epsilon \leq NB_k \cdot N_{k\epsilon}$. Therefore, assuming $NB_k \leq (bk)^d$, we get

$$\begin{aligned} N_\epsilon &\leq (bk)^d N_{\epsilon k} \\ &\leq \left[(bk)^d\right]^2 N_{\epsilon k^2} \leq \ldots \leq \left[(bk)^d\right]^i N_{\epsilon k^i} \leq \ldots \leq \left[(bk)^d\right]^{\lceil \log_k(1/\epsilon) \rceil} N_1 \\ &\leq \left[(bk)^d\right]^{\log_k(k/\epsilon)} \cdot 1 \\ &= \left(\tfrac{k}{\epsilon}\right)^{d\left(\frac{\ln b}{\ln k}+1\right)}, \end{aligned}$$

since $(bk)^{\log_k(k/\epsilon)} = (k/\epsilon)^{\log_k(bk)}$.

Proposition 3.16 (Data-efficiency) *For any $\epsilon > 0$, $\delta > 0$, and any concept space (C, P) with $NB_4(C, \mathrm{P}) < \infty$: Procedure* `Sfoc` *observes an average training sample size of at most*

$$\mathrm{E}\, T_{\texttt{Sfoc}}(C, \mathrm{P}, \epsilon, \delta) \;<\; \frac{13.0356}{\epsilon}\left(\ln NB_4(C, \mathrm{P}) + \ln\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta} + 1.4\right). \tag{3.2}$$

for any target concept $c \in C$.

Proof `Sfoc` runs in $S = \log_2(1/\epsilon)$ stages. Each stage i constructs a $2^{-(i+1)}$-cover, V_i, of a local $2^{-(i-1)}$-neighborhood, and calls `sprt`($h(x) \neq c(x)$, $2^{-(i+1)}$, 2^{-i}, $\delta/(|V_i|S)$, 0) for each $h \in V_i$. Since V_i is a $2^{-(i+1)}$-cover, we know it contains at least one $2^{-(i+1)}$-good hypothesis h_i. Therefore, from Lemma A.2 in Appendix A, we know that `sprt` eventually accepts this hypothesis wp1, and from Lemma A.4 we know that this takes an average number of training examples of at most

$$\mathrm{E}\, T_{\texttt{sprt}(\; h_i(x) \neq c(x),\; 2^{-(i+1)},\; 2^{-i},\; \delta/(|V_i|S),\; 0\;)} \;<\; 6.5178(2^i)\left(\ln\frac{|V_i|S}{\delta} + 1\right).$$

Finally, since $|V_i| \leq NB_4(C, \mathrm{P})$, the total number of training examples observed by `Sfoc` is bounded by:

$$\begin{aligned}
\mathrm{E}\, T_{\texttt{Sfoc}}(C, \mathrm{P}, \epsilon, \delta) &< \sum_{i=1}^{S} 6.5178(2^i)\ln\frac{e\, NB_4(C, \mathrm{P})\, S}{\delta} \\
&= 6.5178(2^{S+1} - 1)\ln\frac{e\, NB_4(C, \mathrm{P})\, S}{\delta} \\
&< \frac{13.0356}{\epsilon}\left(\ln NB_4(C, \mathrm{P}) + \ln\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta} - \ln\ln 2 + 1\right),
\end{aligned}$$

plugging in $S = \log_2(1/\epsilon)$. ▮

Proposition 3.18 *For any $0 < \alpha \leq 1$*

$$\begin{aligned}
N_\alpha(\mathsf{initials}, \mathsf{uniform}) &= \left\lceil \frac{1}{2\alpha} \right\rceil \\
NB_4(\mathsf{initials}, \mathsf{uniform}) &\leq 4
\end{aligned}$$

Proof ***Part 1:*** To show $N_\alpha(\mathsf{initials}, \mathsf{uniform}) \leq \lceil 1/(2\alpha) \rceil$, construct an α-cover of the space by choosing $\lceil 1/(2\alpha) \rceil$ concepts defined by endpoints $\{\alpha(2n-1) : n = 1, 2, ..., \lceil 1/(2\alpha) \rceil\}$. Clearly, these endpoints are spaced 2α apart and any endpoint x_c in $[0, 1]$ must be within at least α of some cover-point.

Part 2: To show $N_\alpha(\mathsf{initials}, \mathsf{uniform}) \geq \lceil 1/(2\alpha) \rceil$, notice that an initial segment h can α-cover at most an neighborhood of initial segment concepts defined by endpoints in $[x_h - \alpha, x_h + \alpha]$. This neighborhood has length at most 2α, and hence at least $\lceil 1/(2\alpha) \rceil$ endpoints are required to cover the entire interval $[0, 1]$.

Part 3: Finally, to show $NB_4(\mathsf{initials}, \mathsf{uniform}) \leq 4$, notice that the 4α-neighborhood, $B_{4\alpha}(c)$, of any initial concept c is defined by an interval of endpoints $[x_c - 4\alpha, x_c + 4\alpha]$. This neighborhood clearly has length at most 8α. Thus, an α-cover of this neighborhood can be constructed by choosing at most 4 initial segment concepts defined by endpoints $x_c - 3\alpha$, $x_c - \alpha$, $x_c + \alpha$, $x_c + 3\alpha$. ▮

Proposition 3.20 *For any $0 < \alpha < 1/d$,*

$$\left(\frac{1}{2e\alpha}\right)^d \quad \le \quad N_\alpha(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}) \quad \le \quad \left\lceil \frac{1}{2\alpha} \right\rceil^d$$
$$NB_4(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}) \quad \le \quad 1.4(6e)^d$$

Proof ***Part 1:*** $N_\alpha(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}) \le \lceil 1/(2\alpha)\rceil^d$: An obvious α-cover of this space can be constructed by choosing $\lceil 1/(2\alpha)\rceil$ endpoints $2\alpha/d$ apart at

$$\{\, (i-1+\alpha[2n-1])/d \;:\; n = 1, 2, ..., \lceil 1/(2\alpha)\rceil \,\}$$

for each subdomain $X_i = [(i-1)/d, i/d)$, and then composing d-π-$\mathsf{initial}$ concepts by independently choosing an endpoint from each subdomain. This forms an α-cover of the space, since for any d-π-$\mathsf{initial}$ concept, c, the distance between c's endpoint in subdomain X_i, x_c^i, and the nearest cover-point in X_i is at most α/d; which means the total distance between c and the nearest cover-concept can be at most $d \times \alpha/d = \alpha$. Since there are d independent subdomains with $\lceil 1/(2\alpha)\rceil$ points in each, this construction generates $\lceil 1/(2\alpha)\rceil^d$ cover-concepts in total.

Part 2: $N_\alpha(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform}) \ge (2e\alpha)^{-d}$: Here it will be useful to think of d-π-$\mathsf{initial}$ concepts c as tuples $\langle x_c^1, ..., x_c^d\rangle \in [0, 1/d]^d$. It is easy to see that the total volume of this space is $(1/d)^d$. We will show that the volume of any α-neighborhood of a tuple $\langle x_c^1, ..., x_c^d\rangle \in [0, 1/d]^d$ is bounded by $(2e\alpha/d)^d$, and therefore any α-covering of $(d\text{-}\pi\text{-}\mathsf{initials}, \mathsf{uniform})$ by d-π-$\mathsf{initial}$ concepts requires at least a number of distinct concepts

$$N_\alpha \;\ge\; \frac{\left(\frac{1}{d}\right)^d}{\left(\frac{2e\alpha}{d}\right)^d} \;=\; \left(\frac{1}{2e\alpha}\right)^d .$$

To prove this, fix an arbitrary tuple $\langle x_c^1, ..., x_c^d\rangle \in [\alpha, 1/d - \alpha]^d$. We wish to calculate the volume of the α-neighborhood of the concept c defined by this tuple. Note that we have

$$d_{\mathsf{U}}(c, h) \le \alpha \quad \Longleftrightarrow \quad |x_c^1 - x_h^1| + ... + |x_c^d - x_h^d| \le \alpha \tag{B.6}$$

for any other d-π-$\mathsf{initial}$ concept h. Therefore, the α-neighborhood of c consists of all d-π-$\mathsf{initial}$ concepts h defined by tuples $\langle x_h^1, ..., x_h^d\rangle = \langle x_c^1 + \xi_1, ..., x_c^d + \xi_d\rangle$ where $-\alpha \le \xi_i \le \alpha$ and $|\xi_1| + ... + |\xi_d| \le \alpha$. So all we need to do is determine the volume of the simplex

$$R_\alpha \;=\; \left\{ \langle \xi_1, ..., \xi_d\rangle \in [-\alpha, \alpha]^d \;:\; \sum_{i=1}^d |\xi_i| \le \alpha \right\}.$$

Note that R_α can be decomposed into 2^d disjoint subregions, corresponding to the ways a sign $(+1, -1)$ can be chosen for each ξ_i:

$$\begin{aligned} R_\alpha \;&=\; \sum_{\vec\sigma \in \{+1,-1\}^d} \vec\sigma \odot \left\{ \langle \xi_1, ..., \xi_d\rangle \in [0, \alpha]^d : \sum_{i=1}^d \xi_i \le \alpha \right\} \\ &\triangleq\; \sum_{\vec\sigma \in \{+1,-1\}^d} \vec\sigma \odot r_\alpha, \end{aligned}$$

where $\odot$ means coordinate-wise multiplication: $\langle x_1, ..., x_k\rangle \odot \langle y_1, ..., y_k\rangle = \langle x_1y_1, ..., x_ky_k\rangle$. Thus, the total volume of R_α is just 2^d times the volume of a component simplex r_α. By Lemma B.3(a) below we know that $vol(r_\alpha) = \alpha^d/(d!)$, and therefore

$$vol(R_\alpha) \;=\; \frac{2^d\alpha^d}{d!} \;<\; \frac{(2d)^d}{\left(\frac{d}{e}\right)^d} \;=\; \left(\frac{2e\alpha}{d}\right)^d,$$

since $d! > (d/e)^d$ by Stirling's approximation [Purdom and Brown, 1985, Section 4.5.5].

Part 3: NB_4(d-π-initials, uniform) $\leq 1.4(6e)^d$: Again, it will be useful to think of d-π-initial concepts as tuples $\langle x_c^1, ..., x_c^d \rangle \in [0, 1/d]^d$. For a d-π-initial concept c we can construct an α-cover of its 4α-neighborhood as follows. Recall from (B.6) that the 4α-neighborhood of c consists of all d-π-initial concepts h defined by tuples $\langle x_h^1, ..., x_h^d \rangle = \langle x_c^1 + \xi_1, ..., x_c^d + \xi_d \rangle$, where $-4\alpha \leq \xi_i \leq 4\alpha$ and $|\xi_1| + ... + |\xi_d| \leq 4\alpha$. So we construct an α-cover of the simplex

$$\begin{aligned} R_{4\alpha} &= \left\{ \langle \xi_1, ..., \xi_d \rangle \in [-4\alpha, 4\alpha]^d \ : \ \sum_{i=1}^d |\xi_i| \leq 4\alpha \right\} \\ &= \sum_{\vec{\sigma} \in \{+1,-1\}^d} \vec{\sigma} \odot \left\{ \langle \xi_1, ..., \xi_d \rangle \in [0, 4\alpha]^d \ : \ \sum_{i=1}^d \xi_i \leq 4\alpha \right\} \\ &\triangleq \sum_{\vec{\sigma} \in \{+1,-1\}^d} \vec{\sigma} \odot r_{4\alpha}. \end{aligned}$$

To construct an α-cover of a component simplex $r_{4\alpha}$: Take an α/d-cover of $[0, 4\alpha]$ for each coordinate $i = 1, ..., d$ which consists of the $2d$ points $\{\alpha(2n-1)/d : n = 1, ..., 2d\}$. Then construct cover-tuples $\langle \xi_1, ..., \xi_d \rangle$ by choosing an endpoint from each coordinate-cover; but maintaining the overall constraint that $\sum_i \xi_i \leq 5\alpha$. This forms an α-cover of $r_{4\alpha}$ because for any tuple $\langle x^1, ..., x^d \rangle \in r_{4\alpha}$, each component x^i is at most α/d away from the nearest cover-point ξ_i in coordinate i, and by (B.6) this means $\langle x^1, ..., x^d \rangle$ can be at most $d \times \alpha/d = d$ away from the nearest cover-tuple $\langle \xi_1, ..., \xi_d \rangle$.[2]

It remains to count the number of tuples $\langle \xi_1, ..., \xi_d \rangle$ satisfying the constraints that $\xi_i \in \{\alpha(2n-1)/d : n = 1, ..., 2d\}$ and $\sum_i \xi_i \leq 5\alpha$. To make the counting problem easier, consider the transformed set of tuples $\langle \zeta_1, ..., \zeta_d \rangle$ where $\zeta_i = (d\xi_i/\alpha + 1)/2$. This equinumerous set contains tuples $\langle \zeta_1, .., \zeta_d \rangle$ such that $\zeta_i \in \{1, ..., 2d\}$ and $\sum_i \zeta_i \leq (5d+1)/2$. Now, *enlarge* this set to contain tuples $\langle \zeta_1, ..., \zeta_d \rangle$ such that $\zeta_i \in \{1, ..., 3d\}$ and $\sum_i \zeta_i \leq 3d$. By Lemma B.3(b) below, the number of tuples in this larger integer simplex is bounded by

$$\begin{aligned} \sum_{n_1 > 0, ..., n_d > 0} 1_{\{n_1 + ... + n_d \, \leq \, 3d\}} &< \frac{(3d+1)^d}{d!} \\ &< \frac{(3d+1)^d}{(d/e)^d} \\ &= (3e)^d \left(1 + \frac{1}{3d}\right)^d \ < \ 1.4(3e)^d. \end{aligned} \tag{B.7}$$

Finally, since $R_{4\alpha}$ consists of 2^d isomorphic copies of $r_{4\alpha}$, repeating the construction for each $r_{4\alpha}$ yields an α-cover of $R_{4\alpha}$ with size less than $2^d \cdot 1.4(3e)^d = 1.4(6e)^d$. ▌

Lemma B.3 (a) *For $z > 0$ and any integer $k > 0$,*

$$\int_{z_1 \geq 0, \, z_2 \geq 0, \, ..., \, z_k \geq 0} 1_{\{z_1 + z_2 + \cdots + z_k \, \leq \, z\}} \ dz_1 \, dz_2 \ldots dz_k \ = \ \frac{z^k}{k!}.$$

(b) *For integers $n_1, n_2, ..., n_k$, and n, $k > 0$,*

$$\sum_{n_1 > 0, \, n_2 > 0, \, ..., \, n_k > 0} 1_{\{n_1 + n_2 + \cdots + n_k \, \leq \, n\}} \ < \ \frac{(n+1)^k}{k!}.$$

[2]Note that this tuple $\langle \xi_1, ..., \xi_d \rangle$ is guaranteed to be in the cover for any $\langle x^1, ..., x^d \rangle \in r_{4\alpha}$, since the nearest potential cover-tuple to $\langle x^1, ..., x^d \rangle$ must have $\sum_i \xi_i \leq \sum_i (x^i + \alpha/d) \leq 4\alpha + \alpha = 5\alpha$, and hence gets included in the construction.

Proof We prove (a) by induction on k. *Base case*: For $k=1$ it is obvious that $\int_{z_1\geq 0} 1_{\{z_1\leq z\}}dz_1 = z$. *Induction step*: Assume the relation holds for numbers up to $k-1$, then compute

$$\begin{aligned}
&\int_{z_1\geq 0,\ldots,z_k\geq 0} 1_{\{z_1+\cdots+z_k\,\leq\, z\}}\ dz_1\ldots dz_k \\
&= \int_0^z \left(\int_{z_1\geq 0,\ldots,z_{k-1}\geq 0} 1_{\{z_1+\cdots+z_{k-1}\,\leq\, z-z_k\}}\ dz_1\ldots dz_{k-1}\right) dz_k \\
&= \int_0^z \left(\frac{(z-z_k)^{k-1}}{(k-1)!}\right) dz_k \qquad \text{(by induction hypothesis)} \\
&= -\frac{(z-z_k)^k}{k!}\bigg|_0^k = \frac{z^k}{k!}.
\end{aligned}$$

Part (b) follows as a simple consequence of the fact that, for integers $n_1, n_2, \ldots, n_k$,

$$\sum_{n_1>0,\ldots,n_k>0} 1_{\{n_1+\cdots+n_k\,\leq\, n\}} < \int_{z_1\geq 0,\ldots,z_k\geq 0} 1_{\{z_1+\cdots+z_k\,\leq\, n+1\}}\ dz_1\ldots dz_k. \blacksquare$$

Proposition 3.22 `Sfoc` *can be implemented to solve* (d-π-initials, uniform, ϵ, δ) *in time*

$$O\left(\frac{d^2}{\epsilon}\left(d\ln d + \ln\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right)\right).$$

Proof The key to finding an efficient implementation of Procedure `Sfoc` (Figure 3.3) is to execute each stage in polynomial time. Recall that at each Stage s, Procedure `Sfoc` calls `sprt`($h(x)\neq c(x)$, α, 4α, $\delta/(|V_s|S)$, 0) for every hypothesis in some α-cover V_s of a 4α-neighborhood, $\alpha = 2^{-(s+1)}$. The problem with the space (d-π-initials, uniform) is that the size of the local α-covers we can construct is exponential in d (*e.g.*, in Proposition 3.20 (Part 3) above). So we cannot possibly implement `Sfoc` efficiently by constructing these covers explicitly. Fortunately, for (d-π-initials, uniform) it is possible to implicitly test every concept in this cover in polynomial time.

The idea is to notice that by (B.6) the error of a d-π-initial hypothesis h is just the *sum* of its errors in each subdomain $|x_c^1 - x_h^1| + \cdots + |x_c^d - x_h^d|$. So if we construct local covers that permit us to test the errors on each subdomain independently, we should be able to derive a polynomial time implementation. Procedure `NBtest` (Figure B.1) does just this: given a hypothesis h, `NBtest` implicitly constructs an α-cover of h's 4α-neighborhood and decides in polynomial time whether any concept in the cover makes fewer than M misclassifications on a given training sequence $c\mathbf{x}^t$.

This procedure can be used to implement each stage of Procedure `Sfoc` by calling `NBtest`($h_{s-1}, \alpha, M_t, c\mathbf{x}^t$) with $\alpha = 2^{-(s+1)}$ and

$$M_t = \frac{t\ln\frac{1-\alpha}{1-4\alpha} - \ln\frac{|V_s|S}{\delta}}{\ln\frac{1-\alpha}{1-4\alpha} + \ln 4}$$

after each training example, where $|V_s|$ is the size of the local cover `NBtest` constructs at Stage s. This mimics the behavior of calling `sprt` on every concept in an α-cover of h_{s-1}'s 4α-neighborhood, V_s, since the call to `sprt` halts as soon as some cover-concept makes a number of mistakes M (in t training examples) that causes

$$M\ln\frac{1}{4} + (t-M)\ln\frac{1-\alpha}{1-4\alpha} \geq \ln\frac{|V_s|S}{\delta}.$$

So, given the correctness and polynomial time efficiency of `NBtest`, we are done.

Correctness: First, in Step 1 `NBtest` explicitly constructs an α/d-cover for each of the 4α-neighborhoods of h's endpoints. The cross-product of these subdomain covers then implicitly defines an α-cover of h's

Procedure `NBtest` $(h, \alpha, M, c\mathbf{x}^T)$

INPUT: d-π-initial hypothesis h,
cover parameter $\alpha > 0$,
mistake threshold M,
training sequence $c\mathbf{x}^T$.

CONSIDERS: a fixed α-cover, V, of h's 4α-neighborhood $B_{4\alpha}(h)$.

RETURN: a d-π-initial concept $h^* \in V$ that makes fewer than M mistakes on $c\mathbf{x}^T$; "NONE" if no such concept exists in V.

PROCEDURE:

1. (Initialize.) For each subdomain X_i, fix the $4d$ endpoints

$$V_i \;=\; x_h^i + \{\, -(4\alpha - \alpha/d),\ ...,\ -3\alpha/d,\ \ -\alpha/d,\ \ \alpha/d,\ \ 3\alpha/d,\ ...,\ 4\alpha - \alpha/d \,\}.$$

   The mistake counts for each of these endpoints is stored in a subdomain array

$$M_i[-(4d-1), ..., -3, -1, 1, 3, ..., (4d-1)].$$

2. (Tally the errors.) For each training example $\langle x_t, c(x_t)\rangle$, $t = 1, ..., T$:

   The domain object x_t must "land" in some subdomain, say X_{i_t} (*i.e.*, $x_t \in X_{i_t}$), so consider the set of endpoints $V_{i_t} \subset X_{i_t}$ (*i.e.*, consider the array M_{i_t}).

   - If $c(x_t) = 0$, increment the mistake count $M_{i_t}[j]$++ for every $v_j \geq x_t$.
   - If $c(x_t) = 1$, increment the mistake count $M_{i_t}[j]$++ for every $v_j < x_t$.

3. (Find the cover-concept with minimum error.) For each dimension $i = 1, ..., d$, find the $v_{j_i} \in V_i$ with minimum mistake count $M_i[j_i]$. Let h^* be the d-π-initial concept defined by these minimum error endpoints $\langle v_{j_1}, ..., v_{j_d}\rangle$.

4. (Decide.) If $\sum_{i=1}^d M_i[j_i] < M$ return h^*, else return "NONE."

Figure B.1: Procedure `NBtest`

4α-neighborhood (since the resulting set is a superset of the cover constructed in Proposition 3.20 (Part 3) above). In Step 2, `NBtest` then counts the number of mistakes each endpoint makes in each subdomain. Finally in Step 3, `NBtest` considers the cover-concept h^* defined by the minimum error endpoints in each subdomain. This concept is guaranteed to be the minimum error cover-concept because the errors in each subdomain are additive. That is, some cover-concept beats the mistake bound if and only if h^* beats it.

Efficiency: Note that an obvious incremental version of `NBtest` runs in $O(d^2)$ time per training example. So we can use this procedure to implement `Sfoc` in time $O(d^2)$ times the number of training examples observed. From Proposition 3.16, noting that `NBtest` constructs local covers of size $NB_4 = (4d)^d$, we can determine that `Sfoc` observes at most $O\left(1/\epsilon\,[d\ln d + \ln\ln(1/\epsilon) + \ln(1/\delta)]\right)$ training examples on average. This gives an overall (expected) running time of $O\left([d^2/\epsilon]\cdot[d\ln d + \ln\ln(1/\epsilon) + \ln(1/\delta)]\right)$. ▮

Proposition 3.23 *For* $\alpha = 2^{-k}$,

$$\sum_{i=0}^{\log_2(1/\alpha)-1}\binom{n}{i} \;\le\; N_\alpha(\mathsf{monomials, uniform}) \;\le\; \sum_{i=0}^{\log_2(1/\alpha)}\binom{n}{i} \;\le\; \left(\frac{en}{\log_2(1/\alpha)}\right)^{\log_2(1/\alpha)}$$

Proof ***Part 1:*** To show $N_\alpha(\mathsf{monomials, uniform}) \le \sum_{i=0}^{\log_2(1/\alpha)}\binom{n}{i}$, construct an α-cover of the space consisting of $\mathsf{monomial}$ concepts that contain $\log_2(1/\alpha)$ or fewer attributes; *i.e.*, all c_i such that $|c_i| \le \log_2(1/\alpha)$. To see that this is indeed an α-cover of $(\mathsf{monomials, uniform})$, consider any $\mathsf{monomial}$ concept c_2 not in the cover. By construction c_2 contains at least $\log_2(1/\alpha)+1$ attributes and must be the superset of some cover-concept c_1. By Proposition 3.38 (Part 2) below this implies $d_{\mathsf{U}}(c_1, c_2) < \alpha$, since c_1 contains at most $\log_2(1/\epsilon)$ attributes. Therefore, this collection is an α-cover. Obviously the cover has size $\sum_{i=0}^{\log_2(1/\alpha)}\binom{n}{i}$, and Blumer *et al.* [1989, Proposition A2.1(iii)] show that $\sum_{i=0}^{d}\binom{n}{i} \le (en/d)^d$.

Part 2: To show $N_\alpha(\mathsf{monomials, uniform}) \ge \sum_{i=0}^{\log_2(1/\alpha)-1}\binom{n}{i}$, construct a collection of $\mathsf{monomial}$ concepts that are pairwise separated by 2α (a so-called "2α-packing" of the space). This will determine a lower bound on N_α since any concept that is within α^- of some packing-concept cannot be within α^- of any other packing-concept, by the triangle inequality. This means that any α^--cover of the space must include a distinct cover-concept for each packing-concept.

By Proposition 3.38 (Part 1) below, we know that any two concepts c_1 and c_2 of size $|c_i| \le \log_2(1/\alpha) - 1$ are separated by $d_{\mathsf{U}}(c_1, c_2) \ge 2\alpha$. Therefore, the collection of all $\mathsf{monomial}$ concepts that contain $\log_2(1/\alpha)-1$ or fewer attributes constitutes a 2α-packing of the space. Clearly, this collection has size $\sum_{i=0}^{\log_2(1/\alpha)-1}\binom{n}{i}$. ▮

Proposition 3.24 *For* $\alpha = 2^{-k} > 2^{-n-2}$, *the smallest* $\mathsf{monomial}$ $c_n = \{a_1, ..., a_n\}$ *has*

$$N_\alpha B_{4\alpha}(c_n) \;>\; \binom{n}{\log_2(1/\alpha)-1} \;=\; \Omega\left(\frac{n}{\ln(1/\alpha)}\right)^{\ln(1/\alpha)}.$$

Proof From (3.3) we know that the distance between c_n and any other $\mathsf{monomial}$ concept c is given by $d_{\mathsf{U}}(c_n, c) = 2^{-|c|} - 2^{-n}$. Therefore, assuming $\alpha > 2^{-n-2}$, the 4α-neighborhood of c_n consists of all $\mathsf{monomial}$ concepts that contain $\log_2(1/\alpha) - 2$ or more attributes. (To see this, note that $|c| \ge \log_2(1/\alpha) - 2$ implies $d_{\mathsf{U}}(c_n, c) \le 4\alpha$, but $|c| \le \log_2(1/\alpha) - 3$ implies $d_{\mathsf{U}}(c_n, c) \ge 8\alpha - 2^{-n} > 8\alpha - 4\alpha = 4\alpha$ for $\alpha > 2^{-n-2}$.) Now, consider a strict subset of c_n's 4α-neighborhood consisting of the $\mathsf{monomial}$ concepts defined by exactly $\log_2(1/\alpha) - 1$ attributes. Notice that any two distinct concepts in this set have $|c_1 \cup c_2| \ge |c_1| + 1$, and therefore $d_{\mathsf{U}}(c_1, c_2) \ge 2\cdot 2^{-|c_1|} - 2\cdot 2^{-|c_1|-1} = 2^{-|c_1|} = 2\alpha$ by (3.3). So this set forms a 2α-packing within $B_{4\alpha}(c_n)$, and hence any α^--cover of $B_{4\alpha}(c_n)$ requires at least $\sum_{i=0}^{\log_2(1/\alpha)-1}\binom{n}{i}$ distinct concepts. ▮

Proposition 3.27 (D.f. cac-learning impossible) *For any $0 < \epsilon < 1$, and any concept class C containing two non–mutually-exclusive concepts: Any learner L must fail to meet the cac(ϵ)-criterion for some target concept in C, for some domain distribution* P.

Proof Consider two concepts c_1 and c_2 from C that agree on at least one domain object x_0 and disagree on at least one domain object x_1. Fix an arbitrary $\epsilon > 0$. We want to show that no learner can meet the cac(ϵ)-criterion for both c_1 and c_2 for every possible distribution on $\{x_0, x_1\}$.

First, consider a distribution P_1 such that $\mathrm{P}_1\{x_0\} > 0$ and $\mathrm{P}_1\{x_1\} > \epsilon$. For this distribution we have $d_{\mathrm{P}_1}(c_1, c_2) > \epsilon$, and therefore to meet the cac(ϵ)-criterion a learner cannot guess c_1 with non-zero probability when c_2 is the target, nor guess c_2 with non-zero probability when c_1 is the target. However, by construction we also have $\mathrm{P}_1\{c_1\mathbf{x}^t = c_2\mathbf{x}^t\} > 0$ for all t. Therefore, to meet the cace-criterion with respect to P_1, L must halt with zero probability given $\{c_1\mathbf{x}^t = c_2\mathbf{x}^t\}$ for all t.

Now, consider a distribution P_2 such that $\mathrm{P}_2\{x_0\} = 1$. For this distribution we have $\mathrm{P}_2\{c_1(x) = c_2(x)\} = 1$, and therefore to meet the cac(ϵ)-criterion with respect to P_2 the learner must halt with non-zero probability given $\{c_1\mathbf{x}^t = c_2\mathbf{x}^t\}$ for some $t < \infty$ (returning either c_1 or c_2). This shows that no learner can meet the cac(ϵ)-criterion for both c_1 and c_2 with respect to both distributions P_1 and P_2. ▮

Proposition 3.28 (Fixed-sample-size cac-learning impossible) *For any $0 < \epsilon < 1$, and any space (C, P) that contains two concepts c_1 and c_2 separated by $\epsilon < d_{\mathrm{P}}(c_1, c_2) < 1$: Any fixed-sample-size learner L must fail to meet the cac(ϵ)-criterion for some $c \in C$.*

Proof Fix an arbitrary $\epsilon > 0$ and choose two concepts c_1 and c_2 in (C, P) such that $\epsilon < d_{\mathrm{P}}(c_1, c_2) < 1$. Since $d_{\mathrm{P}}(c_1, c_2) > \epsilon$, a learner L cannot guess c_1 with positive probability when c_2 is the target, nor guess c_2 with positive probability when c_1 is the target (if L is to meet the cac(ϵ)-criterion for both concepts). But we also have $\mathrm{P}_{X^t}\{c_1\mathbf{x}^t = c_2\mathbf{x}^t\} > 0$ for any t; *i.e.*, these two concepts remain undistinguished with positive probability given any finite sequence of training examples.

Now consider an arbitrary fixed-sample-size learner L. We will have $T_L(C, \mathrm{P}, \epsilon) = t$ for some $t < \infty$, and hence $\mathrm{P}_{X^t}\{c_1\mathbf{x}^t = c_2\mathbf{x}^t\} > 0$ for this t. Therefore, at termination the training sample fails to distinguish between c_1 and c_2 with non-zero probability. Here, L must guess c_1 wp1 to meet the cac(ϵ)-criterion for c_1, but then L must fail to meet the cac(ϵ)-criterion for c_2 since it returns an ϵ-bad hypothesis with nonzero probability. ▮

Proposition 3.30 (Correctness) *For any $\epsilon > 0$, and any concept space (C, P) with $R_\epsilon(C, \mathrm{P}) < \infty$:* `Scov` *halts wp1 and meets the cac(ϵ)-criterion for every target $c \in C$.*

Proof Clearly, any hypothesis returned by `Scov` is guaranteed to be ϵ-accurate by construction, so it suffices to show that `Scov` halts wp1. Fix an arbitrary $\epsilon > 0$ and an arbitrary target concept $c \in C$, and let $r = R_\epsilon(C, \mathrm{P})$ and $p = P_\epsilon(C, \mathrm{P}, r, c)$. By hypothesis we have that $r < \infty$ and $p > 0$. Recall from Definition 3.29 in Section 3.5.2 that this means all ϵ-bad concepts are eliminated from C with probability at least p after r training examples.

A crude upper bound on $T_{\texttt{Scov}}$ can be obtained by breaking the training sequence $\mathbf{x}$ into blocks $\mathbf{x}_1^r, \mathbf{x}_2^r, \ldots$ consisting of r examples each. That is, consider an alternative learning procedure `Sblock` that runs `Scov` on blocks of r training examples; returning `Scov`'s hypothesis if it halts, but starting over again on the next block if `Scov` fails. Let $R_{\texttt{Sblock}}$ denote the number of blocks `Sblock` observes before terminating. Then clearly

$$T_{\texttt{Scov}}(\mathbf{x}) \quad \leq \quad r \cdot R_{\texttt{Sblock}}(\mathbf{x}). \tag{B.8}$$

Since `Scov` terminates with probability p on each block, we have $R_{\texttt{Sblock}} \sim \mathsf{geometric}(p)$, and hence $R_{\texttt{Sblock}} < \infty$ wp1. By (B.8) this implies $T_{\texttt{Scov}} < \infty$ wp1 as well. ▮

Proposition 3.31

1. $\text{VC}(C) < \infty$ *implies* $R_\epsilon(C, \text{P}) < \infty$ *for all* $\epsilon > 0$.

   (In fact, $\text{VC}(C) < \infty$ *if and only if for all* $\epsilon > 0$ *there exists an* $r < \infty$ *and* $p > 0$ *such that* $P_\epsilon(C, \text{P}, r) \geq p$ *for all domain distributions* P.*)*

2. *There are spaces* (C, P) *for which* $\text{VC}(C) = \infty$ *and yet* $R_\epsilon(C, \text{P}) < \infty$.

Proof Recall the definition of $R_\epsilon(C, \text{P})$ given in Definition 3.29.

Part 1: We actually prove the stronger equivalence stated in parentheses.

($\Rightarrow$) Assume $\text{VC}(C) < \infty$. Fix an arbitrary $\epsilon > 0$ and choose any $\delta < 1$. By Theorem 2.5 [Shawe-Taylor, Anthony and Biggs, 1993] we have that $r = T_{STAB}(C, \epsilon, 1-p)$ random training examples are sufficient to eliminate every ϵ-bad concept from C with probability at least p, regardless of the domain distribution P. This shows that $P_\epsilon(C, \text{P}, R_\epsilon) \geq p$ by Definition 3.29.

($\Leftarrow$) Fix an arbitrary $\epsilon > 0$. By hypothesis, we know there exists an $r < \infty$ and $p > 0$ such that $P_\epsilon(C, \text{P}, r) \geq p$ for every domain distribution P; *i.e.*, every ϵ-bad concept is eliminated from C after r training examples with probability at least p, regardless of the domain distribution P. Therefore, if we train on k blocks of r training examples, the probability that some ϵ-bad concept remains in C is at most $(1-p)^k$. This means that we can pac(ϵ, δ)-learn C under the d.f. model by collecting $k = (1/p)\ln(1/\delta)$ blocks of r training examples and running Procedure `F` on the total collection. But by Theorem 2.7 [Ehrenfeucht et al., 1989] this implies $\text{VC}(C) < \infty$.

Part 2: A simple example of a space where $\text{VC}(C) = \infty$ and yet $R_\epsilon(C, \text{P}) < \infty$ is ($\{$finite sets$\}$, uniform) on $[0, 1]$. Clearly, the class $\{$finite sets$\}$ has infinite VCdimension. However, every finite set has zero measure under the uniform distribution, and hence the space ($\{$finite sets$\}$, uniform) has zero diameter. This means that we automatically have $R_\epsilon(\{\text{finite sets}\}, \text{uniform}) = 0$ for any $\epsilon > 0$. ▮

Proposition 3.32

1. *If* $R_\epsilon(C, \text{P}) < \infty$ *for all* $\epsilon > 0$, *then* $N_\epsilon(C, \text{P}) < \infty$ *for all* $\epsilon > 0$.

2. *There are concept spaces* (C, P) *for which* $R_\epsilon(C, \text{P}) = \infty$ *and yet* $N_\epsilon(C, \text{P}) < \infty$.

Proof ***Part 1:*** This is obvious from learnability concerns: If $R_\epsilon(C, \text{P}) < \infty$ for $\epsilon > 0$ then we know `Scov` cac(ϵ)-learns (C, P) by Theorem 3.30, and hence `Scov` pac(ϵ, δ)-learns (C, P) for any δ. This means we have $N_{2\epsilon}(C, \text{P}) < \infty$ by Theorem 3.5 [Benedek and Itai, 1991].

Part 2: A trivial example of a space that is finitely coverable but not finitely reducible is ($\{$finite sets$\} \cup \{[0, 1]\}$, uniform) on $[0, 1]$. The collection $\{\varnothing, [0, 1]\}$ is an ϵ-cover of this space for any $\epsilon \geq 0$. On the other hand, any finite training sample leaves consistent concepts a distance 1 from $[0, 1]$, and therefore the space is not ϵ-reducible for any $\epsilon < 1$. ▮

Proposition 3.33 (Data-efficiency) *For any space* (C, P) *with* $R_\epsilon = R_\epsilon(C, \text{P}) < \infty$:

$$\mathrm{E}\, T_{\texttt{Scov}}(C, \text{P}, \epsilon) \;\leq\; \frac{R_\epsilon}{P_\epsilon(C, \text{P}, R_\epsilon)}. \tag{3.4}$$

Proof Recall from (B.8) in the proof of Theorem 3.30 above that $T_{\texttt{Scov}} \leq r \cdot R_{\texttt{Sblock}}$ and $R_{\texttt{Sblock}} \sim$ geometric(p), for $r = R_\epsilon(C, \text{P})$ and $p = P_\epsilon(C, \text{P}, r)$. Since a geometric($p$) random variable has expected value $1/p$, we automatically get $\mathrm{E}\, T_{\texttt{Scov}} \leq r/p$. ▮

Theorem 3.34 (Optimality) *For any* ϵ, *and any (well behaved) concept space* (C, P) *with* $R_\epsilon(c, \text{P}) < \infty$: `Scov` *meets the cac(*$\epsilon$*)-criterion with optimal expected training sample size for any target concept* $c \in C$.

Proof Since Scov stops as soon as an ϵ-accurate hypothesis can be ensured, it seems intuitive that Scov should be optimally data-efficient for learning with certainty. To prove this, we show that if any learner L stops before Scov with non-zero probability for *any* target concept $c_0 \in C$, then L cannot meet the cac-criterion for every target concept in C.

Fix $\epsilon > 0$, and consider a learner $L = (T, H)$ such that $\mathrm{P}_{X^\infty}\{T[c_0\mathbf{x}] < T_{\texttt{Scov}}[c_0\mathbf{x}]\} > 0$ for some $c_0 \in C$. By construction of Scov, we know that the event $\{\mathbf{x} : T[c_0\mathbf{x}] < T_{\texttt{Scov}}[c_0\mathbf{x}]\}$ coincides with the event that L's final hypothesis $H[c_0\mathbf{x}^T]$ does not ϵ-approximate every consistent concept $c \in C$ (for otherwise Scov would already have stopped as well). That is,

$$
\begin{aligned}
&\{\mathbf{x} : T[c_0\mathbf{x}] < T_{\texttt{Scov}}[c_0\mathbf{x}]\} \\
&\quad = \left\{\mathbf{x} : \exists\, c' \in C \text{ such that } d_{\mathrm{P}}(H[c_0\mathbf{x}^T], c') > \epsilon \text{ and } c'\mathbf{x}^T = c_0\mathbf{x}^T\right\} && \text{(B.10)} \\
&\quad = \bigcup_{c' \in C} \left\{\mathbf{x} : d_{\mathrm{P}}(H[c_0\mathbf{x}^T], c') > \epsilon \text{ and } c'\mathbf{x}^T = c_0\mathbf{x}^T\right\}. && \text{(B.11)}
\end{aligned}
$$

By hypothesis, (B.11) occurs with non-zero probability, so it remains only to show that this event occurs with nonzero probability for at least one $c' \in C$.

Notice that if (B.11) were a *countable* union (*i.e.*, C a countable class) we would automatically be finished, since a countable union having non-zero measure implies one of the events in the union must have non-zero measure (and therefore there would be some $c' \in C$ such that $\mathrm{P}\left\{\mathbf{x} : d_{\mathrm{P}}(H[c'\mathbf{x}^T], c') > \epsilon\right\} > 0$, which means that L would not cac(ϵ)-learn c'). However, C can be uncountably infinite in general, so the union (B.11) is not necessarily countable. This raises the possibility that L might be able to distribute its failure probability among uncountably many hypotheses so as to yield a zero probability of failure for any one concept.[3] To prove that this is impossible, I need to assume the concept space is suitably well behaved.

Definition B.4 (Consistent-separability) *A concept space (C, P) is* consistently-separable *if C contains a countable subset $D \subset C$ such that for every $c \in C$ and $c\mathbf{x}^t$ there is a $d \in D$ consistent with $c\mathbf{x}^t$ that approximates c arbitrarily well under d_{P}. (I.e. for every $c \in C$, $\alpha > 0$, $t < \infty$, and $\mathbf{x}^t \in X^t$, there is a $d \in D$ such that $d\mathbf{x}^t = c\mathbf{x}^t$ and $d_{\mathrm{P}}(d, c) < \alpha$).*[4]

Given that (C, P) is consistently-separable, we can prove the theorem as follows. Assume $D \subset C$ is a separating subset for (C, P). Consider an arbitrary object sequence $\mathbf{x}$ in the event (B.10) and let $h = H[c_0\mathbf{x}^T]$. Since $\mathbf{x}$ is in (B.10) there must be some $c' \in C$ such that $c'\mathbf{x}^T = c_0\mathbf{x}^T$ and yet $d_{\mathrm{P}}(h, c') \geq \epsilon + \alpha$ for $\alpha > 0$. Then by the definition of D, there must be a $d \in D$ such that $d\mathbf{x}^T = c'\mathbf{x}^T$ and $d_{\mathrm{P}}(d, c') < \alpha$, and by the triangle inequality we have $d_{\mathrm{P}}(h, d) \geq d_{\mathrm{P}}(h, c') - d_{\mathrm{P}}(d, c') > \epsilon + \alpha - \alpha = \epsilon$ for this d. So we have shown that for any object sequence $\mathbf{x}$ in (B.10) there is a consistent concept in the separating subset $d \in D$ that is not ϵ-approximated by L's final hypothesis. *I.e.*,

$$
\begin{aligned}
&\{\mathbf{x} : T[c_0\mathbf{x}] < T_{\texttt{Scov}}[c_0\mathbf{x}]\} \\
&\quad = \left\{\mathbf{x} : \exists\, c' \in C \text{ such that } d_{\mathrm{P}}(H[c_0\mathbf{x}^T], c') > \epsilon \text{ and } c'\mathbf{x}^T = c_0\mathbf{x}^T\right\} \\
&\quad = \left\{\mathbf{x} : \exists\, d \in D \text{ such that } d_{\mathrm{P}}(H[c_0\mathbf{x}^T], d) > \epsilon \text{ and } d\mathbf{x}^T = c_0\mathbf{x}^T\right\} \\
&\quad = \bigcup_{d \in D} \left\{\mathbf{x} : d_{\mathrm{P}}(H[c_0\mathbf{x}^T], d) > \epsilon \text{ and } d\mathbf{x}^T = c_0\mathbf{x}^T\right\}, && \text{(B.12)}
\end{aligned}
$$

where now the union (B.12) is countable. Since (B.12) has non-zero probability by hypothesis, there must be some $d' \in D$ for which $\mathrm{P}\left\{\mathbf{x} : d_{\mathrm{P}}(H[d'\mathbf{x}^T], d') > \epsilon\right\} > 0$. That is, there must be some $d' \in C$ that is not cac(ϵ)-learned by L; proving the theorem. ▮

[3] Analogous to the way singletons have Lebesgue measure zero and yet the interval $[0, 1]$ has measure 1.

[4] This is a benign condition that is normally satisfied by all concept spaces encountered in practice. For example, any discrete space is consistently-separable, and so are uncountable spaces like (initials, uniform) and (d-π-initials, uniform) on $[0, 1]$, and even (halfspaces, uniform) on $[-1, 1]^n$; these can be consistently-separated by the class of concepts formed by rational coordinates.

Theorem 3.35 (Cac-learnability) *For a (well behaved) space (C, P): If $R_\epsilon(C,\mathrm{P}) = \infty$ for some $\epsilon > 0$, then no learner L can cac(ϵ)-learn (C, P) with a bounded expected training sample size for every $\epsilon > 0$.*

Proof Assume $R_{2\epsilon}(C,\mathrm{P}) = \infty$. We will show that Scov cannot cac(ϵ)-learn (C,P) with a bounded expected sample size at error level ϵ. This will prove the theorem, since by Theorem 3.34 above we know that no learner can stop before Scov and still meet the cac(ϵ)-criterion for every $c \in C$.

Assume there is a $t < \infty$ such that $\mathrm{E}\, T_{\text{Scov}} \le t$ for all $c \in C$. Then

$$\mathrm{P}_{X^\infty}\{T_{\text{Scov}}[c\mathbf{x}] \le 2t\} \quad > \quad \frac{1}{2} \tag{B.13}$$

for all $c \in C$, by Markov's inequality. Recall that, by construction, Scov halts as soon as the set of consistent concepts remaining in C can be ϵ-covered by a single hypothesis, which implies that every 2ϵ-bad concept must have been eliminated from C. Combined with (B.13) this means (C,P) is reduced to a 2ϵ-ball with probability at least $1/2$ after $2t$ training examples, regardless of the target concept $c \in C$. Recalling Definition 3.29, this implies that $P_{2\epsilon}(C,\mathrm{P},2t) > 1/2$ and hence $R_{2\epsilon}(C,\mathrm{P}) \le 2t$, contradicting the assumption that $R_{2\epsilon}(C,\mathrm{P}) = \infty$. ▮

Proposition 3.36 *For $0 < \epsilon < 1/2$, and any $c \in$ initials with $x_c \in [2\epsilon, 1-2\epsilon]$,*

$$T_{\text{Scov}}(\text{initials}, \text{uniform}, \epsilon) \quad \sim \quad \text{negative-binomial}(p = 2\epsilon,\ k = 2).$$

Proof This follows as a simple corollary to Proposition 3.37 (substituting $d = 1$). ▮

Theorem 3.37 *For any $0 < \epsilon < 1/(4d)$, and any target concept $c \in d$-π-initials defined by endpoints $x_c^1, ..., x_c^d$, where $x_c^i \in [\,(i-1)/d + 2\epsilon,\ i/d - 2\epsilon\,]$ for $i = 1, ..., d$,*

$$T_{\text{Scov}}(d\text{-}\pi\text{-initials}, \text{uniform}, \epsilon) \quad \sim \quad \text{negative-binomial}(p = 2\epsilon,\ k = 2d).$$

Proof Fix $\epsilon > 0$ and consider a target c satisfying the stated conditions. Now consider the "2ϵ-brackets" surrounding each endpoint of c: $I_i = [x_c^i - 2\epsilon,\ x_c^i)$ and $J_i = [x_c^i,\ x_c^i + 2\epsilon]$, $i = 1, ..., d$. Notice that each training example can "hit" at most one of these intervals, and moreover that the probability of hitting each interval is 2ϵ, and the location of any hit in an interval is uniformly distributed. Now recall that Scov halts exactly when

$$\sum_{i=1}^{d} (x_c^i - \text{"largest hit in } I_i\text{"}) \;+\; \sum_{i=1}^{d} (\text{"smallest hit in } J_i\text{"} - x_c^i) \quad \le \quad 2\epsilon.$$

Therefore, this is isomorphic to a situation where we have $k = 2d$ disjoint intervals $I_1, ..., I_k$ of length $\alpha = 2\epsilon$, where hits are uniformly distributed in each, and we stop as soon as

$$\sum_{i=1}^{k} \text{"size of smallest hit in } I_i\text{"} \quad < \quad \alpha. \tag{B.14}$$

To formalize this, let $I_i = [\alpha(i-1), \alpha i]$, $i = 1, ..., k$; and define the random variable

$$S(\mathbf{x}^t) \quad = \quad \sum_{i=1}^{k} \underline{Z}_i(\mathbf{x}^t),$$

where

$$Z_i(x) = \begin{cases} x - \alpha(i-1), & \text{if } x \in I_i, \\ \alpha, & \text{if } x \notin I_i. \end{cases} \tag{B.15}$$

Intuitively: $Z_i(x)$ is the size of a "hit" in I_i, $Z_i(x)|(x \in I_i) \sim \mathsf{uniform}[0,\alpha]$, and $\underline{Z}_i(\mathbf{x}^t) \triangleq \min_{x_j \in \mathbf{x}^t} Z_i(x_j)$ measures the size of the smallest hit in I_i given sequence $\mathbf{x}^t$; finally $S(\mathbf{x}^t)$ is the sum of the smallest hits from each I_i, $i = 1, ..., k$. In terms of this formalization, (B.14) above means that

$$\{\mathbf{x} : T_{\mathsf{Scov}} = t\} \;=\; \left\{\mathbf{x} : S(\mathbf{x}^t) < \alpha \text{ but } S(\mathbf{x}^m) \geq \alpha \text{ for all } m < t\right\}.$$

Therefore,

$$\begin{aligned}
& \mathrm{P}_{X^\infty}\{T_{\mathsf{Scov}} = t\} \\
= \;& \mathrm{P}_{X^t}\left\{S(\mathbf{x}^t) < \alpha \text{ but } S(\mathbf{x}^m) \geq \alpha \text{ for all } m < t\right\} && \text{(B.16)} \\
= \;& \mathrm{P}_{X^t}\left\{\begin{array}{c} \exists\ (\min)\ x_{n_1}, ..., x_{n_{k-1}} \in \mathbf{x}^t \text{ such that } S(\langle x_{n_1}, ..., x_{n_{k-1}}, x_t\rangle) < \alpha, \\ \text{but this holds for no other size } k \text{ subset of } \mathbf{x}^t \end{array}\right\} \\
= \;& \binom{t-1}{k-1}\mathrm{P}_{X^{t-k}}\left\{\begin{array}{c} \text{every observation } x \in \mathbf{x}^{t-k} \text{ is larger} \\ \text{than } x_{n_1}, ..., x_{n_{k-1}} \text{ in } I_{\ell_1}, ..., I_{\ell_{k-1}} \text{ and} \\ \text{larger than } \alpha - \sum_{j=1}^{k-1} Z_{\ell_j}(x_{n_j}) \text{ in } I_{\ell_t} \end{array}\middle|\, S(\mathbf{x}^k) < \alpha\right\}\mathrm{P}_{X^k}\left\{S(\mathbf{x}^k) < \alpha\right\}. && \text{(B.17)}
\end{aligned}$$

To explain: the event (B.16) means there must be an observation $x_i \in \mathbf{x}^t$ in each subdomain I_i to cause $S(\mathbf{x}^t) < \alpha$, and moreover this is caused for the first time by the last observation x_t (which lands in I_{ℓ_t}). Therefore, there must be *minimum* observations $x_{n_1}, ..., x_{n_{k-1}}$ in each subdomain $I_{\ell_1}, ..., I_{\ell_{k-1}}$ that, combined with x_t in I_{ℓ_t}, cause termination. Also, no other observation in I_{ℓ_t} besides x_t can be small enough to cause termination with $x_{n_1}, ..., x_{n_{k-1}}$. This is illustrated in Figure B.2.

Now consider the conditional probability in (B.17).

$$\begin{aligned}
& \mathrm{P}_{X^{t-k}}\left\{\begin{array}{c} \text{every observation } x \in \mathbf{x}^{t-k} \text{ yields} \\ Z_{\ell_j}(x) \geq Z_{\ell_j}(x_{n_j}), j = 1, ..., k-1, \\ \text{and } Z_{\ell_t}(x) \geq \alpha - S(\mathbf{x}^k) + Z_{\ell_t}(x_t) \end{array}\middle|\, S(\mathbf{x}^k) < \alpha\right\} \\
= \;& \mathrm{P}_X\left\{\begin{array}{c} Z_{\ell_j}(x) \geq Z_{\ell_j}(x_{n_j}), j = 1, ..., k-1, \\ \text{and } Z_{\ell_t}(x) \geq \alpha - S(\mathbf{x}^k) + Z_{\ell_t}(x_t) \end{array}\middle|\, S(\mathbf{x}^k) < \alpha\right\}^{t-k}
\end{aligned}$$

since the observations are independent,

$$\begin{aligned}
= \;& \left[1 - \mathrm{P}_X\left\{\begin{array}{c} \exists j \in 1, ..., k-1\ Z_{\ell_j}(x) < Z_{\ell_j}(x_{n_j}), \\ \text{or } Z_{\ell_t}(x) < \alpha - S(\mathbf{x}^k) + Z_{\ell_t}(x_t) \end{array}\middle|\, S(\mathbf{x}^k) < \alpha\right\}\right]^{t-k} && \text{(B.18)} \\
= \;& (1-\alpha)^{t-k}, && \text{(B.19)}
\end{aligned}$$

since the probability of the conditional event in (B.18) is obviously α, as shown in Figure B.2.

Next, consider the probability $\mathrm{P}_{X^k}\left\{S(\mathbf{x}^k) < \alpha\right\}$ in (B.17). Notice that $S(\mathbf{x}^t) < \alpha$ implies that $\mathbf{x}^t$ contains exactly one observation in each subdomain $I_1, ..., I_k$. Thus,

$$\begin{aligned}
\mathrm{P}_{X^k}\left\{S(\mathbf{x}^k) < \alpha\right\} \;&=\; k! \cdot \mathrm{P}_{X^k}\left\{Z_1(x_1) + \cdots Z_k(x_k) < \alpha \,|\, x_j \in I_j, j = 1, ..., k\right\} \\
&=\; k!\frac{\alpha^k}{k!} \;=\; \alpha^k, && \text{(B.20)}
\end{aligned}$$

by Lemma B.3(a) above (recalling that $Z_j(x)|(x \in I_j) \sim \mathsf{uniform}[0,\alpha]$ by (B.15)).

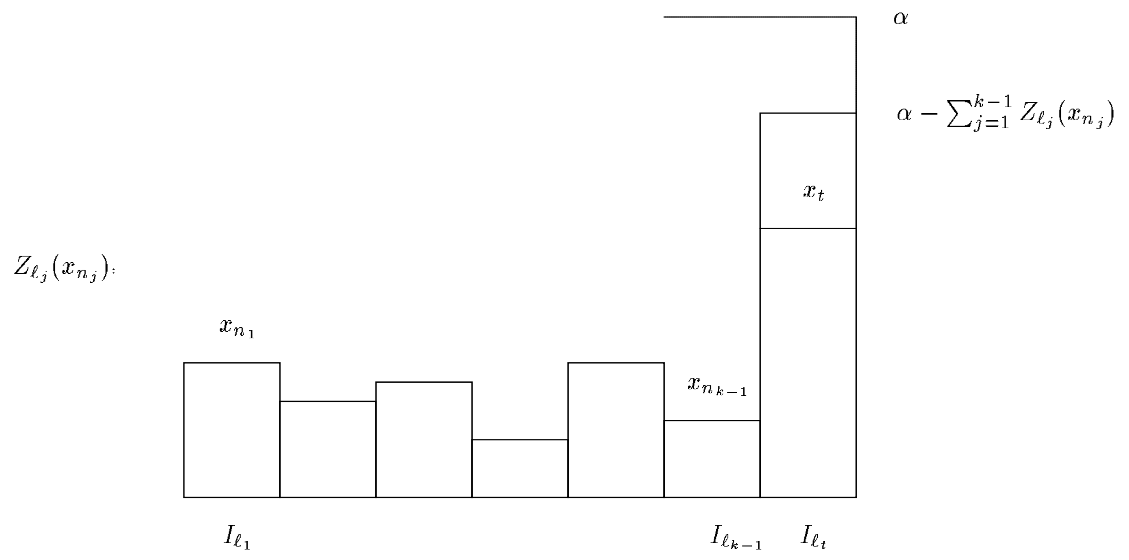

Figure B.2: For $\mathbf{x}^t$ to cause the stopping event for the first time at time t, every observation besides $x_{n_1}, ..., x_{n_{k-1}}, x_t$ must occur *outside* the indicated region in the subdomains $I_{\ell_1}, ..., I_{\ell_{k-1}}, I_{\ell_t}$.

So finally, combining (B.17), (B.19), and (B.20), we have

$$\mathrm{P}_{X^\infty}\{T_{\mathsf{Scov}} = t\} \;=\; \binom{t-1}{k-1}\alpha^k(1-\alpha)^{t-k},$$

which shows that $T_{\mathsf{Scov}} \sim$ negative-binomial$(p = \alpha = 2\epsilon, k = 2d)$. ▮

Proposition 3.38 *For distinct* monomial *concepts c_1 and c_2 (assuming $\epsilon = 2^{-k}$):*

1. *If $|c_1| \le \log_2(1/\epsilon) - 1$, then $d_{\mathsf{U}}(c_1, c_2) \ge \epsilon$ for all $c_2 \ne c_1$.*
2. *If $|c_1| = \log_2(1/\epsilon)$, then $d_{\mathsf{U}}(c_1, c_2) < \epsilon$ iff $c_1 \subset c_2$, but $d_{\mathsf{U}}(c_1, c_2) \ge \epsilon/2$ for all $c_2 \ne c_1$.*
3. *If $|c_1| \ge \log_2(1/\epsilon) + 1$, then $d_{\mathsf{U}}(c_1, c_2) < \epsilon$ if $|c_2| \ge \log_2(1/\epsilon) + 1$.*

Proof Recall that the distance d_{U} between any two monomial concepts c_1 and c_2 with respect to the uniform distribution is given by

$$d_{\mathsf{U}}(c_1, c_2) \;=\; 2^{-|c_1|} + 2^{-|c_2|} - 2 \cdot 2^{-|c_1 \cup c_2|}, \tag{3.3}$$

where c_1 and c_2 are thought of as sets of attributes. Two facts are then immediate:

Fact 1: If $c_1 \not\subset c_2$ then $d_{\mathsf{U}}(c_1, c_2) \ge 2^{-|c_1|}$; since $c_1 \not\subset c_2$ implies $|c_1 \cup c_2| \ge |c_2| + 1$.

Fact 2: If $c_1 \subset c_2$ then $d_{\mathsf{U}}(c_1, c_2) = 2^{-|c_1|} - 2^{-|c_2|}$; since $c_1 \subset c_2$ implies $|c_1 \cup c_2| = |c_2|$.

From these facts it is easy to prove each of the three relations as follows.

Part 1: Assume $|c_1| \le \log_2(1/\epsilon) - 1$. Consider the two cases $c_1 \not\subset c_2$ and $c_1 \subset c_2$. If $c_1 \not\subset c_2$, then from Fact 1 we have $d_{\mathsf{U}}(c_1, c_2) \ge 2^{-|c_1|}$, which immediately yields $d_{\mathsf{U}}(c_1, c_2) \ge 2\epsilon$. If on the other hand $c_1 \subset c_2$, then we must have $|c_2| \ge \log_2(1/\epsilon)$, and from Fact 2 we get $d_{\mathsf{U}}(c_1, c_2) \ge 2\epsilon - \epsilon = \epsilon$.

Part 2: Assume $|c_1| = \log_2(1/\epsilon)$. Consider the two cases $c_1 \not\subset c_2$ and $c_1 \subset c_2$. If $c_1 \not\subset c_2$, then from Fact 1 we automatically have $d_{\mathsf{U}}(c_1, c_2) \ge \epsilon$. On the other hand, if $c_1 \subset c_2$, then from Fact 2 we must have $d_{\mathsf{U}}(c_1, c_2) = \epsilon - 2^{-|c_2|} < \epsilon$. Also in this case we must have $d_{\mathsf{U}}(c_1, c_2) \ge \epsilon/2$, since $|c_2| \ge \log_2(1/\epsilon) + 1$.

Part 3: Assume $|c_1| \ge \log_2(1/\epsilon) + 1$ and $|c_2| \ge \log_2(1/\epsilon) + 1$. Clearly here we have $d_{\mathsf{U}}(c_1, c_2) \;\le\; \epsilon/2 + \epsilon/2 - 2 \cdot 2^{-|c_1 \cup c_2|} \;<\; \epsilon$. ▮

Theorem 3.39 *For $\epsilon < 1/2$ and any target concept $c \in$ monomials$\{0,1\}^n$,*

$$\mathrm{E}\, T_{\mathsf{Scov}}(\mathsf{monomials}, \mathsf{uniform}, \epsilon) \;\le\; \frac{2}{\epsilon}\left(\ln\frac{1}{\epsilon}\right)\log_2(en).$$

Proof We consider two cases: (1) the target monomial c is large (defined by $\log_2(1/\epsilon)$ or fewer attributes); and (2) c is small (defined by $\log_2(1/\epsilon)+1$ or more attributes).

Part 1: Assume $|c| \le \log_2(1/\epsilon)$. Note that in this case Scov's termination is guaranteed once every other monomial concept has been eliminated from the space. We derive an upper bound on the expected time for this to occur.

Without loss of generality, assume c is defined by conjoining the first s attributes attributes, $c = \{a_1, ..., a_s\}$, where $s \le \log_2(1/\epsilon)$. Recall that the domain $X = \{0,1\}^n$ consists of domain objects $x = \langle a_1, ..., a_n\rangle \in \{0,1\}^n$, and that $c(x) = 1$ if and only if $x = \langle 1, ..., 1, a_{s+1}, .., a_n\rangle$; *i.e.*, if and only if $a_{i_j} = 1$ on all indices specified by c.

1. Then the following set of examples is sufficient to eliminate every monomial h that is defined by a strict subset of the attributes $\{a_1, ..., a_s\}$ in c (*i.e.*, h is "too general" on some domain object x: $h(x) = 1$ when $c(x) = 0$):

$$\begin{array}{cccccccccc}
a_1 & a_2 & \ldots & a_s & & \ldots & & a_n & & \\
0 & 1 & \ldots & 1 & * & * & \ldots & * & \to & 0\\
1 & 0 & \ldots & 1 & * & * & \ldots & * & \to & 0\\
 & \vdots & & & & \vdots & & & & \\
1 & 1 & \ldots & 0 & * & * & \ldots & * & \to & 0.
\end{array}$$

   Notice that: there are s such patterns, each pattern is a *disjoint* event from the other such patterns, and each pattern occurs with probability $2^{-s} \ge \epsilon$. Therefore, by Lemma B.5 below, we have that the expected time to observe all such patterns is

$$\mathrm{E}\, T_{general} \;\le\; \frac{H_s}{\epsilon}. \tag{B.22}$$

2. Similarly, the following set of examples is sufficient to eliminate every monomial h that includes some attribute outside of c (*i.e.*, h is "too specific" on some domain object x: $h(x) = 0$ when $c(x) = 1$):

$$\begin{array}{cccccccccc}
a_1 & a_2 & \ldots & a_s & & \ldots & & a_n & & \\
1 & 1 & \ldots & 1 & 0 & * & \ldots & * & \to & 1\\
1 & 1 & \ldots & 1 & * & 0 & \ldots & * & \to & 1\\
 & \vdots & & & & \vdots & & & & \\
1 & 1 & \ldots & 1 & * & * & \ldots & 0 & \to & 1.
\end{array}$$

   Notice that: there are $n - s$ such patterns, each pattern is an *independent* event from the other such patterns given the initial 11...1 sequence, and each pattern occurs with probability 1/2 given the initial 11...1 sequence. Therefore, by Lemma B.6 below, we know that the expected time to observe all such patterns is

$$\begin{aligned}
\mathrm{E}\, T_{specific} &= 2^s \frac{H_{n-s}}{-ln(1/2)} \\
&\le \frac{H_{n-s}}{\epsilon \ln 2}.
\end{aligned} \tag{B.23}$$

Therefore, in total we get $T_{\mathsf{Scov}} \leq T_{general} + T_{specific}$. Combining (B.23) and (B.22) gives

$$\begin{aligned} \mathrm{E}\, T_{\mathsf{Scov}} &\leq \frac{1}{\epsilon}\left(H_s + \frac{H_{n-s}}{\ln 2}\right) \\ &< \frac{1}{\epsilon \ln 2}(H_{n-s} + H_s) \qquad \text{since } 1 < (\ln 2)^{-1}, \\ &= \frac{H_n}{\epsilon \ln 2} \leq \frac{\log_2 n}{\epsilon}. \qquad \text{(B.24)} \end{aligned}$$

Part 2: Assume $|c| \geq \log_2(1/\epsilon) + 1$. Note that in this case Scov's termination is guaranteed once every *large* monomial has been eliminated from the space. This is because, by Proposition 3.38(Part 3) above, the class of all small monomial concepts (defined by $\log_2(1/\epsilon) + 1$ or more attributes) has diameter less than ϵ, and therefore Scov can terminate by guessing any one of these. In fact, it suffices to eliminate the monomials defined by exactly $\log_2(1/\epsilon)$ attributes, since this will automatically eliminate every larger monomial defined by fewer attributes. Thus, we derive an upper bound on the expected time for this to occur.

Notice that any monomial h defined by $\log_2(1/\epsilon)$ attributes must be eliminated with probability at least $\epsilon/2$ on each training example. (This is because c's positive extension can have size at most $\epsilon/2$ by assumption, whereas h's positive extension must have size ϵ.) Also notice that there are $\binom{n}{\log_2(1/\epsilon)}$ monomial concepts defined by exactly $\log_2(1/\epsilon)$ attributes. Therefore, assuming (conservatively) that each large monomial is eliminated on a disjoint portion of the domain, Lemma B.5 below shows that

$$\begin{aligned} \mathrm{E}\, T_{\mathsf{Scov}} &\leq \frac{2}{\epsilon} \ln \binom{n}{\log_2(1/\epsilon)} \\ &< \frac{2}{\epsilon} \ln \left(\frac{en}{\log_2(1/\epsilon)}\right)^{\log_2(1/\epsilon)} \\ &< \frac{2}{\epsilon}\left(\ln \frac{1}{\epsilon}\right) \log_2(en), \qquad \text{for } \epsilon < 1/2. \qquad \text{(B.25)} \end{aligned}$$

Summary: For the two cases of *(i)* a large target concept, and *(ii)* a small target concept, we obtain the bounds (B.24) and (B.25) respectively, so the theorem holds in general. ▮

Lemma B.5 *If T is the time to observe k* disjoint *events of size p, then*

$$\mathrm{E}\, T = \frac{H_k}{p} \leq \frac{\ln k}{p},$$

where H_k is the kth harmonic number.

Proof Since the probability of observing any one of the k events is kp, the time T_1 to observe the first event is distributed geometric(kp) and hence $\mathrm{E}\, T_1 = 1/(kp)$. Similarly, given that the first event has been observed, the probability of observing any one of the remaining $k-1$ events is $(k-1)p$, and therefore the time T_2 to observe any one of these events is distributed geometric($(k-1)p$); giving $\mathrm{E}\, T_2 = 1/[(k-1)p]$. Continuing in this manner for $T = T_1 + T_2 + \ldots + T_i + \ldots + T_k$ we get

$$\begin{aligned} \mathrm{E}\, T &= \frac{1}{kp} + \frac{1}{(k-1)p} + \ldots + \frac{1}{(k-i+1)p} + \ldots + \frac{1}{2p} + \frac{1}{p} \\ &= \frac{1}{p}\left(\frac{1}{k} + \frac{1}{k-1} + \ldots + \frac{1}{k-i+1} + \ldots + \frac{1}{2} + 1\right) \\ &= \frac{H_k}{p}. \ ▮ \end{aligned}$$

Lemma B.6 *If T is the time to observe k* independent *events of size p, then*

$$\mathrm{E}\,T \;<\; \frac{H_k}{-\ln(1-p)} \;\leq\; \frac{\ln k}{-\ln(1-p)}.$$

Proof (This proof is due to Russell Greiner.) First consider:

$$\begin{aligned}
\mathrm{P}\{T \leq t\} &= \mathrm{P}\bigcap_{i=1}^{k}\{\text{ event } i \text{ observed by time } t\ \} \\
&= \prod_{i=1}^{k}\mathrm{P}\{\text{ event } i \text{ observed by time } t\ \} \qquad \text{(by independence)} \\
&= \prod_{i=1}^{k} 1-(1-p)^t \;=\; \left(1-q^t\right)^k,
\end{aligned}$$

where $q = 1-p$. Thus,

$$\begin{aligned}
\mathrm{P}\{T > t\} &= \mathrm{P}\{\text{ not all events observed by time } t\ \} \\
&= 1-(1-q^t)^k.
\end{aligned}$$

Now, to determine $\mathrm{E}\,T$, notice that

$$\begin{aligned}
\mathrm{E}\,T &= \sum_{t=1}^{\infty}\mathrm{P}\{T \geq t\} \\
&= \sum_{t=1}^{\infty} 1-(1-q^{t-1})^k \;<\; \int_0^{\infty} 1-(1-q^s)^k\, ds,
\end{aligned}$$

where $s = t-1$. Consider the transformation $u = 1-q^s$, giving $du = -(\ln q)(1-u)ds$, and also note that $u = 0$ for $s = 0$, and $u = 1$ for $s = \infty$ (since $q < 1$). Therefore,

$$\begin{aligned}
\mathrm{E}\,T &< \frac{1}{-\ln q}\int_0^1 \frac{1-u^k}{1-u}\, du \\
&= \frac{1}{-\ln q}\int_0^1 1+u+u^2+\ldots+u^{k-1}\, du \\
&= \frac{1}{-\ln q}\sum_{i=1}^{k}\frac{u^i}{i}\Bigg|_{u=0}^{1} \\
&= \frac{1}{-\ln q}\sum_{i=1}^{k}\frac{1}{i} - 0 \;=\; \frac{H_k}{-\ln q}. \quad \blacksquare
\end{aligned}$$

Proposition 3.43 *For* (initials, uniform)*: $\mathrm{P}_{x^\infty}\{T_{\mathtt{Scov}} > t\} < e^{-t\epsilon\left(1-\frac{1}{t\epsilon}\right)^2}$ for $t \geq 1/\epsilon$. Thus, for $\epsilon \leq e^{-3/2}$, $\delta \leq e^{-1/2}$, and using the cover constructed in Proposition 3.18, we get*

$$\mathrm{P}_{x^\infty}\{T_{\mathtt{Scov}} \;>\; T_{\mathtt{BI}}\} \;<\; (e\epsilon\delta)^{30}.$$

Proof This follows as a simple corollary to Proposition 3.44 below (setting $d = 1$). $\blacksquare$

Proposition 3.44 *For* (d-π-initials, uniform)*:* $\mathrm{P}_{X^\infty}\{T_{\mathsf{Scov}} > t\} < e^{-t\epsilon\left(1-\frac{d}{t\epsilon}\right)^2}$ *for* $t \geq d/\epsilon$. *Thus, for* $\epsilon \leq e^{-3/2}$, $\delta \leq e^{-d/2}$, *and using the cover constructed in Proposition 3.20, we get*

$$\mathrm{P}_{X^\infty}\{T_{\mathsf{Scov}} > T_{\mathtt{BI}}\} < (e\epsilon)^{30d}\delta^{30}.$$

Proof ***Part 1:*** Recall from Theorem 3.37 that we have $T_{\mathsf{Scov}} \sim$ negative-binomial$(p = 2\epsilon, k = 2d)$ for (d-π-initials, uniform).[5] By definition, the probability that a negative-binomial random variable T takes on the value t corresponds to the probability that the k^{th} success of a series of independent Bernoulli trials (with probability p of success) occurs on the t^{th} trial. Therefore, in our case

$$\begin{aligned}
\mathrm{P}_{X^\infty}\{T_{\mathsf{Scov}} > t\} &= \mathrm{P}\{\ 2d^{th} \text{ success after } t^{th} \text{ trial }\} \\
&= \mathrm{P}\{\text{ fewer than } 2d \text{ successes in } t \text{ trials }\} \\
&\leq e^{-t\epsilon\left(1-\frac{d}{t\epsilon}\right)^2} \qquad \text{(B.26)}
\end{aligned}$$

for $t \geq d/\epsilon$, using Chernoff bounds [Hagerup and Rüb, 1989/90].

Part 2: Recall from Figure 3.1 and Proposition 3.20 that

$$T_{\mathtt{BI}}(d\text{-}\pi\text{-initials, uniform}, \epsilon, \delta) \geq \frac{32}{\epsilon}\left(d \ln \frac{1}{e\epsilon} + \ln \frac{1}{\delta}\right).$$

Therefore, plugging in $T_{\mathtt{BI}}$ for t in (B.26) gives

$$\begin{aligned}
\mathrm{P}_{X^\infty}\{T_{\mathsf{Scov}} > T_{\mathtt{BI}}\} &\leq e^{-\epsilon T_{\mathtt{BI}}\left(1-\frac{d}{\epsilon T_{\mathtt{BI}}}\right)^2} \\
&\leq e^{-\epsilon T_{\mathtt{BI}}\left(1-\frac{1}{32}\right)^2} \\
&\qquad \text{since } d/(\epsilon T_{\mathtt{BI}}) \leq 1/32 \text{ for } \epsilon \leq e^{-3/2} \text{ and } \delta \leq e^{-d/2}, \\
&= e^{-32\left(d \ln \frac{1}{e\epsilon} + \ln \frac{1}{\delta}\right)\left(\frac{31}{32}\right)^2} \\
&\leq e^{-30\left(d \ln \frac{1}{e\epsilon} + \ln \frac{1}{\delta}\right)} = (e\epsilon)^{30d}\delta^{30}. \ \blacksquare
\end{aligned}$$

Proposition 3.46

$$\begin{aligned}
T_{\mathsf{Scut}}(\text{initials, uniform}, \epsilon, \delta) &\leq \frac{2}{\epsilon}\left(1 + \frac{1}{2}\ln\frac{1}{\delta}\right) \\
T_{\mathsf{Scut}}(d\text{-}\pi\text{-initials, uniform}, \epsilon, \delta) &\leq \frac{2}{\epsilon}\left(d + \frac{1}{2}\ln\frac{1}{\delta}\right)
\end{aligned}$$

Proof ***Part 1:*** Follows as a simple corollary to the second part (plugging $d = 1$).

Part 2: Recall that T_{Scut} is given directly by δ-tail$_{\mathsf{Scov}}$ for the problem under consideration. For (d-π-initials, uniform, ϵ) we have that $\mathrm{P}_{X^\infty}\{T_{\mathsf{Scov}} > t\} < e^{-t\epsilon\left(1-\frac{d}{t\epsilon}\right)^2}$ for $t \geq d/\epsilon$ from Proposition 3.44 above. Therefore, we will show that

$$t = \frac{2}{\epsilon}\left(d + \frac{1}{2}\ln\frac{1}{\delta}\right) \qquad \text{(B.27)}$$

[5] Under the assumption that c has $x_c^i \in [i/d + \epsilon, (i+1)/d - \epsilon]$ for $i = 1, ..., d$. It suffices to consider this case since Scov stops faster for d-π-initial concepts with endpoints nearer the subdomain boundaries.

implies $e^{-t\epsilon\left(1-\frac{d}{t\epsilon}\right)^2} < \delta$ and hence $\mathrm{P}_{x^\infty}\{T_{\mathtt{Scov}} > t\} < \delta$. Let $\gamma = \ln(1/\delta)$. Note that it suffices to show that (B.27) implies $t\epsilon\left(1-\frac{d}{t\epsilon}\right)^2 \geq \gamma$. To this end, consider

$$\begin{aligned}
t\epsilon\left(1-\frac{d}{t\epsilon}\right)^2 &= (2d+\gamma)\left(1-\frac{1}{\frac{1}{d}(2d+\gamma)}\right)^2 && \text{since } t\epsilon = 2d+\gamma, \\
&= d\gamma\left(\frac{2}{\gamma}+\frac{1}{d}\right)\left(1-\frac{1}{\gamma\left(\frac{2}{\gamma}+\frac{1}{d}\right)}\right)^2 \\
&= d\gamma\alpha\left(1-\frac{1}{\gamma\alpha}\right)^2 && \text{letting } \alpha = 2/\gamma + 1/d.
\end{aligned}$$

We now show $(1-1/[\gamma\alpha])^2 \geq 1/(d\alpha)$, and the result will follow.

$$\begin{aligned}
1-\frac{2}{\gamma\alpha}+\frac{1}{\gamma^2\alpha^2} &= 1-\left(\frac{2}{\gamma\alpha}+\frac{1}{d\alpha}\right)+\frac{1}{\gamma^2\alpha^2}+\frac{1}{d\alpha} \\
&= 1-\frac{1}{\alpha}\left(\frac{2}{\gamma}+\frac{1}{d}\right)+\frac{1}{\gamma^2\alpha^2}+\frac{1}{d\alpha} \\
&= 1-\left(\frac{1}{\alpha}\right)\alpha+\frac{1}{\gamma^2\alpha^2}+\frac{1}{d\alpha} \\
&= \frac{1}{\gamma^2\alpha^2}+\frac{1}{d\alpha} > \frac{1}{d\alpha}. \blacksquare
\end{aligned}$$

Appendix C

Technical details: Chapter 4

C.1 Preliminaries

Recall that we are using the notation $\underline{Z}_t$ to stand for the first order statistic of t i.i.d. observations of a real random variable $Z : X \to I\!\!R$; *i.e.*, $\underline{Z}_t(\mathbf{x}^t) = \min_{x_i \in \mathbf{x}^t}\{Z(x_1), ..., Z(x_t)\}$.

Proposition 4.5 *For any bounded random variable* $Z : X \to I\!\!R^+$ *with* $0 < \mathrm{P}\{Z = 0\} < 1$, $\mathrm{E}\,\underline{Z}_t = e^{\Theta(-t)}$.

Proof Let $0 \le Z \le \bar{z}$. First, since $\mathrm{E}\,\underline{Z}_t \le \bar{z}\mathrm{P}\{Z > 0\}^t$, and $\mathrm{P}\{Z > 0\} < 1$ by hypothesis, we get $\mathrm{E}\,\underline{Z}_t = e^{O(-t)}$. Next, since $\mathrm{P}\{Z = 0\} < 1$ there must be some $z' > 0$ such that $\mathrm{P}\{Z \ge z'\} > 0$, and hence $\mathrm{E}\,\underline{Z}_t \ge z'\mathrm{P}\{Z \ge z'\}^t = e^{\Omega(-t)}$. ▌

Proposition 4.6 *For any random variable* $Z \sim \mathsf{uniform}(0, 1)$, $\mathrm{E}\,\underline{Z}_t = \Theta(t^{-1})$.

Proof It is not hard to show that for a bounded random variable Z,

$$\mathrm{E}\,\underline{Z}_t \;=\; \int_0^{\bar{z}} zt(1 - F_Z(z))^{t-1} f_Z(z)dz,$$

where $0 \le Z \le \bar{z}$, and f_Z and F_Z are Z's density and distribution functions respectively [Larsen and Marx, 1981, p.99]. In particular, if $Z \sim \mathsf{uniform}(0, \bar{z})$ then it has density and distribution functions $f_Z = \frac{1}{\bar{z}}$ and $F_Z = \frac{z}{\bar{z}}$ over $0 \le z \le \bar{z}$. A simple calculation then shows

$$\mathrm{E}\,\underline{Z}_t \;=\; \int_0^{\bar{z}} zt\left(1 - \frac{z}{\bar{z}}\right)^{t-1} \frac{1}{\bar{z}}dz \;=\; \frac{\bar{z}}{t+1}. \;▌$$

Proposition 4.7 (Finite UB) *For any finite concept class* C*: Any consistent hypothesizer* H *for* C *obtains an exponential learning curve (i.e.,* $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) = e^{O(-t)}$ *) for every target concept* $c \in C$*, regardless of the domain distribution* P.

Proof Fix an arbitrary target concept $c \in C$ and domain distribution P. There will be at most $N = |C| - 1$ non-zero difference sets $D_0 = \{c\triangle c_i : \mathrm{P}(c\triangle c_i) > 0\}$. Let p_0 be the minimum such probability. Then the probability that some difference set remains unobserved after t training examples is at most $N(1 - p_0)^t$. Notice that observing a domain object in each difference set implies that a consistent hypothesizer H for C will produce a hypothesis with zero error. Therefore, $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) \le N(1 - p_0)^t = e^{O(-t)}$. ▌

Proposition 4.8 (Universal LB) *For any non-trivial concept class C: There is a domain distribution* P *that forces any hypothesizer H to obtain an exponential learning curve (i.e.,* $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c', \mathbf{x}^t) = e^{\Omega'(-t)}$ *) for some target concept $c' \in C$.*

Proof Since C is non-trivial there must be two concepts c_1, $c_2 \in C$ such that $(c_1 \mathbin{\triangle} c_2) \neq \varnothing$ and $(c_1 \equiv c_2) \neq \varnothing$. Therefore we can fix a domain distribution P such that $d_{\mathrm{P}}(c_1, c_2) = p$ for some $0 < p < 1$ (*e.g.*, choose an $x_1 \in (c_1 \mathbin{\triangle} c_2)$ and $x_2 \in (c_1 \equiv c_2)$, and set $\mathrm{P}\{x_1\} = p$, $\mathrm{P}\{x_2\} = 1 - p$). For this distribution any hypothesizer H must obtain for a given sample size t

$$\begin{aligned}
&\operatorname*{avg}_{c_i \in \{c_1, c_2\}} \mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c_i, \mathbf{x}^t) \\
&\quad\geq \frac{1}{2}\, \mathrm{E}_{\mathbf{x}^t} \left[\, err(H, \mathrm{P}, c_1, \mathbf{x}^t) + err(H, \mathrm{P}, c_2, \mathbf{x}^t) \;\middle|\; c_1\mathbf{x}^t = c_2\mathbf{x}^t \,\right] \; \mathrm{P}_{X^t}\{c_1\mathbf{x}^t = c_2\mathbf{x}^t\} \\
&\quad= \frac{1}{2}\, \mathrm{E}_{\mathbf{x}^t} \left[\, err(H, \mathrm{P}, c_1, \mathbf{x}^t) + err(H, \mathrm{P}, c_2, \mathbf{x}^t) \;\middle|\; c_1\mathbf{x}^t = c_2\mathbf{x}^t \,\right] \quad (1-p)^t \\
&\quad\geq \frac{1}{2} \;\; p \;\; (1-p)^t \qquad = \; e^{\Omega(-t)}.
\end{aligned}$$

The last inequality holds since, for any $\mathbf{x}^t$ such that $c_1\mathbf{x}^t = c_2\mathbf{x}^t$, we get $H[c_1\mathbf{x}^t] = H[c_2\mathbf{x}^t] = h$ and hence $d_{\mathrm{P}}(h, c_1) + d_{\mathrm{P}}(h, c_2) \geq d_{\mathrm{P}}(c_1, c_2) = p$ by the triangle inequality. Finally, note that obtaining an average expected error of at least $e^{\Omega(-t)}$ for every t implies that H must obtain at least this expected error on one of c_1 or c_2 for infinitely many t. ▮

Proposition 4.11 (Chain UB) *For any concept chain C: Any consistent hypothesizer H for C obtains a rational learning curve (i.e.,* $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) = O(t^{-1})$ *) for every target concept $c \in C$, regardless of the domain distribution* P.

Proof Since any non-trivial chain obviously has VCdimension 1, the results of Haussler, Littlestone and Warmuth [1988] show that their special hypothesis strategy HLW obtains $\mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{HLW}, \mathrm{P}, c, \mathbf{x}^t) = O(t^{-1})$ in this case. Also, by [Haussler, Littlestone and Warmuth, 1988, Theorem 6.1] we know that *any* consistent hypothesizer H for C must obtain $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) = O((\ln t)/t)$. Here, we strengthen these results slightly by showing that any consistent hypothesizer H for C actually obtains $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) = O(t^{-1})$. Proving this also allows us to introduce some definitions and notation that will be needed later.

Definition C.1 (Uncertainty interval) *For a concept chain C, notice that any training sequence $c\mathbf{x}^t$ determines an* uncertainty interval *about the target concept c, denoted $[s[c\mathbf{x}^t], \ell[c\mathbf{x}^t]] = \{h \in C : s[c\mathbf{x}^t] \subseteq h \subseteq \ell[c\mathbf{x}^t]\}$, where $s[c\mathbf{x}^t]$ and $\ell[c\mathbf{x}^t]$ are the smallest and largest concepts consistent with $c\mathbf{x}^t$ respectively. Formally, we define the smallest concept consistent with a single training example cx by $s[cx] = \varnothing$ if $x \notin c$, and $s[cx] = \bigcap \{h \in C : h \subseteq c \text{ and } x \in h\}$ if $x \in c$; and the largest consistent concept by $\ell[cx] = X$ if $x \in c$, and $\ell[cx] = \bigcup \{h \in C : h \supseteq c \text{ and } x \notin h\}$ if $x \notin c$. Then, for a sequence $c\mathbf{x}^t$ we define $s[c\mathbf{x}^t] = \bigcup_{x \in \mathbf{x}^t} s[cx]$ and $\ell[c\mathbf{x}^t] = \bigcap_{x \in \mathbf{x}^t} \ell[cx]$. Finally, notice that any uncertainty interval $[s[c\mathbf{x}^t], \ell[c\mathbf{x}^t]]$ has a* width *under* P *given by $wid(C, \mathrm{P}, c, \mathbf{x}^t) = \mathrm{P}\,(\ell[c\mathbf{x}^t] - s[c\mathbf{x}^t])$.*

So for a concept chain C, we can think of the training examples as monotonically reducing the width of the uncertainty interval about an unknown target concept c. Clearly, any consistent hypothesizer H for C must guess a hypothesis from this interval, so the error of H's hypothesis must be bounded by the width of this interval; *i.e.*, $err(H, \mathrm{P}, c, \mathbf{x}^t) \leq wid(C, \mathrm{P}, c, \mathbf{x}^t)$. So it only remains to show that $\mathrm{E}_{\mathbf{x}^t}\, wid(C, \mathrm{P}, c, \mathbf{x}^t) = O(t^{-1})$. To this end, we observe that the worst case situation is represented by the *uniform chain*.

Definition C.2 (Uniform chain) *For a domain $X = [0, 1]$, let $I = \{\mathbf{i} = [0, i] : i \in [0, 1]\}$ be the class of initial segments of $[0, 1]$, and let* U *denote the* uniform *distribution over $[0, 1]$. Then the* uniform chain *is the concept space (I, U) formed from I and* U. *Note that the uniform chain (I, U) satisfies the identity $d_{\mathrm{U}}(\mathbf{i}, \mathbf{j}) = |i - j|$.*

Lemma C.3 below shows that for any $c \in (C, \mathrm{P})$ there is an $\mathbf{i} \in (I, \mathrm{U})$ such that

$$\mathrm{E}_{\mathbf{x}^t}\, wid(C, \mathrm{P}, c, \mathbf{x}^t) \;\leq\; \mathrm{E}_{\mathbf{x}^t}\, wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t). \tag{C.1}$$

So it suffices to determine how quickly the uncertainty interval width shrinks for a uniform chain. Here it turns out that we can determine an exact rate of decrease. In fact, Lemma C.4 below shows that for any target interval $\mathbf{i} = [0, i]$ in (I, U),

$$\mathrm{E}_{\mathbf{x}^t}\, wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \;=\; \frac{2 - (1-i)^{t+1} - i^{t+1}}{t+1}, \tag{C.2}$$

which is explicitly rational in t. Therefore, combining (C.1) and (C.2) shows that any consistent hypothesizer H for C must obtain $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, x^t) = O(t^{-1})$. ▮

Lemma C.3 *For any $c \in (C, \mathrm{P})$ there is an $\mathbf{i} \in (I, \mathrm{U})$ such that*

$$\mathrm{E}_{\mathbf{x}^t}\, wid(C, \mathrm{P}, c, \mathbf{x}^t) \;\leq\; \mathrm{E}_{\mathbf{x}^t}\, wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t).$$

Proof (There is a slight ambiguity here as the sequences $\mathbf{x}^t$ actually range over different domains X^t in each case, but this has no bearing on the result.) First, consider the space (C, P) and choose an arbitrary target $c \in (C, \mathrm{P})$. For this c, define the random variables $S^c(x) = \mathrm{P}(c - s[cx])$ and $L^c(x) = \mathrm{P}(\ell[cx] - c)$, which measure the distance between c and the smallest and largest concepts in the uncertainty interval $[s[cx], \ell[cx]]$ respectively. Then the variables $\underline{S}^c{}_t(\mathbf{x}^t) = \min\{S^c(x_1), ..., S^c(x_t)\}$ and $\underline{L}^c{}_t(\mathbf{x}^t) = \min\{L^c(x_1), ..., L^c(x_t)\}$ measure the distance between c and the smallest and largest concepts in $[s[c\mathbf{x}^t], \ell[c\mathbf{x}^t]]$. *I.e.*,

$$\begin{aligned} wid(C, \mathrm{P}, c, \mathbf{x}^t) &= \mathrm{P}(\ell[c\mathbf{x}^t] - s[c\mathbf{x}^t]) \\ &= \mathrm{P}(\ell[c\mathbf{x}^t] - c) + \mathrm{P}(c - s[c\mathbf{x}^t]) \\ &= \underline{S}^c{}_t(\mathbf{x}^t) + \underline{L}^c{}_t(\mathbf{x}^t). \end{aligned}$$

Now, turning our attention to the space (I, U), consider a corresponding target concept $\mathbf{i} = [0, \mathrm{P}(c)] \in I$. For this concept we can define $\underline{S}^{\mathbf{i}}{}_t$ and $\underline{L}^{\mathbf{i}}{}_t$ as above. Notice that, by construction, this concept $\mathbf{i}$ has the property that the width of its uncertainty interval can shrink no faster than c's:

P1 *For any $y \leq \mathrm{P}(c)$ we have $\mathrm{P}\{S^c < y\} \geq \mathrm{U}\{S^{\mathbf{i}} < y\}$; and for any $y \leq 1 - \mathrm{P}(c)$ we have $\mathrm{P}\{L^c < y\} \geq \mathrm{U}\{L^{\mathbf{i}} < y\}$.*

This obviously yields the result since for positive random variables X and Y, $F_X \geq F_Y$ implies $\mathrm{E}\,X \leq \mathrm{E}\,Y$. Finally, to prove P1, note that $\mathrm{U}\{S^{\mathbf{i}} < y\} = y$ for $y \leq \mathrm{P}(c)$ by construction, whereas $\mathrm{P}\{S^c < y\} \geq y$ over this range. Similarly, $\mathrm{U}\{L^{\mathbf{i}} < y\} = y$ for $y \leq 1 - \mathrm{P}(c)$, and yet $\mathrm{P}\{L^c < y\} \geq y$ over this range. (To see this for L^c, let $\ell_y = \bigcap\{h \in C : \mathrm{P}(h - c) \geq y\}$ and notice that $\mathrm{P}\{L^c < y\} = \mathrm{P}(\ell_y - c) \geq y$. A similar argument works for S^c.) ▮

Lemma C.4 *For any initial segment $\mathbf{i} = [0, i]$ in (I, U),*

$$\mathrm{E}_{\mathbf{x}^t}\, wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \;=\; \frac{2 - (1-i)^{t+1} - i^{t+1}}{t+1}.$$

Proof Consider an arbitrary target concept $\mathbf{i} = [0, i] \in I$. Let $S(x)$ and $L(x)$ be random variables for $\mathbf{i}$ as defined in the proof of Lemma C.3 above (dropping the superscript $\mathbf{i}$). Then by definition we have $wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) = \underline{S}_t(\mathbf{x}^t) + \underline{L}_t(\mathbf{x}^t)$, and hence

$$\begin{aligned} \mathrm{E}_{\mathbf{x}^t}\, wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) &= \mathrm{E}_{\mathbf{x}^t}\left[\, \underline{S}_t(\mathbf{x}^t) + \underline{L}_t(\mathbf{x}^t) \,\right] \\ &= \sum_{k=0}^{t} \binom{t}{k} \mathrm{U}(\mathbf{i})^k\, \mathrm{U}(\mathbf{i}^c)^{t-k} \Big(\; \mathrm{E}_{\mathbf{x}^k}\left[\underline{S}_k \mid x_1, ..., x_k \in \mathbf{i}\right] \\ &\qquad\qquad + \;\; \mathrm{E}_{\mathbf{x}^{t-k}}\left[\underline{L}_{t-k} \mid x_{k+1}, ..., x_t \notin \mathbf{i}\,\right]\Big). \end{aligned}$$

Obviously in this case $\mathrm{U}(\mathbf{i}) = i$ and $\mathrm{U}(\mathbf{i}^c) = 1 - i$. Now notice that for $S(x)$ and $L(x)$ defined as above, we have $S|(x \in \mathbf{i}) \sim \mathsf{uniform}[0, i)$ and $L|(x \in \mathbf{i}^c) \sim \mathsf{uniform}(0, 1-i)$. Therefore, Proposition 4.6 above shows that for any random variable $R \sim \mathsf{uniform}(0, r)$ we have $\mathrm{E}_{\mathbf{x}^t}\,[\underline{R}_t(\mathbf{x}^t)] = r/(t+1)$, and hence

$$\mathrm{E}_{\mathbf{x}^t}\, wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \;=\; \sum_{k=0}^{t} \binom{t}{k}\, i^k\, (1-i)^{t-k} \left(\frac{i}{k+1} + \frac{1-i}{t-k+1} \right).$$

Each of these two terms can be reduced via the Binomial Theorem [Brualdi, 1977, Chapter 4] to yield the stated result. For example, the first term yields

$$\begin{aligned}
&\sum_{k=0}^{t} \binom{t}{k} i^k (1-i)^{t-k} \frac{i}{k+1} \\
&= \frac{1}{t+1} \sum_{k=0}^{t} \binom{t+1}{k+1} i^{k+1} (1-i)^{(t+1)-(k+1)} \\
&= \frac{1}{t+1} \sum_{\ell=0}^{t} \binom{t+1}{\ell} i^{\ell} (1-i)^{t+1-\ell} \\
&= \frac{1}{t+1} \left[1 - (1-i)^{t+1} \right]. \;\blacksquare
\end{aligned}$$

C.2 Continuous chains

Theorem 4.12 (Continuous LB) *For any continuous concept chain C: There is a domain distribution* P *that forces any hypothesizer H to obtain a rational learning curve (i.e.,* $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c', \mathbf{x}^t) = \Omega'(t^{-1})$ *) for some target concept $c' \in C$.*

Proof Since C is continuous it can be indexed $C = \{c_y : y \in [0,1]\}$ where $c_y \subset c_z$ for $y < z$. Given this indexing, we can fix a domain distribution P such that $d_{\mathrm{P}}(c_y, c_z) = |y - z|$.[1] Notice that the resulting space (C, P) is isomorphic to the uniform chain (I, U) in Definition C.2. Therefore, it suffices to establish the lower bound for (I, U).

For this space we already know by Lemma C.4 that the width of any uncertainty interval only decreases rationally to zero; *i.e.*, $\mathrm{E}_{\mathbf{x}^t}\, wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) = \Omega(t^{-1})$ for any $\mathbf{i} = [0, i] \in I$. So it would be surprising if a hypothesizer could do significantly better than this for every target concept in I. To prove that any hypothesizer is forced to exhibit rational convergence for *some* fixed target concept $\mathbf{i} \in I$, we employ the same averaging argument used in Proposition 4.8. In particular, we fix a prior distribution Q on the collection of initial segment concepts I, and argue that any hypothesizer H must obtain a large expected error on average over the concepts $\mathbf{i} \in I$.

Consider the uniform prior Q on I. Here it turns out that the simple "midpoint" guessing strategy MP (Figure C.1) is *Bayes-optimal* for this prior: that is, Lemma C.5 below shows that *any* hypothesizer H must obtain

$$\mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \;\geq\; \mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{MP}, \mathrm{U}, \mathbf{i}, \mathbf{x}^t), \tag{C.3}$$

where target concepts $\mathbf{i}$ are chosen randomly according to Q. Therefore, it suffices to establish a lower bound on MP's average expected error. Here we see that, not surprisingly, MP achieves an average expected error that is a fixed fraction of the expected width of the uncertainty interval. In particular, Lemma C.6 below shows

$$\mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{MP}, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \;=\; \frac{1}{4}\, \mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t). \tag{C.4}$$

[1] This distribution can be constructed by the same procedure used to construct the Lebesgue measure on $[0,1]$; see *e.g.*, [Ash, 1972, Chapter 1].

Strategy MP $(\mathbf{ix}^t)$ for the uniform chain (I, U).

INPUT: a training sequence $\mathbf{ix}^t$ labelled by some unknown target interval $\mathbf{i} \in I$, which yields the uncertainty interval $[\mathbf{s}[\mathbf{ix}^t], \mathbf{l}[\mathbf{ix}^t]]$.

PROCEDURE:

- Guess the midpoint concept $\mathbf{m} = [0, m]$ defined by the endpoint $m = (s + \ell)/2$.

Figure C.1: Strategy MP

So, combining (C.2), (C.3), and (C.4), we see that any hypothesizer H must obtain

$$\mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \;\geq\; \frac{1}{4} \int_0^1 \frac{2 - (1-i)^{t+1} - i^{t+1}}{t+1}\, di$$

$$= \frac{1}{2(t+2)}. \tag{C.5}$$

Clearly, since this lower bound holds on average over all $\mathbf{i} \in I$, it must hold for *some* $\mathbf{i}_t \in I$ for each training sample size t. However, we need to establish the stronger claim that there is a single $\mathbf{i}$ in I that forces $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) = \Omega(t^{-1})$ for infinitely many training sample sizes t. To this end, Lemma C.7 below shows that for any $\alpha > 0$

$$\mathrm{Q}\left\{ \mathbf{i} \in I \;:\; \mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \geq \frac{1-\alpha}{2(t+2)} \;\; i.o.\;\, t \right\} > 0.$$

I.e., any hypothesizer will be forced to obtain an expected error above this bound infinitely often for a nontrivial portion of the concepts in I. This gives the result.[2] ▮

Lemma C.5 *For any hypothesizer* H, $\;\mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \geq \mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{MP}, \mathrm{U}, \mathbf{i}, \mathbf{x}^t)$.

Proof We will show that the hypotheses produced by MP are Bayes-optimal for the uniform prior Q on I and the uniform domain distribution U on $[0,1]$. The result then follows by a well known fact about Bayes-optimal prediction; *cf.* [Duda and Hart, 1973, Chapter 2].

Intuitively, this result is rather clear. Given a training sequence $\mathbf{z}^t = \langle\langle x_1, y_1\rangle, ..., \langle x_t, y_t\rangle\rangle$ yielding the uncertainty interval $[\mathbf{s}[\mathbf{z}^t], \mathbf{l}[\mathbf{z}^t])$, the posterior probability that a domain object $x \in [s[\mathbf{z}^t], \ell[\mathbf{z}^t])$ gets classified as $y = 1$ is just the proportion of initial segment concepts $\mathbf{i} \in [\mathbf{s}[\mathbf{z}^t], \mathbf{l}[\mathbf{z}^t])$ that contain x. Therefore, the Bayes-optimal classification for x is $y = 1$ just when $x \leq m = (s[\mathbf{z}^t] + \ell[\mathbf{z}^t])/2$.

To formally prove this, consider an arbitrary training sequence $\mathbf{z}^t = \langle\langle x_1, y_1\rangle, ..., \langle x_t, y_t\rangle\rangle$ that is consistent with some target segment $\mathbf{i} \in I$. We can compute the posterior probability that a particular domain object x gets classified as $y = 1$ as follows.

$$\begin{aligned}
\mathrm{P}(y \mid x, \mathbf{z}^t) &= \int_0^1 p(y, \mathbf{i} \mid x, \mathbf{z}^t)\, di \\
&= \int_0^1 \mathrm{P}(y \mid \mathbf{i}, x, \mathbf{z}^t)\, p(\mathbf{i} \mid x, \mathbf{z}^t)\, di \\
&= \int_0^1 \mathrm{P}(y \mid \mathbf{i}, x)\, p(\mathbf{i} \mid \mathbf{z}^t)\, di,
\end{aligned}$$

[2]Note that Haussler, Littlestone and Warmuth [Theorem 3.2, 1994] have independently established a result similar to (C.5) (in subsequent work to [Haussler, Littlestone and Warmuth, 1988]). However their argument is quite different and they do not supply the final step (Lemma C.7). The proof presented here generalizes more readily to Theorem 4.15 below.

since y is independent of $\mathbf{z}^t$ given x and $\mathbf{i}$, and $\mathbf{i}$ is independent of x. Now, notice that

$$p(\mathbf{i} \,|\, \mathbf{z}^t) \;=\; \begin{cases} 0 & \text{if} \quad i \notin [s[\mathbf{z}^t], \ell[\mathbf{z}^t]) \\ \dfrac{1}{\ell[\mathbf{z}^t] - s[\mathbf{z}^t]} & \text{if} \quad i \in [s[\mathbf{z}^t], \ell[\mathbf{z}^t]), \end{cases}$$

where $s[\mathbf{z}^t]$ is the largest positive example and $\ell[\mathbf{z}^t]$ is the smallest negative example in $\mathbf{z}^t$, and notice that $\mathrm{P}(y = 1 \,|\, \mathbf{i}, x) = 1$ if $x > i$ and $\mathrm{P}(y = 1 \,|\, \mathbf{i}, x) = 0$ if $x \le i$. Therefore, the posterior probability that an object x is classified as $y = 1$ given $\mathbf{z}^t$ is given by

$$\begin{aligned} \mathrm{P}(y = 1 \,|\, x, \mathbf{z}^t) \;&=\; \int_x^{\ell[\mathbf{z}^t]} \frac{1}{\ell[\mathbf{z}^t] - s[\mathbf{z}^t]}\, di \\ &=\; \frac{\ell[\mathbf{z}^t] - x}{\ell[\mathbf{z}^t] - s[\mathbf{z}^t]}. \end{aligned} \tag{C.6}$$

for $x \in [s[\mathbf{z}^t], \ell[\mathbf{z}^t])$. (Note that this posterior probability is 1 if $x < s[\mathbf{z}^t]$, and 0 if $x \ge \ell[\mathbf{z}^t]$.) Then, following the standard Bayes decision procedure, given $\mathbf{z}^t$, we classify x as 1 exactly when $\mathrm{P}(y = 1 \,|\, x, \mathbf{z}^t) \ge 1/2$. But by (C.6) this occurs when $x \le (s[\mathbf{z}^t] + \ell[\mathbf{z}^t])/2$, which is exactly what MP's hypothesis does. ▮

Lemma C.6 $\quad \mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{MP}, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \;=\; \frac{1}{4}\, \mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t).$

Proof First, notice that $\mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{MP}, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) = \mathrm{E}_{\mathbf{x}^t}\, \mathrm{E}_{\mathbf{i}}\, err(\mathsf{MP}, \mathrm{U}, \mathbf{i}, \mathbf{x}^t)$ by Fubini's Theorem. Now consider an arbitrary (ordered) object sequence $\mathbf{x}^t = \{x_1 < x_2 < \cdots < x_t\}$ that partitions the chain I into $t+1$ subintervals $I_{[0,x_1)}, I_{[x_1,x_2)}, \ldots, I_{[x_t,1]}$, where $I_{[x_n, x_{n+1})} = \{\mathbf{i} \in I : x_n \le i < x_{n+1}\}$. That is, each subinterval contains initial segment concepts that identically label the objects in $\mathbf{x}^t$. Consider an arbitrary subinterval $I_{[x_n,x_{n+1})} = [\mathbf{s}, \mathbf{l})$. For this subinterval, MP always guesses the same hypothesis, $\mathbf{m}$, defined by the endpoint $m = (s + \ell)/2$. Thus,

$$\begin{aligned} \int_s^\ell err(\mathsf{MP}, \mathrm{U}, \mathbf{i}, \mathbf{x}^t)\, di \;&=\; \int_s^\ell d_{\mathrm{U}}(\mathbf{m}, \mathbf{i})\, di \\ &=\; \int_s^\ell |m - i|\, di \\ &=\; 2\int_s^m m - i \;\; di \\ &=\; \frac{1}{4}\int_s^\ell \ell - s \;\; di \;=\; \frac{1}{4}\int_s^\ell wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t)\, di. \end{aligned}$$

The result then follows since

$$\begin{aligned} \mathrm{E}_{\mathbf{i}}\, err(\mathsf{MP}, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \;&=\; \sum_{n=0}^{t} \int_{x_n}^{x_{n+1}} err(\mathsf{MP}, \mathrm{U}, \mathbf{i}, \mathbf{x}^t)\, di \\ &=\; \frac{1}{4}\int_0^1 wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t)\, di \\ &=\; \frac{1}{4}\, \mathrm{E}_{\mathbf{i}}\, \mathrm{E}_{\mathbf{x}^t}\, wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t). \end{aligned}$$ ▮

Lemma C.7 *For any $\alpha > 0$,*

$$\mathrm{Q}\left\{\mathbf{i} \in I \;:\; \mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \geq \frac{1-\alpha}{2(t+2)} \;\; i.o. \;\; t\right\} > 0.$$

Proof We want to use the result that a large error is forced on average over all $\mathbf{i} \in I$ for each t to show that a large error must be forced for a significant proportion of the concepts $\mathbf{i} \in I$ for each t, and then show that this means a large error must be forced for some particular $\mathbf{i} \in I$ for infinitely many t.

Let $err(\mathbf{i}, t) = \mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{U}, \mathbf{i}, \mathbf{x}^t)$. Focusing on the distribution Q over I, we are first interested in the event

$$B_t \;\; \triangleq \;\; \left\{\mathbf{i} \in I : err(\mathbf{i}, t) \geq \frac{1-\alpha}{2(t+2)}\right\}$$

which contains the concepts $\mathbf{i} \in I$ that force a large error for training sample size t. To prove that this event has significant positive measure under Q for every t, fix an arbitrary $t > 0$ and think of $R = err(\mathbf{i}, t)$ as a random variable over $\mathbf{i}$. Then by (C.5) we know that $\mathrm{E}R \geq 1/(2(t+2))$, and by Lemma C.4 we know that if H is consistent for I then $R \leq 2/(t+1)$. Combining these two facts gives $0 \leq R \leq 8\mathrm{E}R$. Now, letting $q \triangleq \mathrm{Q}(B_t) = \mathrm{Q}\{\mathbf{i} : R(\mathbf{i}) \geq (1-\alpha)\mathrm{E}R\}$, we have $\mathrm{E}R \leq (1-q)(1-\alpha)\mathrm{E}R + 8q\mathrm{E}R$. Finally, notice that this holds if and only if $q = \mathrm{Q}(B_t) \geq \alpha/(\alpha+7)$.[3]

Now we consider the event

$$B \;\; \triangleq \;\; \bigcap_{n=1}^{\infty} \bigcup_{t=n}^{\infty} B_t$$

which contains the concepts $\mathbf{i} \in I$ that force H to exhibit $err(\mathbf{i}, t) \geq (1-\alpha)/(2(t+2))$ for infinitely many training sample sizes t. We wish to prove that $\mathrm{Q}(B) > 0$ and hence there exists some $\mathbf{i} \in B \subset I$ that forces $err(\mathbf{i}, t) \geq (1-\alpha)/(2(t+2))$ i.o. t. To do this, let $B^T = \bigcap_{n=1}^{T} \bigcup_{t=n}^{\infty} B_t$. Notice that $B^T \downarrow B$, and hence $\mathrm{Q}(B^T) \downarrow \mathrm{Q}(B)$ by [Ash, 1972, Theorem 1.2.7]. But now see that $B_T \subseteq B^T$ for all T, and therefore $\mathrm{Q}(B^T) \geq \mathrm{Q}(B_T) \geq \alpha/(\alpha+7)$ for all T. This implies $\mathrm{Q}(B) \geq \alpha/(\alpha+7)$, and we are done. ▮

C.3 Dense chains

Theorem 4.15 (Dense LB) *For any dense concept chain C: There is a domain distribution* P *that forces any hypothesizer H to obtain a rational learning curve* (*i.e.*, $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c', \mathbf{x}^t) = \Omega^*(t^{-1})$) *for some target concept $c' \in C$.*

Proof We establish the slightly weakened proposition that $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) = \Omega'(t^{-1-\epsilon})$ for any $\epsilon > 0$ (hence the notation Ω^*). The basic idea is to generalize the proof of Theorem 4.12 to handle arbitrary dense chains. Here we must face the fact that C need not be continuous in general (*e.g.*, C might only be countably infinite) so we cannot directly reduce the problem to a uniform chain, as before. Instead, we have to define a domain distribution P that *simulates* the structure of a uniform chain as closely as possible. To do this, we explicitly construct a dense subchain of C on a countable subdomain of X and then define the appropriate distributions P and Q.

Construction First, construct a dense chain C_0 on a countable domain X_0 by selecting concepts from C and domain objects from X in a series of Stages $k = 0, 1, 2, ...$ as follows: At Stage 0, select any two concepts $c_0 \subset c_1$ from C and choose a domain object x_1 between them (*i.e.*, choose $x_1 \in c_1 - c_0$ such that there remains $c_2, c_3 \in C$ with $c_0 \subset c_2 \subset c_3 \subset c_1$ and $x_1 \in c_3 - c_2$; see Figure C.2). Next, at Stage 1, choose a domain object x_2 and concept c_2 between c_0 and x_1, and then a concept c_3 and domain object x_3 between x_1 and c_1; maintaining the alternation between domain objects and target concepts shown in Figure C.2. Then

[3]Note that if H is not consistent for I then we can always construct a consistent hypothesizer H' such that $err(H', \mathrm{U}, \mathbf{i}, \mathbf{x}^t) \leq \min\left\{err(H, \mathrm{U}, \mathbf{i}, \mathbf{x}^t),\; wid(I, \mathrm{U}, \mathbf{i}, \mathbf{x}^t)\right\}$ for all $\mathbf{i} \in I$, $\mathbf{x}^t \in X^t$; so it suffices to consider a consistent hypothesizer.

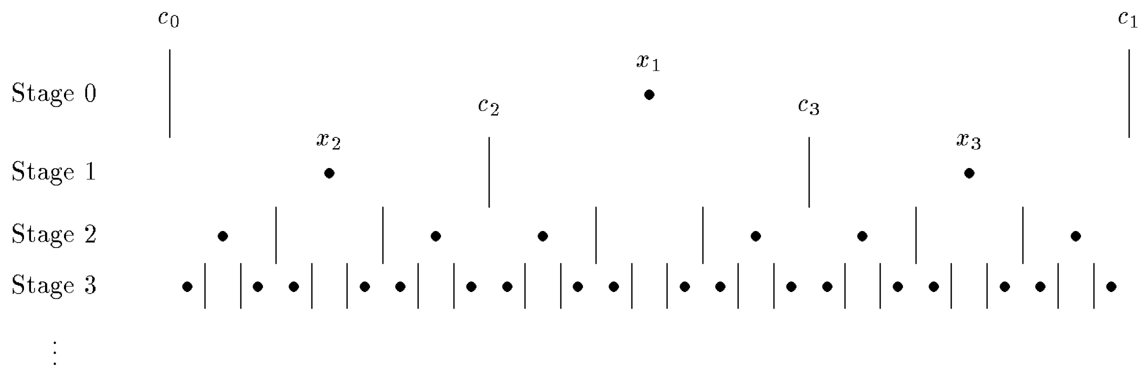

Figure C.2: Constructing a dense subchain on a countable subdomain: Each line indicates a concept that contains the domain objects to the left of the line (indicated by bullets). Repeating this construction for stages $k = 0, 1, 2, ...$ yields in a dense chain $C = \{c_0, c_1, ...\}$ defined on a countable domain $X = \{x_1, x_2, ...\}$.

for all subsequent Stages $k \geq 2$, choose a domain object and target concept in each "gap" left from previous stages, maintaining the alternation between target concepts and domain objects after each stage, again, as shown in Figure C.2. Notice that the density of C permits us to continue this process indefinitely, and therefore we obtain a dense subchain C_0 defined on a countable subdomain X_0. This construction provides us with a canonical structure on which to define our probability distributions. (We now drop the subscript 0 for the remainder of this proof, with the understanding that C and X now refer to the constructed C_0 and X_0 throughout.)

Now to define a domain distribution P on X: Notice that P cannot be uniform since X is only countably infinite. However, we can approximate a uniform distribution by assigning probabilities as follows. First, at Stage 0, assign $\mathrm{P}\{x_0\} = 0$ to the only domain object added at Stage 0. Then for Stages $k = 1, 2, ...$, assign a probability of $p_k = (3^\epsilon - 1)/(2 \cdot 3^{k(1+\epsilon)-1})$ to each of the domain objects x_i added at Stage k. This gives a well-defined probability distribution for any $\epsilon > 0$, since there are a total of $N_k = 2 \cdot 3^{k-1}$ objects added at each Stage k, and summing over stages $1, 2, ...$ yields a total probability of $(3^\epsilon - 1)\sum_{k=1}^{\infty} 3^{-\epsilon k} = 1$. Notice that we can use the parameter ϵ to control the "uniformity" of P, in that choosing smaller values of ϵ forces the probabilities p_k assigned at each stage to converge more slowly to 0, hence making the distribution more uniform.

Finally, to define the prior distribution Q on C we proceed in exactly the same way as for P on X above. That is, at Stage 0 assign $\mathrm{Q}\{c_0\} = \mathrm{Q}\{c_1\} = 0$, and then at each subsequent Stage $k \geq 1$ assign a probability $q_k = p_k$ to each of the concepts c_j added at Stage k. This yields a well defined probability distribution Q on C exactly as above, since during each Stage $k > 0$ an equal number of domain objects and target concepts are added to the construction.

Given this explicit construction of C, X, P, and Q, we can now repeat the lower bound argument from Theorem 4.12 as follows. First we must verify that P is indeed a "hard" domain distribution in the sense that it forces uncertainty intervals to shrink rationally as a function of t. Lemma C.8 below shows that for any $c \in C$

$$\mathrm{E}_{\mathbf{x}^t}\, wid(C, \mathrm{P}, c, \mathbf{x}^t) \quad > \quad \frac{\alpha_c}{t+1}, \tag{C.7}$$

for a constant $\alpha_c > 0$.

Given that this slow convergence holds for all target concepts in C it would be surprising if a hypothesizer could achieve significantly faster convergence for every possible target $c \in C$. To prove this, we follow the same averaging argument used in Proposition 4.8 and Theorem 4.12 above: Given the prior distribution Q over C, we argue that any hypothesizer H must achieve a large expected error *on average* over the target concepts $c \in C$. To do this, we consider a simple learning strategy MC (Figure C.3) that turns out to achieve near-optimal average expected error for the distributions P and Q defined above. Lemma C.9 below shows

---

Strategy MC $(c\mathbf{x}^t)$ for the constructed chain (C, P) with prior Q.

INPUT: a training sequence $c\mathbf{x}^t$ labelled by some unknown target concept $c \in C$, which yields the uncertainty interval $[s[c\mathbf{x}^t], \ell[c\mathbf{x}^t]]$.

PROCEDURE:

- Guess the concept $h \in [s, \ell]$ with maximum prior probability according to Q.

Figure C.3: Strategy MC

that any hypothesizer H must obtain

$$\mathrm{E}_c \, \mathrm{E}_{\mathbf{x}^t} \, err(H, \mathrm{P}, c, \mathbf{x}^t) \;\; \geq \;\; \beta \;\; \mathrm{E}_c \, \mathrm{E}_{\mathbf{x}^t} \, err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t), \tag{C.8}$$

for a fixed constant $\beta > 0$.

Therefore, it suffices to establish a rational lower bound on MC's average expected error. Not surprisingly, MC must obtain an average error that is at least a fixed fraction of the width of the uncertainty interval (discounting the fact that MC can guess the target concept exactly with a small probability). Lemma C.10 below shows that

$$\mathrm{E}_c \, \mathrm{E}_{\mathbf{x}^t} \, err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t) \;\; \geq \;\; \frac{\delta}{(t+2)^{2\epsilon}} \, \mathrm{E}_c \, \mathrm{E}_{\mathbf{x}^t} \, wid(C, \mathrm{P}, c, \mathbf{x}^t), \tag{C.9}$$

for a fixed constant $\delta > 0$.

Finally, combining (C.8), (C.9) and (C.7), we see that any hypothesizer H must obtain

$$\mathrm{E}_c \, \mathrm{E}_{\mathbf{x}^t} \, err(H, \mathrm{P}, c, \mathbf{x}^t) \;\; \geq \;\; \frac{\gamma}{(t+2)^{1+2\epsilon}},$$

for a constant $\gamma = \bar{\alpha}\beta\delta > 0$ (where $\bar{\alpha} > 0$ is the average value of α_c over $c \in C$). Clearly, since this bound holds on average over all concepts in C, it must hold for some $c_t \in C$ for each training sample size t. However, as in Theorem 4.12, we need to establish the stronger claim that there is a *single* concept in C that forces $\mathrm{E}_{\mathbf{x}^t} \, err(H, \mathrm{P}, c, \mathbf{x}^t) = \Omega(t^{-1-\epsilon})$ for infinitely many training sample sizes t. To this end, Lemma C.11 below establishes that for any $\lambda > 0$

$$\mathrm{Q}\left\{ c \in C \; : \; \mathrm{E}_{\mathbf{x}^t} \, err(H, \mathrm{P}, c, \mathbf{x}^t) \geq \frac{(1-\lambda)\gamma}{(t+2)^{1+2\epsilon}} \;\; i.o. \;\; t \right\} > 0.$$

That is, any hypothesizer must exhibit (near) rational convergence for a non-trivial portion of the concepts in C, as measured by Q. Finally, notice that we can freely choose ϵ to be arbitrarily close to zero, so the theorem follows. ▮

Lemma C.8 *For any* $c \in C$, $\;\; \mathrm{E}_{\mathbf{x}^t} \, wid(C, \mathrm{P}, c, \mathbf{x}^t) > \alpha_c/(t+1)$ *for a constant* $\alpha_c > 0$.

Proof The key reason we obtain rational convergence here is that, by construction, the region around any target concept is sufficiently dense to ensure (C, P) behaves like a uniform chain. To show this, consider an arbitrary concept $c \in C$ and let $[c - D, c + D]$ denote the subinterval of X containing all objects within a d_{P}-distance D of c. Also, let $r = 3^{1+\epsilon}$. Then we have

P2 $$\mathrm{E}_{\mathbf{x}^n} \left[wid(C, \mathrm{P}, c, \mathbf{x}^n) \,\middle|\, x_1, ..., x_n \in [c - D, c + D] \right] \; > \; \frac{D}{r(n+1)}.$$

Given P2, it is easy to prove the lemma by a simple application of the Binomial Theorem

$$\begin{aligned}
&\mathrm{E}_{\mathbf{x}^t}\, wid(C, \mathrm{P}, c, \mathbf{x}^t) \\
&> \sum_{n=0}^{t} \binom{t}{n} (2D)^n (1-2D)^{t-n} \ \ \mathrm{E}_{\mathbf{x}^n}\left[wid(C, \mathrm{P}, c, \mathbf{x}^n) \middle|\, x_1, ..., x_n \in [c-D, c+D] \right] \\
&> \sum_{n=0}^{t} \binom{t}{n} (2D)^n (1-2D)^{t-n} \ \frac{D}{r(n+1)} \\
&\geq \frac{D}{r(t+1)}.
\end{aligned}$$

To prove P2: Consider an arbitrary concept c added at some stage K of the construction. Notice that by the definition of P, c must have neighboring concepts (on both sides) at each distance $D_n = (3^\epsilon - 1)(3/2)\sum_{k=n+1}^{\infty} r^{-k} = \lambda \cdot r^{-n}$ for every $n \geq K$ (where λ is just a fixed positive constant). Let $D = D_K = \lambda \cdot r^{-K}$. This means that inside a local neighborhood $[c-D, c+D]$ of c, the chain (C, P) must behave as if it were a compressed version of the uniform chain (I, U). To see this, consider the right hand neighborhood $[c, c+D]$ and notice that the distribution of distances from c to its right-hand neighbors is bounded by r times a $\mathsf{uniform}(0,1)$ distribution (Figure C.4). In particular, for $d \leq D$ we have $\mathrm{P}\{L^c[x] < d\} \leq r\ \mathrm{P}\{R < d\}$ for $R \sim \mathsf{uniform}(0,1)$. That is, given the event $x \in [c, c+D]$, the distribution for $L^c[x] | x \in [c, c+D]$ is upper bounded by a $\mathsf{uniform}(0, D/r)$ distribution, as shown in Figure C.4.

Now, recalling that for positive random variables X and Y, $F_X \leq F_Y$ implies $\mathrm{E}\,X \geq \mathrm{E}\,Y$, we can simply apply the Binomial Theorem to obtain

$$\begin{aligned}
&\mathrm{E}_{\mathbf{x}^n}\left[wid(C, \mathrm{P}, c, \mathbf{x}^n) \middle|\, x_1, ..., x_n \in [c-D, c+D] \right] \\
&= \sum_{i=0}^{n} \binom{n}{i} \left(\frac{1}{2}\right)^n \Big(\mathrm{E}_{\mathbf{x}^i}\left[\underline{S}^c{}_i \,\middle|\, x_1, ..., x_i \in [c-D, c] \right] \\
&\qquad\qquad + \ \mathrm{E}_{\mathbf{x}^{n-i}}\left[\underline{L}^c{}_{n-i} \,\middle|\, x_{i+1}, ..., x_n \in [c, c+D] \right] \Big) \\
&\geq \sum_{i=0}^{n} \binom{n}{i} \left(\frac{1}{2}\right)^n \left[\frac{D}{r(i+1)} + \frac{D}{r(n-i+1)} \right] \\
&> \frac{D}{r(n+1)}. \ \blacksquare
\end{aligned}$$

Lemma C.9 *For any hypothesizer H,*

$$\mathrm{E}_c\,\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) \ \geq \ \beta\ \mathrm{E}_c\,\mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t),$$

for a fixed constant $\beta > 0$.

Proof The main reason MC obtains near-optimal average expected error is that the max-weight concept in any uncertainty interval, I, possesses a minimum fraction of I's total weight under Q. Let $q^* = \max_{c \in I} \mathrm{Q}(c)$ denote the maximum prior probability of any concept in I.

P3 *For any uncertainty interval I,* $\quad q^* \ \geq\ \rho\ \mathrm{Q}(I) \quad$ *for a fixed constant $\rho = (1-3^{-\epsilon})/2$.*

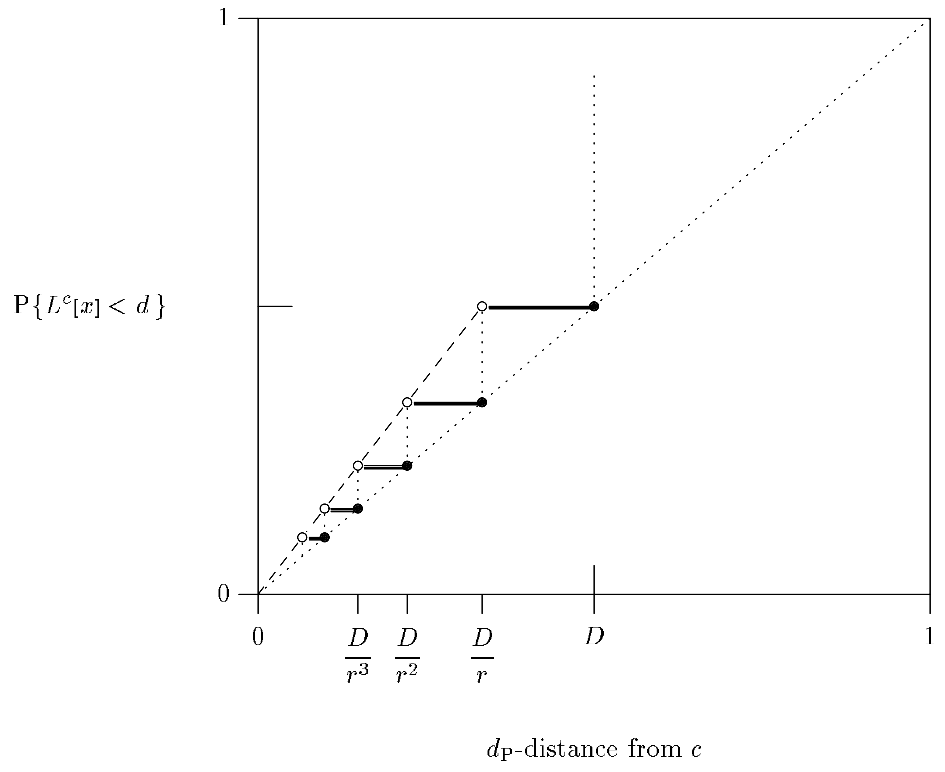

Figure C.4: The solid lines indicate the distribution of d_P-distances from a target concept c to its right-side neighbors in a dense chain C. Here $D = \lambda \cdot r^{-K}$, where $K = \mathit{stage}(c)$, $r = 3^{1+\epsilon}$, and λ is a fixed positive constant. The dashed line shows how the distribution of d_P-distances, given that the distance is less than D, is bounded by a $\mathsf{uniform}(0, D/r)$ distribution.

Given P3, we prove the lemma as follows: First note that we can rearrange the order of summation to obtain $\mathrm{E}_c \, \mathrm{E}_{\mathbf{x}^t} \, err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t) = \mathrm{E}_{\mathbf{x}^t} \, \mathrm{E}_c \, err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t)$, so it suffices to consider an arbitrary fixed $\mathbf{x}^t$. Notice that a sequence $\mathbf{x}^t = \{x_1 < x_2 < \cdots < x_t\}$ partitions the chain C into $t+1$ subintervals $C_{(\leftarrow, x_1)}$, $C_{(x_1, x_2)}$, ..., $C_{(x_t, \rightarrow)}$, where $C_{(x_n, x_{n+1})} = \{c \in C : x_n \in c \textit{ and } x_{n+1} \notin c\}$. *I.e.*, the concepts in each subinterval identically label $\mathbf{x}^t$.

Now, consider an arbitrary subinterval $I = C_{(x_n, x_{n+1})}$ and compare the performance of MC to an arbitrary hypothesizer H in this subinterval. Since $\mathbf{x}^t$ is fixed, we can think of the target concepts $c \in I$ as being drawn randomly according to the distribution Q. Notice that, since $c_1 \mathbf{x}^t = c_2 \mathbf{x}^t$ for any $c_1, c_2 \in I$, *any* hypothesizer must produce the same hypothesis for all target concepts $c \in I$. Thus, given targets $c \in I$, H produces a fixed hypothesis $H[c\mathbf{x}^t] = h$, and MC guesses the concept $c^* \in I$ with maximum prior probability according to Q.[4] We now show that any hypothesis h must obtain an average error over random concepts drawn from I that is at least a fixed fraction of c^*'s average error. Let $q^* = \mathrm{Q}(c^*) = \max_{c \in I} \mathrm{Q}(c)$, $Q = \sum_{c \in I} \mathrm{Q}(c)$, $Q^* = Q - q^*$, and $d^* = d_{\mathrm{P}}(h, c^*)$. Then, by the triangle inequality, we get

$$\begin{aligned} D(h) &\triangleq \sum_{c \in I} d_{\mathrm{P}}(h, c) \, \mathrm{Q}(c) \\ &\geq d_{\mathrm{P}}(h, c^*) q^* \;+ \sum_{c \in I - \{c^*\}} \theta \left[d_{\mathrm{P}}(c, c^*) - d_{\mathrm{P}}(h, c^*) \right] \, \mathrm{Q}(c) \\ &\geq d^* q^* + \theta \left[D(c^*) - d^* Q^* \right], \end{aligned} \tag{C.10}$$

where θ is a threshold function such that $\theta(x) = x$ if $x > 0$ and $\theta(x) = 0$ otherwise.

Now we just minimize this lower bound as a function of d^*. Here we have two cases: If $q^* > Q^*$ then (C.10) is minimized by choosing $d^* = 0$, which gives $D(h) \geq D(c^*)$. If, on the other hand, $q^* \leq Q^*$ then (C.10) is minimized by choosing $d^* = D(c^*)/Q^*$, which gives $D(h) \geq D(c^*) \, q^*/Q^*$. In this case we can just apply P3 to obtain $D(h) \geq D(c^*) \rho/(1-\rho)$. Thus, in either case we obtain $D(h) \geq \beta D(c^*)$ for a fixed constant $\beta \geq \rho/(1-\rho) > 0$.

Finally, we note that this shows any hypothesizer H must obtain

$$\begin{aligned} \mathrm{E}_c \, err(H, \mathrm{P}, c, \mathbf{x}^t) &= \sum_{n=0}^{t} \sum_{c \in C_{(x_n, x_{n+1})}} d_{\mathrm{P}}(H[c\mathbf{x}^t], c) \;\; \mathrm{Q}\{c\} \\ &\geq \sum_{n=0}^{t} \beta \sum_{c \in C_{(x_n, x_{n+1})}} d_{\mathrm{P}}(\mathsf{MC}[c\mathbf{x}^t], c) \;\; \mathrm{Q}\{c\} \\ &= \beta \, \mathrm{E}_c \, err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t). \end{aligned}$$

Proof of P3: Note that any subinterval I of C contains at least one, and at most two concepts of maximum weight under Q. (This is true since, by construction, any interval that contains three concepts from a stage K, must contain at least one concept from an earlier stage $N < K$; *cf.* Figure C.2.) Therefore, if c^* is a max-weight concept of I, then I can have: at most one other concept of maximum weight $q^* = \mathrm{Q}(c^*)$; at most 6 concepts of weight q^*/r, $r = 3^{1+\epsilon}$; at most $2 \cdot 3^k$ concepts of weight q^*/r^k for all stages $k \geq 0$; *etc.* (*I.e.*, this is symmetric to considering $I = C - \{c_0, c_1\}$ in Figure C.2 and choosing $c^* = c_2$.) This gives a total weight of $\mathrm{Q}(I) \leq q^* \sum_{k=0}^{\infty} 2 \cdot 3^k r^{-k} = 2q^*/(1 - 3^{-\epsilon})$. ∎

Lemma C.10 *For a constant $\delta > 0$,*

$$\mathrm{E}_c \, \mathrm{E}_{\mathbf{x}^t} \, err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t) \;\geq\; \frac{\delta}{(t+2)^{2\epsilon}} \, \mathrm{E}_c \, \mathrm{E}_{\mathbf{x}^t} \, wid(C, \mathrm{P}, c, \mathbf{x}^t),$$

[4] The proof of P3 below notes that there can be (at most) two concepts with maximum prior probability in an uncertainty interval I, so we just assume MC deterministically picks one of them.

Proof As in Lemma C.9 we rearrange the summation to obtain the identity $\mathrm{E}_c\,\mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{MC},\mathrm{P},c,\mathbf{x}^t) = \mathrm{E}_{\mathbf{x}^t}\,\mathrm{E}_c\, err(\mathsf{MC},\mathrm{P},c,\mathbf{x}^t)$. So consider an arbitrary fixed $\mathbf{x}^t = \{x_1 < x_2 < \cdots < x_t\}$, think of the target concept c as being randomly drawn according to Q, and consider MC's performance for $\mathbf{x}^t$. First note that there is a nonzero probability that MC guesses the target concept exactly. So we need to argue that *(a)* MC does not guess the target with too high a probability; and *(b)* given that MC does not guess the target, it must achieve an average error that is at least a fixed fraction of the uncertainty interval width. Let $\neg\mathsf{MC}[\mathbf{x}^t] \triangleq \{c : \mathsf{MC}[c\mathbf{x}^t] \neq c\}$ denote the set of concepts in C that MC does *not* guess given any possible labelling of the object sequence $\mathbf{x}^t$. Then for any fixed $\mathbf{x}^t$ we have

P4 $\quad \mathrm{Q}(\neg\mathsf{MC}[\mathbf{x}^t]) \ \geq\ \dfrac{1}{3^\epsilon(t+2)^\epsilon}.$

P5
$$\begin{aligned}&\mathrm{E}_c\left[\, err(\mathsf{MC},\mathrm{P},c,\mathbf{x}^t)\,\middle|\, \neg\mathsf{MC}[\mathbf{x}^t]\right]\\ &\quad\geq\ \frac{1}{8r^3}\ \mathrm{E}_c\left[\, wid(C,\mathrm{P},c,\mathbf{x}^t)\,\middle|\,\neg\mathsf{MC}[\mathbf{x}^t]\right]\ \mathrm{Q}(\neg\mathsf{MC}[\mathbf{x}^t]) \qquad \text{for } r = 3^{1+\epsilon}.\end{aligned}$$

Given these two facts, it is easy to prove the lemma as follows. Applying P4 and P5 yields

$$\begin{aligned}\mathrm{E}_c\, err(\mathsf{MC},\mathrm{P},c,\mathbf{x}^t) &= \mathrm{E}_c\left[\, err(\mathsf{MC},\mathrm{P},c,\mathbf{x}^t)\,\middle|\,\neg\mathsf{MC}[\mathbf{x}^t]\right]\quad \mathrm{Q}(\neg\mathsf{MC}[\mathbf{x}^t])\\ &\geq \mathrm{E}_c\left[\, wid(C,\mathrm{P},c,\mathbf{x}^t)\,\middle|\,\neg\mathsf{MC}[\mathbf{x}^t]\right]\quad \delta(t+2)^{-2\epsilon},\end{aligned}$$

for a constant $\delta = (8r^3 3^{2\epsilon})^{-1} > 0$. Now, averaging over $\mathbf{x}^t$ and re-arranging the sum yields

$$\begin{aligned}\mathrm{E}_{\mathbf{x}^t}\mathrm{E}_c err(\mathsf{MC},\mathrm{P},c,\mathbf{x}^t) &\geq \delta(t+2)^{-2\epsilon}\ \mathrm{E}_{\mathbf{x}^t}\mathrm{E}_c\left[\, wid(C,\mathrm{P},c,\mathbf{x}^t)\,\middle|\,\neg\mathsf{MC}[\mathbf{x}^t]\right]\\ &= \delta(t+2)^{-2\epsilon}\ \mathrm{E}_c\mathrm{E}_{\mathbf{x}^t}\left[\, wid(C,\mathrm{P},c,\mathbf{x}^t)\,\middle|\,\neg\mathsf{MC}[c]\right],\end{aligned}$$

where $\neg\mathsf{MC}[c] \triangleq \{\mathbf{x}^t : \mathsf{MC}[c\mathbf{x}^t] \neq c\}$ is the set of object sequences $\mathbf{x}^t \in X^t$ where MC does not guess c. This proves the lemma, since for any c we have

$$\mathrm{E}_{\mathbf{x}^t}\left[\, wid(C,\mathrm{P},c,\mathbf{x}^t)\,\middle|\,\neg\mathsf{MC}[c]\right]\ \geq\ \mathrm{E}_{\mathbf{x}^t}\, wid(C,\mathrm{P},c,\mathbf{x}^t).$$

(Intuitively, this follows because the uncertainty intervals where c is the max-weight concept tend to be small. That is, consider a fixed left boundary of an uncertainty interval around c and notice that every right boundary that gives $c \in \neg\mathsf{MC}[c]$ is strictly further away from c than any boundary where $\mathsf{MC}[c\mathbf{x}^t] = c$.)

Proof of P4: We get this bound because $\mathbf{x}^t$ partitions C into at most $t+1$ subintervals and MC can guess at most one concept per subinterval. Thus, the probability that MC guesses a random target concept $c \in C$ is bounded by the sum of the largest $t+1$ probabilities in C. That is, $\mathrm{Q}(\neg\mathsf{MC}[\mathbf{x}^t]) \geq \sum_{i=t+2}^{\infty} q_i$ where $\{q_i\}_{i=2}^{\infty}$ is the sequence of probabilities assigned in the construction of Q. So, letting $Q_T = \sum_{i=T}^{\infty} q_i$, we seek a lower bound on Q_T for $T = t+2$.

To determine this lower bound, notice that by the construction of C, the total number of concepts added in Stages 1 through K inclusive is $\sum_{k=1}^{K} 2\cdot 3^{k-1} = 3^K - 1$, so the index of the last concept added at Stage K is 3^K. This means that $Q_T \geq \sum_{k=\lceil 1+\log_3 T\rceil}^{\infty} Q_k$, where $Q_k \triangleq (3^\epsilon - 1)3^{-\epsilon k}$ is the total probability assigned at Stage k of the construction. Thus,

$$\begin{aligned}Q_T &\geq (3^\epsilon - 1)\sum_{k=\lceil 1+\log_3 T\rceil}^{\infty} 3^{-\epsilon k}\\ &= (3^\epsilon - 1)\frac{(3^{-\epsilon})^{\lceil 1+\log_3 T\rceil}}{1-3^{-\epsilon}}\\ &= 3^\epsilon(3^{-\epsilon})^{\lceil 1+\log_3 T\rceil}\\ &\geq 3^\epsilon(3^{-\epsilon})^{2+\log_3 T}\ =\ 3^{-\epsilon}T^{-\epsilon}.\end{aligned}$$

Proof of P5: This inequality holds because the region around the max-weight concept in any uncertainty interval is sufficiently dense in both P and Q to simulate the effects of a uniform prior on a uniform chain (as in Lemma C.6). Here we are interested in the conditional distribution of Q given $\neg\mathsf{MC}[\mathbf{x}^t]$, which is defined by

$$\mathrm{Q}^*(c) \;=\; \begin{cases} \mathrm{Q}(c)/Q^* & \text{if } c \in \neg\mathsf{MC}[\mathbf{x}^t], \\ 0 & \text{otherwise,} \end{cases}$$

where Q^* is the normalizing constant given by $Q^* = \mathrm{Q}(\neg\mathsf{MC}[\mathbf{x}^t])$. Let E_c^* denote expectation over c with respect to this conditional distribution. We seek a lower bound on

$$\mathrm{E}_c^*\, err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t) \;=\; \sum_{n=0}^{t} \int_{c\in(x_n,x_{n+1})} err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t)\;\, d\mathrm{Q}^*(c), \tag{C.11}$$

where (x_n, x_{n+1}) denotes the subinterval of concepts $c \in C$ between x_n and x_{n+1}.

To establish this lower bound consider an arbitrary subinterval $I = (x_n, x_{n+1})$ and let $c^* = \mathrm{argmax}_{c\in I}\mathrm{Q}(c)$. Notice that we can split the summation over I into two halves

$$\begin{aligned} &\int_{c\in I} err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t)\;\, d\mathrm{Q}^*(c) \\ &\quad= \int_{c\in(x_n,c^*)} d_{\mathrm{P}}(c^*, c)\;\, d\mathrm{Q}^*(c) \quad+\quad \int_{c\in(c^*,x_{n+1})} d_{\mathrm{P}}(c^*, c)\;\, d\mathrm{Q}^*(c). \end{aligned} \tag{C.12}$$

So consider one of the half intervals $J = (c^*, x_{n+1})$. It is not hard to show that the average d_{P}-distance from c^* to $c \in J$ is at least a fixed fraction of J's width under P:[5] To see this, note that by construction, for each $c \in (c^*, x_{n+1})$ added at Stage K of the construction we can assign a distinct x from Stage $K+1$ between c^* and c (namely, the x at Stage $K+1$ closest to c in (c^*, c); see Figure C.2). Then, for any subinterval (c^*, c) we get $\mathrm{P}(c^*, c) \geq \mathrm{Q}(c^*, c)/r = \mathrm{Q}^*(c^*, c)\; Q^*/r$, where $r = 3^{1+\epsilon}$. This means that for any d_{P}-distance d such that $d \leq \mathrm{P}(J)$ we get $\mathrm{Q}^*\{c \in J : d_{\mathrm{P}}(c^*, c) \leq d\} < rd/Q^*$. So, thinking of $d_{\mathrm{P}}(c^*, c)$ as a random variable over c, we can see that the distribution function for $d_{\mathrm{P}}(c^*, c)\,|\, c \in J$ is bounded by a $\mathsf{uniform}\,(\,0, \mathrm{P}(J)\, Q^*/r\,)$ distribution, as shown in Figure C.5. Thus we get $\mathrm{E}_c^*\,[\,d_{\mathrm{P}}(c^*, c) \mid J\,] \geq \mathrm{P}(J)\; Q^*/(2r)$, and hence

$$\begin{aligned} \int_{c\in J} d_{\mathrm{P}}(c^*, c)\;\, d\mathrm{Q}^*(c) \;&=\; \mathrm{E}_c^*\,[\,d_{\mathrm{P}}(c^*, c) \mid J\,]\;\; \mathrm{Q}^*(J) \\ &\geq\; \frac{Q^*}{2r}\, \mathrm{P}(J)\;\; \mathrm{Q}^*(J). \end{aligned} \tag{C.13}$$

Now, re-considering the complete interval I, note that one of the half intervals (x_n, c^*) or (c^*, x_{n+1}) must be at least half the d_{P}-width of I. Without loss of generality, assume $\mathrm{P}(J) \geq \mathrm{P}(I)/2$. Then we can argue that $\mathrm{Q}^*(J) \geq \mathrm{P}(J)/r \geq \mathrm{P}(I)/(2r) \geq \mathrm{Q}^*(I)/(2r^2)$ as above. Combining this with (C.12) and (C.13) gives

$$\begin{aligned} \int_{c\in I} err(\mathsf{MC}, \mathrm{P}, c, \mathbf{x}^t)\;\, d\mathrm{Q}^*(c) \;&=\; \int_{c\in I} d_{\mathrm{P}}(c^*, c)\;\; \mathrm{Q}^*(c) \\ &\geq\; \frac{Q^*}{8r^3}\, \mathrm{P}(I)\;\, \mathrm{Q}^*(I) \\ &=\; \frac{Q^*}{8r^3} \int_{c\in I} wid(I, \mathrm{P})\;\; d\mathrm{Q}^*(c). \end{aligned}$$

Substituting this back into (C.11) yields the stated bound. ▮

[5] Note that we use the notation (c^*, x_{n+1}) to refer ambiguously to both the set of concepts and the set of domain objects between c^* and x_{n+1}. The intended meaning should be clear from context.

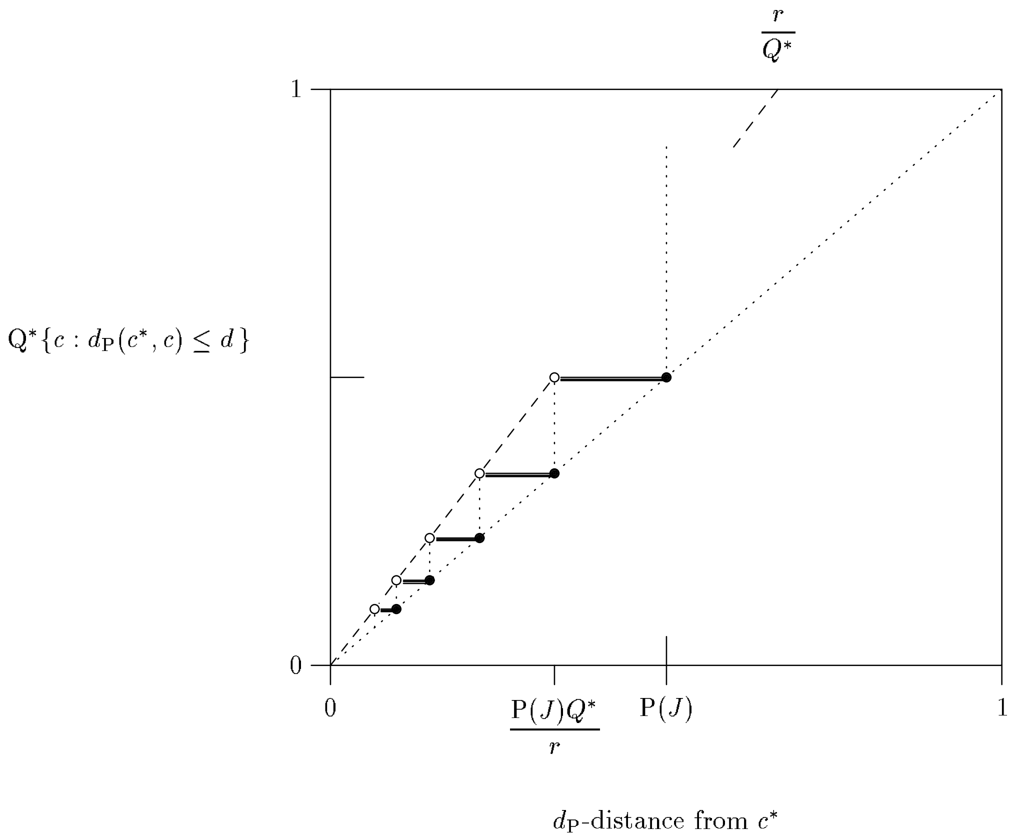

Figure C.5: The solid lines indicate the Q^*-distribution of d_P-distances from a max-weight target concept c^* to its right-side neighbors in a dense chain C. The dashed line shows how the Q^*-distribution of d_P-distances, given that the distance is less than $P(J)$, is bounded by a $\mathsf{uniform}(0, P(J)Q^*/r)$ distribution.

Lemma C.11 *For any $\lambda > 0$,*

$$\mathrm{Q}\left\{ c \in C \ : \ \mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c, \mathbf{x}^t) \geq \frac{(1-\lambda)\gamma}{(t+2)^{1+2\epsilon}} \ \ i.o. \ \ t \right\} > 0.$$

Proof Follows from basically the same argument as Lemma C.7. ∎

C.4 Scattered chains

Lemma 4.17 (Corollary to Hausdorff's Theorem) *For a scattered concept chain C, there is some least ordinal γ such that* (i) *every concept in C has order $\beta \leq \gamma$, and* (ii) *all limit concepts of a particular order β are* isolated *in concepts of the same or higher order.*

Proof First we need to formalize the notion of the *order* of a limit concept.

Definition C.12 (Limits and order) *For a chain C, let $i(C)$ denote the set of* isolated *concepts in C; i.e., concepts with a least-larger and greatest-smaller neighbor in C. Also let $C^0 = i(C)$. Then we define $C^{1+} = C - C^0$ to be the* limit *concepts of C. The limit concepts with order exactly* 1 *are given by $C^1 = i(C^{1+})$. Continuing in this way for arbitrary ordinals α, we define the concepts with order* at least α *by $C^{\alpha+} = C - \bigcup_{\beta<\alpha} C^\beta$, and the concepts with order* exactly α *by $C^\alpha = i(C^{\alpha+})$. Notice that for any ordinal α we have $C = C^{\alpha+} \cup \bigcup_{\beta<\alpha} C^\beta$.*

The key issue is to show that collecting successively higher order limit concepts in this way eventually exhausts a scattered chain.

P6 *For any scattered chain C, there exists some ordinal γ such that $C = \bigcup_{\beta\leq\gamma} C^\beta$.*

Notice that P6 implies both properties (i) and (ii) above, by the definition of C^β. To prove P6 we must resort to an inductive characterization of scattered linear orderings first developed by Hausdorff (an excellent treatment of this subject is given in Rosenstein's monograph [Rosenstein, 1982]). This characterization is based on constructing the following "condensation" map: We say that two concepts are a finite distance apart if there are only finitely many concepts between them in the ordering. Then the *finite condensation* map $f : C \to 2^C$ is defined by $f(c) = \{d \in C : c \textit{ and } d \textit{ are a finite distance apart}\}$. The effect of this map is to collapse the chain into a collection of subintervals (that is, subintervals of the chain C, not the domain X). The key point is to notice that these subintervals *themselves* form a linear-ordering, so we can naturally define iterates of this map as follows. For a *successor* ordinal $\beta + 1$, define $f^{\beta+1}(c) = \bigcup \{ f^\beta(d) : f^\beta(d) \textit{ and } f^\beta(c) \textit{ are a finite distance apart}\}$; and for any *limit* ordinal λ, define $f^\lambda(c) = \bigcup_{\beta<\lambda} \{ f^\beta(c) \}$. Then we have the following relations.

P7 *$f^\beta(C) = f^\alpha(C)$ for all $\beta \geq \alpha$ if and only if $f^\alpha(C)$ is dense or a singleton.*

(This proposition is more or less immediate from the definitions; see *e.g.*, [Rosenstein, 1982, p.81].) Now, define γ to be the *least* ordinal for which $f^\beta(C) = f^\gamma(C)$ for all $\beta \geq \gamma$. (We know that such a γ must exist, since for any chain C with cardinality κ there is an ordinal $\alpha < \kappa+$ such that $f^\beta(C) = f^\alpha(C)$ for all $\beta \geq \alpha$ [Rosenstein, 1982, Theorem 5.9].) From P7 it is easy to see that

P8 *$f^\gamma(C)$ is a singleton if and only if C is scattered* [Rosenstein, 1982, Exercise 5.11.2].

This gives a necessary and sufficient characterization of scattered concept chains in terms of $f^\gamma(C)$. So now all we need to do is related this characterization of a scattered chain C to its decomposition into limit points as given in the definition above. Below we prove

P9 *For any ordinal β, there can be at most* 2 *concepts from $C^{\beta+}$ in any interval of $f^\beta(C)$.*

This gives the result, since: From P8 we know that if C is scattered then $f^{\gamma}(C)$ is a single interval. Combined with P9, this means $C^{\gamma+}$ contains at most 2 concepts, and hence $C^{(\gamma+1)+} = \varnothing$. Since by definition $C = C^{\alpha+} \cup \bigcup_{\beta<\alpha} C^{\beta}$ for any α, we have shown that $C = \bigcup_{\beta\leq\gamma} C^{\beta}$; establishing P6 and hence the lemma.

Proof of P9: Proof is by induction on ordinals. *Base:* Simply define f^0 to be the singleton intervals of C. *Successor ordinal:* For any successor ordinal $\beta+1$ we know there are at most 2 concepts from $C^{\beta+}$ in any subinterval of f^{β} by the induction hypothesis. Now assume there are 3 concepts $a \subset b \subset c$ from $C^{(\beta+1)+}$ in a single interval of $f^{\beta+1}$. Since they are in a single interval in $f^{\beta+1}$, these concepts must have belonged to intervals in f^{β} that were only a finite distance apart. But then, by the induction hypothesis, $a \subset b \subset c$ must only be a finite distance apart in $C^{\beta+}$. But this means b must be isolated in $C^{\beta+}$, and hence cannot belong to $C^{(\beta+1)+}$; a contradiction. *Limit ordinal:* For any limit ordinal λ we know there are at most 2 concepts from $C^{\beta+}$ in any subinterval of f^{β}, for all $\beta < \lambda$, by the induction hypothesis. Now assume there are 3 concepts $a \subset b \subset c$ from $C^{\lambda+}$ in a single interval of f^{λ}. Since they are in a single interval in f^{λ}, there must be some $\beta_1 < \lambda$ such that a and b belong to an interval of f^{β_1}, and some $\beta_2 < \lambda$ such that b and c belong to an interval of f^{β_2}. But then all 3 concepts must belong to a single interval of f^{β} for $\beta = \max\{\beta_1, \beta_2\} < \lambda$; a contradiction. ▮

Lemma 4.18 *Any chain C that is closed under $\cup$, $\cap$ is also complete and bounded, and satisfies a natural version of the Bolzano-Weierstrass property.*

Proof We define the usual topological concepts for linear orderings (see *e.g.*, [Rosenstein, 1982, Chapter 2]).

Definition C.13 (Compactness properties) *For a chain C, a* Dedekind cut *of C is a partition of C into two nonempty subsets $\langle U, V\rangle$ where $u \subset v$ for all $u \in U$ and $v \in V$. A* gap *is a Dedekind cut $\langle U, V\rangle$ where U has no maximal concept and V has no minimal concept. We say that a chain C is:* (i) closed *if it is closed under $\cap$ and $\cup$;* (ii) bounded *if it has a minimal and maximal concept;* (iii) Dedekind-complete *if it has no gaps;* (iv) complete *if every upper (lower) bounded subchain of C has a least upper (greatest lower) bound in C; and* (v) Bolzano-Weierstrass *if every infinite subchain of C has a limit in C.*

The definitions and results of this lemma will be used to prove the next two lemmas below. Observe that being closed under $\cup$, $\cap$ implies being Dedekind-complete and bounded, complete and bounded, and Bolzano-Weierstrass. Therefore we call such a chain *compact*.

((i) $\Rightarrow$ (ii)) Obvious: the maximal concept is just $\bigcup\{c \in C\}$ and the minimal concept is $\bigcap\{c \in C\}$, which both must be in C since it is closed under $\cap$, $\cup$.

((i) $\Rightarrow$ (iii)) Consider any partition $\langle U, V\rangle$ of C. Since C is closed under $\cap$, $\cup$, both $\bigcup\{u \in U\}$ and $\bigcap\{v \in V\}$ are in C, meaning that $\langle U, V\rangle$ cannot be a gap.

((iii) $\Rightarrow$ (iv)) For any subchain A of C consider the partition defined by

$$U = \{u \in C : \ \exists a \in A \text{ such that } u \subseteq a\}.$$

Since $\langle U, V\rangle$ cannot be a gap, U must have a maximal concept or V a minimal concept. In either case we can supply a least upper bound on A.

((ii)+(iv) $\Rightarrow$ (v)) Without loss of generality, consider a countably infinite subchain A of C. Since C is complete and bounded, A must have a greatest lower bound $a_1 \in C$. If $a_1 \notin A$ we are done (since then a_1 would be a limit concept of A), so assume $a_1 \in A$. Continuing in this way, find $a_2 = glb(A - \{a_1\})$, $a_3 = glb(A - \{a_1, a_2\})$, *etc.* Note that if any a_i is not in A then it must be a limit concept of A, so we are left with the case where every $a_1 \subset a_2 \subset \ldots$ belongs to A. But then, since C is complete and bounded, $b = lub(\{a_i\}_1^{\infty})$ must also be in C, and since $b \notin \{a_i\}_1^{\infty}$ by construction, this must be a limit concept of A. ▮

Lemma 4.19 *If C is a scattered chain, then the chain $\mathcal{C}(C)$ formed by closing C under $\cup$, $\cap$ is still scattered.*

Proof Notice that every concept $c \in \mathcal{C}(C)$ either belongs to C, or is given by $c = \bigcup\{u \in U\}$ or $c = \bigcap\{v \in V\}$ for some gap $\langle U, V\rangle$. Thus, $\mathcal{C}(C) = C \cup \mathcal{U}(C) \cup \mathcal{V}(C)$, where $\mathcal{U}(C)$ denotes the concepts added for gaps where U has no maximal concept, and $\mathcal{V}(C)$ denotes the concepts added for gaps where V has no minimal concept. Assume that C is scattered but $\mathcal{C}(C)$ is somewhere-dense. We will show that this leads to a contradiction.

First, since $\mathcal{C}(C)$ is somewhere-dense the following proposition shows that one of $\mathcal{U}(C)$ or $\mathcal{V}(C)$ must also be somewhere-dense.

P10 *Removing a scattered subchain S from a dense chain D leaves a somewhere-dense chain $D - S$.*

(This proposition is easy to prove: Since S is scattered there must be two concepts $s_1 \subset s_2$ in S with no $s_3 \in S$ between them. But then the subinterval (s_1, s_2) of D, which is dense, is properly contained in $D - S$.) So, without loss of generality, assume $\mathcal{U}(C)$ contains a dense subchain U. Notice that C is *between* U in the sense that for every pair $u_1 \subset u_2$ in U there must be a $c \in C$ such that $u_1 \subset c \subset u_2$ (for if not, then there would be two distinct gaps $\langle U_1, V_1\rangle$, $\langle U_2, V_2\rangle$ of C with $U_2 - U_1 = \varnothing$, which cannot be). But then the following proposition shows that C must be somewhere-dense as well; a contradiction.

P11 *If a chain B is between a dense chain D, then B must also be somewhere-dense.*

To prove this proposition, first note that we can find concepts $b_1, b_2 \in B$, $d_1, d_2 \in D$ such that $b_1 \subset d_1 \subset d_2 \subset b_2$ (just pick four concepts $d_3 \subset d_1 \subset d_2 \subset d_4$ from D and choose b_1, b_2 between the first and last pairs respectively). Now, for any such quadruple $b_1 \subset d_1 \subset d_2 \subset b_2$ we can always find $b_3 \in B$ and $d_3, d_4 \in D$ such that $b_1 \subset d_1 \subset d_3 \subset b_3 \subset d_4 \subset d_2 \subset b_2$ (just choose d_3 and d_4 between d_1 and d_2, and b_3 between d_3 and d_4). Thus, we can continue this process indefinitely to construct a dense subchain of B. ▮

Lemma 4.20 *For a scattered chain C that is closed under $\cap$, $\cup$, there is always a concept $c \in C$ of* maximal *order consistent with any finite sequence of training examples.*

Proof Recall that any sequence of training examples $c\mathbf{x}^t$ determines an uncertainty interval $[s[c\mathbf{x}^t], \ell[c\mathbf{x}^t]]$. Also recall from the definition of an uncertainty interval (Definition C.1) that s and ℓ are defined by unions and intersections of concepts from C, and hence must also belong to C. This means that $[s, \ell]$ is a *compact* subinterval of C (*i.e.*, $[s, \ell]$ is also closed under $\cap$, $\cup$). Since $[s, \ell]$ is also scattered, by Lemma 4.17 we know that there exists a *least* ordinal γ such that $[s, \ell] = \bigcup_{\beta \leq \gamma} [s, \ell]^\beta$. (That is, γ is the least ordinal such that all limit concepts in $[s, \ell]$ have order at most γ.)

It suffices to show that $[s, \ell]^\gamma$ is non-empty, as this will supply the needed maximal order limit concepts. Assume $[s, \ell]^\gamma$ is empty. Then clearly γ must be infinite and $[s, \ell]^\beta$ must be non-empty for all $\beta < \gamma$ (otherwise γ would not be the least such ordinal). But then consider the subchain $\{c^\beta \in C^\beta : \beta < \gamma\}$ of $[s, \ell]$ formed by choosing a single limit concept of each order $\beta < \gamma$. By the compactness of $[s, \ell]$, this infinite subchain must have a *limit* concept c in $[s, \ell]$ (*cf.* the Bolzano-Weierstrass property of Lemma 4.18). This concept c cannot be in $[s, \ell]^\beta$ for any $\beta < \gamma$, and hence must belong to $[s, \ell]^{\gamma+}$; a contradiction. ▮

Theorem 4.21 (Scattered UB) *For any scattered concept chain C: The hypothesis guessing strategy* CHOLC *obtains an exponential learning curve (i.e.,* $\mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{CHOLC}, \mathrm{P}, c, \mathbf{x}^t) = e^{O(-t)}$ *) for every target concept $c \in C$, regardless of the domain distribution* P.

Proof By Lemmas 4.19 and 4.20, CHOLC is well defined for any scattered concept chain C. By Lemma 4.17, CHOLC is guaranteed to achieve exponential convergence for any target concept c in C, since fixing a domain distribution preserves the order structure of the chain, or collapses subintervals of the chain together. (Note that collapsing subintervals cannot produce new limit concepts or increase the order of existing target concepts; beyond identifying them with already existing such concepts.) Therefore, the result holds for any domain distribution P. ▮

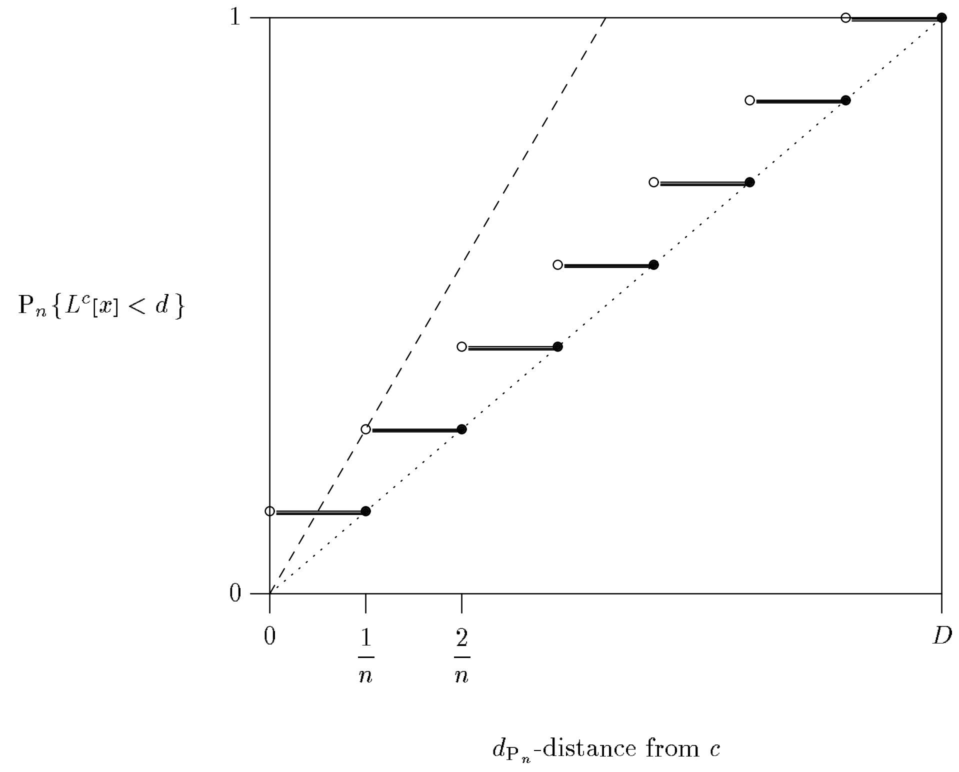

Figure C.6: Distribution of d_{P_n}-distances from a target concept c to its right-side neighbors.

C.5 Other results

Proposition 4.24 *For any $c \in (C_n, \mathrm{P}_n)$,*

$$\left(1 - \frac{1}{n}\right)^t \frac{1}{4(t+1)} \;<\; \mathrm{E}_{\mathbf{x}^t}\, wid(C_n, \mathrm{P}_n, c, \mathbf{x}^t) \;<\; \left(1 - \frac{2}{n}\right)^t \frac{2}{t+1}.$$

Proof (Sketch of upper bound) Notice that we can embed the discrete chain (C_n, P_n) in a uniform chain (I, U). This means that we can lower bound the distribution of distances from a target concept $c \in C_n$ by a uniform distribution as shown in Figure C.6. Therefore, recalling that $F_X \le F_Y$ implies $\mathrm{E}X \ge \mathrm{E}Y$ for $X \ge 0$ and $Y \ge 0$, we can upper bound the expected width of the uncertainty interval by assuming c is in a uniform chain, and multiplying this width by the probability that neither of c's nearest neighbors is eliminated. Applying Lemma C.4 then gives

$$\mathrm{E}_{\mathbf{x}^t}\, wid(C_n, \mathrm{P}_n, c, \mathbf{x}^t) \;<\; \left(1 - \frac{2}{n}\right)^t \frac{2}{t+1}.$$

(Sketch of lower bound) Consider an arbitrary target concept $c \in C_n$. We know that c must be at least a distance $D \ge \frac{1}{2}(1 - \frac{1}{n})$ from one end of the chain, so without loss of generality, assume $\mathrm{P}(X - c) \ge D$. Now consider the distance between c and the largest consistent concept in the chain, which we measure by a random variable L^c. From Figure C.6 we can see that given c's nearest neighbor is not eliminated, the distribution of distances from c to its largest consistent neighbor can be upper bounded by a $\mathsf{uniform}(0, K)$ distribution, for $K = \frac{D}{2(1-1/n)} \ge 1/4$. Therefore, we can lower bound this expected distance by

$$\mathrm{E}_{\mathbf{x}^t}\left[\underline{L^c}_t(\mathbf{x}^t)\middle|\, x_1, ..., x_t \in X - c\right]\; \mathrm{P}_x(X - c)^t \quad\ge\quad \frac{1}{4(t+1)}\left(1 - \frac{1}{n}\right)^t. \;\blacksquare$$

Theorem 4.29 *For any space (C, P) such that $N_\alpha(C,\mathrm{P}) = \Theta(1/\alpha)^d$ for some d as $\alpha \to 0$:*

1. *Strategy* BC *achieves* $\mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{BC}, \mathrm{P}, c, \mathbf{x}^t) = O((d/t)\ln(t/d))$ *for any target* $c \in C$.
2. *Any hypothesizer* H *obtains* $\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c', \mathbf{x}^t) = e^{\Omega'(-t/d)}$ *for some target* $c' \in C$.

Proof ***Part 1:*** Let a and d be constants such that $N_\alpha(C,\mathrm{P}) \le (a/\alpha)^d$. Fix an arbitrary $t \ge 32ed/a$ and let $\alpha \triangleq \alpha_t = (32d/t)\ln(at/[32d])$, as given in the definition of BC (Figure 4.6). First, it is obvious that

$$\mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{BC}, \mathrm{P}, c, \mathbf{x}^t) \quad \le \quad 2\alpha + \mathrm{P}_{x^t}\left\{err(\mathsf{BC}, \mathrm{P}, c, \mathbf{x}^t) \ge 2\alpha\right\}. \tag{C.14}$$

Now, to upper bound the probability term, recall that BC constructs an α-cover V of the space with size $|V| = N_\alpha(C,\mathrm{P})$ and returns the hypothesis in V that obtains minimum empirical error on the training sequence. By construction, V is guaranteed to contain a hypothesis h^* with true error at most α. Therefore,

$$\begin{aligned}
&\mathrm{P}_{x^t}\left\{err(\mathsf{BC}, \mathrm{P}, c, \mathbf{x}^t) \ge 2\alpha\right\} \\
&\quad = \quad \mathrm{P}_{x^t}\left\{\text{for } h' = \mathrm{argmin}_{h\in V}\ emp.err.(h, c\mathbf{x}^t) \text{ we have } err(h') \ge 2\alpha\right\} \\
&\quad \le \quad \mathrm{P}_{x^t}\left\{\begin{array}{l} emp.err.(h^*, c\mathbf{x}^t) \ge 3\alpha/2 \text{ or} \\ \exists h' \in V \text{ with } err(h') \ge 2\alpha \text{ and } emp.err.(h', c\mathbf{x}^t) \le 3\alpha/2 \end{array}\right\} \\
&\quad \le \quad \mathrm{P}\left\{\, p \le \alpha \text{ yet } \nu_t \le 3\alpha/2 \,\right\} \ + \ (|V|-1)\ \mathrm{P}\left\{\, p \ge 2\alpha \text{ yet } \nu_t \le 3\alpha/2 \,\right\} \\
&\quad < \quad |V| e^{-t\alpha/16},
\end{aligned} \tag{C.15}$$

by Chernoff bounds [Hagerup and Rüb, 1989/90]. Finally, plugging (C.15) into (C.14) and using the fact that $|V| = N_\alpha(C,\mathrm{P}) \le (a/\alpha)^d$, we get

$$\begin{aligned}
\mathrm{E}_{\mathbf{x}^t}\, err(\mathsf{BC}, \mathrm{P}, c, \mathbf{x}^t) \quad &\le \quad 2\alpha + \left(\frac{a}{\alpha}\right)^d e^{-t\alpha/16} \\
&= \quad 2\alpha + \exp\left(-d \ln \frac{32d}{at} \ln \frac{at}{32d} - 2d \ln \frac{at}{32d}\right) \\
&\le \quad 2\alpha + \exp\left(-d \ln \frac{32d}{at} - 2d \ln \frac{at}{32d}\right) \\
&\qquad \text{since } \ln \tfrac{32d}{at} \ge 1 \text{ for } t \ge 32ed/a, \\
&= \quad 2\alpha + \exp\left(-d \ln \frac{at}{32d}\right) \\
&= \quad \frac{64d}{t} \ln \frac{at}{32d} + \left(\frac{32d}{at}\right)^d \\
&\le \quad \frac{64d}{t} \ln \frac{at}{32d} + \frac{32d}{at} \ln \frac{at}{32d} \qquad \text{for } t \ge 32ed/a,\ d \ge 1, \\
&= \quad \frac{96d}{t} \ln \frac{at}{32d}.
\end{aligned}$$

Part 2: We prove this part by: *(i)* explicitly constructing a subset of the original class C, *(ii)* defining a prior over this subset, and then *(iii)* arguing that any hypothesizer H must obtain a large expected error *on average* over this subset, for all training sample sizes t. To do this we will construct "α-packings" of the space, which consist of collections of pairwise α-separated concepts. If we let $M_\alpha(C, \mathrm{P})$ denote the size of the largest α-packing of the space (C, P), then it is well known that $N_\alpha(C,\mathrm{P}) \le M_\alpha(C,\mathrm{P}) \le N_{\alpha/2}(C,\mathrm{P})$ [Kolmogorov and Tihomirov, 1961]. Therefore, since we have constants b and d such that $N_\alpha(C,\mathrm{P}) \ge (b/\alpha)^d$ by assumption, we know that $M_\alpha(C,\mathrm{P}) \ge (b/\alpha)^d$.

Now, we define a subset of C by constructing a series of α-packings of (C,P) at various scales $\alpha = \alpha_0, \alpha_1, ..., etc.$ Specifically, let $\alpha_n = b\,3^{-n/d}$ for $n = 0, 1, 2, ...$, and consider α-packings U_{α_0}, U_{α_1}, ... of size $|U_{\alpha_n}| = 3^n$. (We know there must be α-packings of size at least $M_{\alpha_n} \ge (b/\alpha_n)^d = 3^n$ for each α_n.) Then define the disjoint subsets

$$V_{\alpha_n} = U_{\alpha_n} - \sum_{i=0}^{n-1} U_{\alpha_i},$$

where each V_{α_n} contains at least $3^n/2$ pairwise α_n-separated concepts. (We know that V_{α_n} contains this many concepts, since $|V_{\alpha_n}| \ge |U_{\alpha_n}| - \sum_{i=0}^{n-1} |U_{\alpha_i}| = 3^n - \sum_{i=0}^{n-1} 3^i = 3^n - (3^n-1)/2 > 3^n/2$.) Now consider the concept class $C_0 = \bigcup_{n=0}^{\infty} V_{\alpha_n}$, and define a prior Q on C_0 by setting $\mathrm{Q}(V_{\alpha_n}) = (1-3^{-1/d})3^{-n/d}$ for each V_{α_n} and defining Q to be uniform within each class V_{α_n}. (The constant $1-3^{-1/d}$ merely ensures $\sum_{n=0}^{\infty} \mathrm{Q}(V_{\alpha_n}) = 1$.) Then for an arbitrary hypothesizer H we can calculate its average expected error for t training examples as follows.

First notice that

$$\mathrm{E}_c \mathrm{E}_{\mathbf{x}^t}\, err(H,\mathrm{P},c,\mathbf{x}^t) \;=\; \mathrm{E}_{\mathbf{x}^t} \mathrm{E}_c\, err(H,\mathrm{P},c,\mathbf{x}^t)$$

by Fubini's theorem. Now for an arbitrary fixed $\mathbf{x}^t$ we obtain

$$\begin{aligned}
\mathrm{E}_c\, err(H,\mathrm{P},c,\mathbf{x}^t) &= \sum_{n=0}^{\infty} \mathrm{E}_c \left[\, err(H,\mathrm{P},c,\mathbf{x}^t) \,\middle|\, c \in V_{\alpha_n} \right] \mathrm{Q}(V_{\alpha_n}) \\
&> \sum_{n=t+1}^{\infty} \frac{\alpha_n}{2}\left(1 - \frac{2^t}{|V_{\alpha_n}|}\right) \mathrm{Q}(V_{\alpha_n}) \\
&\quad \text{by Lemma C.14 below (noting } 2^t < |V_{\alpha_n}| \text{ for } n \ge t+1\text{)}, \\
&= \sum_{n=t+1}^{\infty} \frac{b}{2}\, 3^{-n/d}(1 - 2^{t+1}3^{-n})(1-3^{-1/d})3^{-n/d} \\
&\ge \frac{b(1-3^{-1/d})}{6} \sum_{n=t+1}^{\infty} 3^{-2n/d} \\
&= \frac{b(1-3^{-1/d})}{6(1-3^{-2/d})}\, 3^{-2(t+1)/d} \\
&\ge \frac{b}{12}\, 3^{-2(t+1)/d}.
\end{aligned}$$

The second inequality follows since $1 - 2^{t+1}3^{-n} \ge 1/3$ for $n \ge t+1$, and the last inequality follows because $(1-3^{-1/d})/(1-3^{-2/d}) = (1-x)/(1-x^2) \ge 1/2$ for $x = 1/3 \to 1$. Thus, for the prior Q defined on C_0 we obtain

$$\mathrm{E}_c \mathrm{E}_{\mathbf{x}^t}\, err(H,\mathrm{P},c,\mathbf{x}^t) \;\ge\; \frac{b}{12} 3^{-2(t+1)/d}. \tag{C.16}$$

Finally, we need to prove that this forces H to obtain $\mathrm{E}_{\mathbf{x}^t}\, err(H,\mathrm{P},c',\mathbf{x}^t) = e^{\Omega'(-t/d)}$ on some $c' \in C$ for infinitely many training sample sizes t. That is,

$$\exists c' \in C,\ \exists \beta > 0 \text{ s.t. } \forall t_0,\ \exists t > t_0 \text{ s.t. } \mathrm{E}_{\mathbf{x}^t}\, err(H,\mathrm{P},c',\mathbf{x}^t) \ge e^{-\beta t/d}.$$

Assume to the contrary that

$$\forall c \in C,\ \forall \beta > 0,\ \exists t_0 \text{ s.t. } \forall t > t_0,\ \mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c', \mathbf{x}^t) < e^{-\beta t/d}.$$

This means that for *all* priors Q on C we must obtain

$$\forall \mathrm{Q},\ \forall \beta > 0,\ \exists t_0 \text{ s.t. } \forall t > t_0,\ \mathrm{E}_c\left[\mathrm{E}_{\mathbf{x}^t}\, err(H, \mathrm{P}, c', \mathbf{x}^t)\right] < e^{-\beta t/d},$$

but this directly contradicts (C.16), and we are done. ∎

Lemma C.14 *For any pairwise α-separated set of concepts V with a* uniform *prior* Q *on V: Any hypothesizer H must obtain an average error on t training objects $\mathbf{x}^t$ (where $2^t \le |V_\alpha|$) of at least*

$$\mathrm{E}_c err(H, \mathrm{P}, c, \mathbf{x}^t) \;>\; \frac{\alpha}{2}\left(1 - \frac{2^t}{|V|}\right).$$

Proof First, it is obvious that

$$\begin{aligned} \mathrm{E}_c err(H, \mathrm{P}, c, \mathbf{x}^t) &> \frac{\alpha}{2}\,\mathrm{Q}\left\{c : err(H, \mathrm{P}, c, \mathbf{x}^t) > \frac{\alpha}{2}\right\} \\ &= \frac{\alpha}{2}\left(1 - \mathrm{Q}\left\{c : err(H, \mathrm{P}, c, \mathbf{x}^t) \le \frac{\alpha}{2}\right\}\right). \end{aligned} \tag{C.17}$$

Finally, Claim B.2 in Appendix B shows that

$$\mathrm{Q}\left\{c : err(H, \mathrm{P}, c, \mathbf{x}^t) \le \frac{\alpha}{2}\right\} \;\le\; \frac{2^t}{|V|},$$

so combining this with (C.17) directly gives the result. ∎

Bibliography

Aha, D. W., Kibler, D., and Albert, M. K. 1991. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66.

Albert, M. K. and Aha, D. W. 1991. Analyses of instance-based learning algorithms. In *Proceedings of the Ninth American National Conference on Artificial Intelligence (AAAI-91)*, pages 553–558.

Amari, S., Fujita, N., and Shinomoto, S. 1992. Four types of learning curves. *Neural Computation*, 4:605–618.

Angluin, D. 1988. Queries and concept learning. *Machine Learning*, 2(4):319–342.

Angluin, D. and Laird, P. 1988. Learning from noisy examples. *Machine Learning*, 2(4):343–370.

Ash, R. B. 1972. *Real Analysis and Probability.* Academic Press, San Diego.

Barnard, E. 1994. A model for nonpolynomial decrease in error rate with increasing sample size. *IEEE Transactions on Neural Networks*, 5(6):994–997.

Bartlett, P. L. and Williamson, R. C. 1991. Investigating the distributional assumptions of the pac learning model. In *Proceedings of the Fourth Annual Conference on Computational Learning Theory (COLT-91)*, pages 24–32.

Baum, E. B. 1990. The perceptron algorithm is fast for nonmalicious distributions. *Neural Computation*, 2:248–260.

Baum, E. B. and Haussler, D. 1989. What size net gives valid generalization? *Neural Computation*, 1:151–160.

Baum, E. B. and Lyuu, Y.-D. 1991. The transition to perfect generalization in perceptrons. *Neural Computation*, 3:386–401.

Ben-David, S., Benedek, G. M., and Mansour, Y. 1989. A parameterization scheme for classifying models of learnability. In *Proceedings of the Second Annual Conference on Computational Learning Theory (COLT-89)*, pages 285–302.

Ben-David, S., Cesa-Bianchi, N., and Long, P. M. 1992. Characterizations of learnability for classes of $\{0, ..., n\}$-valued functions. In *Proceedings of the Fifth Annual Conference on Computational Learning Theory (COLT-92)*, pages 333–340.

Benedek, G. and Itai, A. 1988a. Learnability by fixed distributions. In *Proceedings of the Conference on Computational Learning Theory (COLT-88)*, pages 80–90.

Benedek, G. and Itai, A. 1988b. Nonuniform learnability. In *Proceedings of the Fifteenth International Colloquium on Automata, Languages and Programming (ICALP-88)*, pages 82–92.

Benedek, G. and Itai, A. 1991. Learnability with respect to fixed distributions. *Theoretical Computer Science*, 86:377–389.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. 1989. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. 1984. *Classification and Regression Trees.* Wadsworth, Belmont, CA.

Brualdi, R. A. 1977. *Introductory Combinatorics.* North-Holland, New York.

Buchanan, B. G. and Mitchell, T. M. 1978. Model directed learning of production rules. In Waterman, D. A. and Hayes-Roth, F., editors, *Pattern Directed Inference Systems.* Academic Press, New York.

Chernoff, H. 1972. *Sequential Analysis and Optimal Design.* SIAM, Philadelphia.

Clancey, W. J. 1985. Heuristic classification. *Artificial Intelligence*, 27:289–350.

Cohn, D. and Tesauro, G. 1990. Can neural networks do better than the Vapnik-Chervonenkis bounds? In Touretzky, D., editor, *Advances in Neural Information Processing Systems 3.* Morgan Kaufmann, San Mateo, CA.

Cohn, D. and Tesauro, G. 1992. How tight are the Vapnik-Chervonenkis bounds? *Neural Computation*, 4:249–269.

Dennis, J. E. and Schnabel, R. B. 1983. *Numerical Methods for Unconstrained and Nonlinear Equations.* Prentice-Hall, Englewood Cliffs, NJ.

Duda, R. O. and Hart, P. 1973. *Pattern Classification and Scene Analysis.* Wiley, New York.

Dudley, R. M., Kulkarni, S., Richardson, T., and Zeitouni, O. 1994. A metric entropy bound is not sufficient for learnability. *IEEE Transactions on Information Theory*, 40(3):883–885.

Ehrenfeucht, A., Haussler, D., Kearns, M. J., and Valiant, L. 1989. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82:247–261.

Furst, M. L., Jackson, J. C., and Smith, S. W. 1991. Improved learning of AC^0 functions. In *Proceedings of the Fourth Annual Conference on Computational Learning Theory (COLT-91)*, pages 317–325.

Gallant, S. I. 1990. Perceptron-based learning algorithms. *IEEE Transactions on Neural Networks*, 2(1):179–191.

Geman, S., Bienenstock, E., and Doursat, R. 1992. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58.

Gold, E. M. 1967. Language identification in the limit. *Information and Control*, 10:447–474.

Goldman, S., Kearns, M. J., and Schapire, R. 1990. Exact identification of circuits using fixed points of amplification functions. In *Proceedings of the Thirty First Annual IEEE Symposium on Foundations of Computer Science (FOCS-90)*, pages 193–202.

Golea, M. and Marchand, M. 1993. Average case analysis of the clipped Hebb rule for nonoverlapping Perceptron networks. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory (COLT-93)*, pages 151–157.

Hagerup, T. and Rüb, C. 1989/90. A guided tour of Chernoff bounds. *Information Processing Letters*, 33:305–308.

Hampson, S. E. and Volper, D. J. 1986. Linear function neurons: Structure and training. *Biological Cybernetics*, 53:203–217.

Hancock, T. and Mansour, Y. 1991. Learning monotone $k\mu$ DNF formulas on product distributions. In *Proceedings of the Fourth Annual Conference on Computational Learning Theory (COLT-91)*, pages 179–183.

Haussler, D. 1988. Quantifying inductive bias: AI learning algorithms and Valiant's learning framework. *Artificial Intelligence*, 36:117–221.

Haussler, D. 1992. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100:78–150.

Haussler, D., Kearns, M. J., and Schapire, R. 1991. Bounds on the sample complexity of Bayesian learning using information theory and the VC dimension. In *Proceedings of the Fourth Annual Conference on Computational Learning Theory (COLT-91)*, pages 61–74.

Haussler, D., Kearns, M. J., Seung, H. S., and Tishby, N. 1994. Rigorous learning curve bounds from statistical mechanics. In *Proceedings of the Seventh Annual Conference on Computational Learning Theory (COLT-94)*, pages 76–87.

Haussler, D., Littlestone, N., and Warmuth, M. K. 1988. Predicting {0,1}-functions on randomly drawn points. In *Proceedings of the Conference on Computational Learning Theory (COLT-88)*, pages 280–296.

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

Hinton, G. 1989. Unpublished lecture notes.

Hinton, G. 1995. Personal communication.

Jackson, J. 1994. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. In *Proceedings of the Thirty Fifth Annual IEEE Symposium on Foundations of Computer Science (FOCS-94)*, pages 42–52.

Kearns, M. J. 1993. Efficient noise-tolerant learning from statistical queries. In *Proceedings of the Twenty Fifth Annual ACM Symposium on Theory of Computing (STOC-93)*, pages 392–401.

Kearns, M. J. and Li, M. 1988. Learning in the presence of malicious errors. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC-88)*, pages 267–280.

Kearns, M. J., Li, M., Pitt, L., and Valiant, L. 1987a. On the learnability of boolean formulae. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing (STOC-87)*, pages 285–295.

Kearns, M. J., Li, M., Pitt, L., and Valiant, L. 1987b. Recent results on boolean concept learning. In *Proceedings of the Fourth International Conference on Machine Learning (ML-87)*, pages 337–352.

Kearns, M. J. and Valiant, L. G. 1989. Cryptographic limitations on learning Boolean formulae and finite automata. In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing (STOC-89)*, pages 433–444.

Kharitonov, M. 1993. Cryptographic hardness of distribution-specific learning. In *Proceedings of the Twenty Fifth Annual ACM Symposium on Theory of Computing (STOC-93)*, pages 372–381.

Kolmogorov, A. N. and Tihomirov, V. M. 1961. ϵ-entropy and ϵ-capacity of sets in functional spaces. *Amer. Math. Soc. Transl. Ser. 2*, 17:277–364.

Kulkarni, S. 1991. *Problems of Computational and Information Complexity in Machine Vision and Learning.* PhD thesis, MIT, EECS Department.

Langley, P., Iba, W., and Thompson, K. 1992. An analysis of Bayesian classifiers. In *Proceedings of the Tenth American National Conference on Artificial Intelligence (AAAI-92)*, pages 223–228.

Larsen, R. J. and Marx, M. L. 1981. *An Introduction to Mathematical Statistics and its Applications.* Prentice-Hall, Englewood Cliffs, NJ.

le Cun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, D. E., Hubbard, W., and Jackel, L. D. 1989. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551.

Linial, N., Mansour, Y., and Nisan, N. 1989. Constant depth circuits, Fourier transform, and learnability. In *Proceedings of the Thirtieth Annual IEEE Symposium on Foundations of Computer Science (FOCS-89)*, pages 574–579.

Linial, N., Mansour, Y., and Rivest, R. L. 1991. Results on learnability and the Vapnik-Chervonenkis dimension. *Information and Computation*, 90:33–49.

Littlestone, N. 1988. Learning quickly when irrelevant attributes abound: A new linear threshold algorithm. *Machine Learning*, 2(4):285–318.

Littlestone, N. 1989. From online to batch learning. In *Proceedings of the Second Annual Conference on Computational Learning Theory (COLT-89)*, pages 269–284.

Lyuu, Y.-D. and Rivin, I. 1992. Tight bounds on transition to perfect generalization in Perceptrons. *Neural Computation*, 4:854–862.

Michalski, R. S. 1983. A theory and methodology of inductive learning. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning: An Artificial Intelligence Approach*, pages 83–129. Morgan Kaufmann, Los Altos, CA.

Minsky, M. L. and Papert, S. A. 1969. *Perceptrons.* MIT Press, Cambridge, MA.

Mitchell, T. M. 1980. The need for biases in learning generalizations. Technical Report CBM-TR-117, Rutgers University.

Nilsson, N. J. 1965. *Learning Machines.* Morgan Kaufmann, San Mateo, CA.

Oblow, E. M. 1992. Implementing Valiant's learnability theory using random sets. *Machine Learning*, 8(1):45–73.

Opper, M. and Haussler, D. 1991. Generalization performance of Bayes optimal classification algorithm for learning a Perceptron. *Physical Review Letters*, 66(20):2677–2680.

Pazzani, M. J. and Sarrett, W. 1990. Average case analysis of conjunctive learning algorithms. In *Proceedings of the Seventh International Conference on Machine Learning (ML-90)*, pages 339–347.

Piatetsky-Shapiro, G. and Frawley, W. J., editors 1991. *Knowledge Discovery in Databases.* AAAI Press, Menlo Park, CA.

Pitt, L. and Valiant, L. G. 1988. Computational limitations on learning from examples. *Journal of the ACM*, 35(4):965–984.

Pollard, D. 1984. *Convergence of Stochastic Processes.* Springer-Verlag, New York.

Purdom, P. W. J. and Brown, C. A. 1985. *The Analysis of Algorithms.* Holt, Rinehart and Winston, New York.

Qian, N. and Sejnowski, T. J. 1988. Predicting the secondary structure of globular proteins using neural network models. *Journal of Molecular Biology*, 202:865–884.

Quinlan, J. R. 1986. Induction of decision trees. *Machine Learning*, 1(1):81–106.

Rivest, R. L. 1987. Learning decision lists. *Machine Learning*, 2(3):229–246.

Rosenstein, J. G. 1982. *Linear Orderings.* Academic Press, New York.

Schaffer, C. 1994. A conservation law for generalization performance. In *Proceedings of the Eleventh International Conference on Machine Learning (ML-94)*, pages 259–265.

Schapire, R. E. 1992. *The Design and Analysis of Efficient Learning Algorithms.* MIT Press, Cambridge, MA.

Schuurmans, D. 1995. Characterizing rational versus exponential learning curves. In *Proceedings of the Second European Conference on Computational Learning Theory (EuroCOLT-95)*, pages 272–286.

Schuurmans, D. 1996a. Characterizing rational versus exponential learning curves. *Journal of Computer and System Sciences*. Invited submission to special issue. (Under review).

Schuurmans, D. 1996b. Fast distribution-specific learning. In Greiner, R., Petsche, T., Hanson, S., and Rivest, R., editors, *Computational Learning Theory and Natural Learning Systems*, volume 4. MIT Press, Cambridge, MA. (In press).

Schuurmans, D. and Greiner, R. 1995a. Practical PAC learning. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 1169–1175.

Schuurmans, D. and Greiner, R. 1995b. Sequential PAC learning. In *Proceedings of the Eighth Annual Conference on Computational Learning Theory (COLT-95)*, pages 377–384.

Schwartz, D. B., Samalam, V. K., Solla, S. A., and Denker, J. S. 1990. Exhaustive learning. *Neural Computation*, 2:374–385.

Seung, H. S., Sompolinsky, H., and Tishby, N. 1991. Learning curves in large neural networks. In *Proceedings of the Fourth Annual Conference on Computational Learning Theory (COLT-91)*, pages 112–127.

Shawe-Taylor, J., Anthony, M., and Biggs, N. L. 1993. Bounding sample size with the Vapnik-Chervonenkis dimension. *Discrete Applied Mathematics*, 42:65–73.

Shiryayev, A. N. 1978. *Optimal Stopping Rules*. Springer-Verlag, New York.

Valiant, L. G. 1984. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142.

Vapnik, V. N. and Chervonenkis, A. Y. 1971. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280.

Verbeurgt, K. 1990. Learning DNF under the uniform distribution in quasi-polynomial time. In *Proceedings of the Third Annual Conference on Computational Learning Theory (COLT-90)*, pages 314–326.

Wald, A. 1947. *Sequential Analysis*. John Wiley & Sons, New York.

Wantanabe, S. 1987. Inductive ambiguity and the limits of artificial intelligence. *Computational Intelligence*, 3(4):304–309.

Weiss, S. M. and Kulikowski, C. A. 1991. *Computer Systems that Learn*. Morgan Kaufmann, San Mateo, CA.