# Part 1: Overview of the Probably Approximately Correct (PAC) Learning Framework*

David Haussler
haussler@cse.ucsc.edu
Baskin Center for Computer Engineering and Information Sciences
University of California, Santa Cruz, CA 95064

## Summary of Part 1

Here we survey some recent theoretical results on the efficiency of machine learning algorithms. The main tool described is the notion of Probably Approximately Correct (PAC) learning, introduced by Valiant. We define this learning model and then look at some of the results obtained in it. We then consider some criticisms of the PAC model and the extensions proposed to address these criticisms. Finally, we look briefly at other models recently proposed in computational learning theory.

## Introduction

It's a dangerous thing to try to formalize an enterprise as complex and varied as machine learning so that it can be subjected to rigorous mathematical analysis. To be tractable, a formal model must be simple. Thus, inevitably, most people will feel that important aspects of the activity have been left out of the theory. Of course, they will be right. Therefore, it is not advisable to present a theory of machine learning as having reduced the entire field to its bare essentials. All that can be hoped for is that some aspects of the phenomenon are brought more clearly into focus using the tools of mathematical analysis, and that perhaps a few new insights are gained. It is in

*The support from ONR grants N00014-86-K-0454 and N00014-91-J-1162 is gratefully acknowledged. A preliminary version of this part of the chapter appeared in [52].

this light that we wish to discuss the results obtained in the last few years in what is now called PAC (Probably Approximately Correct) learning theory [6].

Valiant introduced this theory in 1984 [124] to get computer scientists who study the computational efficiency of algorithms to look at learning algorithms. By taking some simplified notions from statistical pattern recognition and decision theory, and combining them with approaches from computational complexity theory, he came up with a notion of learning problems that are feasible, in the sense that there is a polynomial time algorithm that "solves" them, in analogy with the class **P** of feasible problems in standard complexity theory. Valiant was successful in his efforts. Since 1984 many theoretical computer scientists and AI researchers have either obtained results in this theory, or complained about it and proposed modified theories, or both.

The field of research that includes the PAC theory and its many relatives has been called computational learning theory. It is far from being a monolithic mathematical edifice that sits at the base of machine learning; it's unclear whether such a theory is even possible or desirable. We argue, however, that insights have been gained from the varied work in computational learning theory. The purpose of this short monograph is to survey some of this work and reveal those insights.

## Definition of PAC Learning

The intent of the PAC model is that successful learning of an unknown target concept should entail obtaining, with high probability, a hypothesis that is a good approximation of it. Hence the name Probably Approximately Correct. In the basic model, the instance space is assumed to be $\{0,1\}^n$, the set of all possible assignments to $n$ Boolean variables (or *attributes*) and concepts and hypotheses are subsets of $\{0,1\}^n$. The notion of approximation is defined by assuming that there is some probability distribution $D$ defined on the instance space $\{0,1\}^n$, giving the probability of each instance. We then let the *error* of a hypothesis $h$ w.r.t. a fixed target concept $c$, denoted $error(h)$ when $c$ is clear from the context, be defined by

$$error(h) = \sum_{x \in h \Delta c} D(x),$$

where $\Delta$ denotes the symmetric difference. Thus, $error(h)$ is the probability that $h$ and $c$ will disagree on an instance drawn randomly according to $D$. The hypothesis $h$ is a good approximation of the target concept $c$ if $error(h)$ is small.

How does one obtain a good hypothesis? In the simplest case one does this by looking at independent random examples of the target concept $c$, each example consisting of an instance selected randomly according to $D$, and a label that is "+" if that instance is in the target concept $c$ (*positive example*), otherwise "−" (*negative example*). Thus, training and testing use the same distribution, and there is no "noise" in either phase. A learning algorithm is then a computational procedure that takes a sample of the target concept $c$, consisting of a sequence of independent random examples of $c$, and returns a hypothesis.

For each $n \geq 1$ let $C_n$ be a set of target concepts over the instance space $\{0,1\}^n$, and let $\mathbf{C} = \{C_n\}_{n \geq 1}$. Let $H_n$, for $n \geq 1$, and $\mathbf{H}$ be defined similarly. We can define PAC learnability as follows: The concept class $\mathbf{C}$ is PAC learnable by the hypothesis space $\mathbf{H}$ if there exists a polynomial time learning algorithm $A$ and a polynomial $p(\cdot,\cdot,\cdot)$ such that for all $n \geq 1$, all target concepts $c \in C_n$, all probability distributions $D$ on the instance space $\{0,1\}^n$, and all $\epsilon$ and $\delta$, where $0 < \epsilon, \delta < 1$, if the algorithm $A$ is given at least $p(n, 1/\epsilon, 1/\delta)$ independent random examples of $c$ drawn according to $D$, then with probability at least $1 - \delta$, $A$ returns a hypothesis $h \in H_n$ with $error(h) \leq \epsilon$. The smallest such polynomial $p$ is called the *sample complexity* of the learning algorithm $A$.

The intent of this definition is that the learning algorithm must process the examples in polynomial time, i.e. be computationally efficient, and must be able to produce a good approximation to the target concept with high probability using only a reasonable number of random training examples. The model is worst case in that it requires that the number of training examples needed be bounded by a single fixed polynomial for all target concepts in $\mathbf{C}$ and all distributions $D$ in the instance space. It follows that if we fix the number of variables $n$ in the instance space and the confidence parameter $\delta$, and then invert the sample complexity function to plot the error $\epsilon$ as a function of training sample size, we do not get what is usually thought of as a learning curve for $A$ (for this fixed confidence), but rather the upper envelope of all learning curves for $A$ (for this fixed confidence), obtained by varying the target concept and distribution on the instance space. Needless to say, this is not a curve that can be observed experimentally. What is usually plotted experimentally is the error versus the training sample size for particular target concepts on instances chosen randomly according to a single fixed distribution on the instance space. Such a curve will lie below the curve obtained by inverting the sample complexity. We will return to this point later.

Another thing to notice about this definition is that target concepts in a concept class **C** may be learned by hypotheses in a different class **H**. This gives us some flexibility. Two cases are of interest. The first is that $\mathbf{C} = \mathbf{H}$, i.e. the target class and hypothesis space are the same. In this case we say that **C** is *properly* PAC learnable. Imposing the requirement that the hypothesis be from the class **C** may be necessary, e.g. if it is to be included in a specific knowledge base with a specific inference engine. However, as we will see, it can also make learning more difficult. The other case is when we don't care at all about the hypothesis space **H**, so long as the hypotheses in **H** can be evaluated efficiently. This occurs when our only goal is accurate and computationally efficient prediction of future examples. Being able to freely choose the hypothesis space may make learning easier. If **C** is a concept class and there exists some hypothesis space **H** such that hypotheses in **H** can be evaluated on given instances in polynomial time and such that **C** is PAC learnable by **H**, then we will say simply that **C** is *PAC learnable*.

There are many variants of the basic definition of PAC learnability. One important variant defines a notion of syntactic complexity of target concepts and, for each $n \geq 1$, further classifies each concept in $C_n$ by its syntactic complexity. Usually the syntactic complexity of a concept $c$ is taken to be the length of (number of symbols or bits in) the shortest representation of $c$ in a fixed concept representation language. In this variant of PAC learnability, the number of training examples is also allowed to grow polynomially in the syntactic complexity of the target concept. This variant is used whenever the concept class is specified by a concept representation language that can represent any boolean function, for example, when discussing the learnability of DNF (Disjunctive Normal Form) formulae or decision trees. Other variants of the model let the algorithm request examples, use separate distributions for drawing positive and negative examples, or use randomized (i.e. coin flipping) algorithms [63]. It can be shown that these latter variants are equivalent to the model described here, in that, modulo some minor technicalities, the concept classes that are PAC learnable in one model are also PAC learnable in the other [55]. Finally, the model can easily be extended to non-Boolean attribute-based instance spaces [49] and instance spaces for structural domains such as the blocks world [50]. Instances can also be defined as strings over a finite alphabet so that the learnability of finite automata, context-free grammars, etc. can be investigated [100].

## Outline of Results for the Basic PAC model

A number of fairly sharp results have been found for the notion of proper PAC learnability. The following summarizes some of these results. For precise definitions of the concept classes involved, the reader is referred to the literature cited. The negative results are based on the complexity theoretic assumption that $\mathbf{RP} \neq \mathbf{NP}$ [101].

1. Conjunctive concepts are properly PAC learnable [124], but the class of concepts in the form of the disjunction of two conjunctions is not properly PAC learnable [101], and neither is the class of existential conjunctive concepts on structural instance spaces with two objects [50].

2. Linear threshold concepts (perceptrons) are properly PAC learnable on both Boolean and real-valued instance spaces [24], but the class of concepts in the form of the conjunction of two linear threshold concepts is not properly PAC learnable [22]. The same holds for disjunctions and linear thresholds of linear thresholds (i.e. multilayer perceptrons with two hidden units). In addition, if the weights are restricted to 1 and 0 (but the threshold is arbitrary), then linear threshold concepts on Boolean instances spaces are not properly PAC learnable [101].

3. The classes of $k$-DNF, $k$-CNF, and $k$-decision lists are properly PAC learnable for each fixed $k$ [125, 111], but it is unknown whether the classes of all DNF functions, all CNF functions, or all decision trees are properly PAC learnable.

Most of the difficulties in proper PAC learning are due to the computational difficulty of finding a hypothesis in the particular form specified by the target class. For example, while Boolean threshold functions with 0-1 weights are not properly PAC learnable on Boolean instance spaces (unless $\mathbf{RP} = \mathbf{NP}$), they are PAC learnable by general Boolean threshold functions. Here we have a concrete case where enlarging the hypothesis space makes the computational problem of finding a good hypothesis easier. The class of all Boolean threshold functions is simply an easier space to search than the class of Boolean threshold functions with 0-1 weights. Similar extended hypothesis spaces can be found for the two classes mentioned in (1.) above that are not properly PAC learnable. Hence, it turns out that these classes are PAC learnable [101, 50]. However, it is not known if any of the classes of DNF functions, CNF functions, decision trees, or multilayer perceptrons with two hidden units are PAC learnable.

It is a much stronger result to show that a concept class is not PAC learnable than it is to show that it is not properly PAC learnable, since the former result implies that the concept class is not PAC learnable by any reasonable hypothesis space. Methods for determining the PAC learnability of concept classes have been developed by Pitt and Warmuth [102]. These methods involve reductions of one learning problem to another, and notions of *learning-completeness* for learning problems that are analogous to the corresponding notions in the theory of complexity classes and reducibilities for general computational problems (e.g. [118]).

Using this framework of Pitt and Warmuth, in conjunction with the results of Goldreich, Goldwasser, and Micali [45], it can be shown that certain learning problems are learning-complete for $\mathbf{P}$ (see [102, 81]), and that the concept classes associated with these problems are not PAC learnable (even in an extremely weak sense) assuming the existence of any cryptographically secure pseudo-random bit generator, which is equivalent to the existence of a certain type of one-way function [73]. While such an assumption is stronger than the assumption that $\mathbf{RP} \neq \mathbf{NP}$, there is still convincing evidence for its validity.

Simpler problems can also be shown not to be PAC learnable based on stronger cryptographic assumptions. In particular, Kearns and Valiant [65] show that a polynomial-time learning algorithm for Deterministic Finite Automata (DFAs) can be used to invert certain cryptographic functions. This is done by first showing that inverting these cryptographic functions reduces to learning arbitrary Boolean formulas (i.e. Boolean expressions using the operators "and", "or" and "not"). Then, since it can be shown that learning Boolean Formulas reduces to learning DFAs, it follows that DFAs are not polynomially learnable, based the assumption that these cryptographic functions are not efficiently invertible.

Kearns and Valiant also obtain some strong negative results for multilayer perceptrons. In particular, by the same methods, they show there exists some constant $d$ and polynomial $p(n)$ such that the class of concepts represented by feedforward neural networks with $n$ Boolean inputs and $d$ hidden layers with at most $p(n)$ hidden units in all, in which each hidden unit computes a linear threshold function, is not polynomially learnable based on the same cryptographic assumptions as above (Theorem 6 of [65]). The smallest value of the depth $d$ for which this result holds has not been determined. As mentioned above, it is possible that this strong result already holds for multilayer perceptrons with just two hidden units.

## Methods for Proving PAC Learnability; Formalization of Bias

All of the positive learnability results above are obtained by

1. showing that there is an efficient algorithm that finds a hypothesis in a particular hypothesis space that is consistent with a given sample of any concept in the target class and

2. that the sample complexity of any such algorithm is polynomial.

By *consistent* we mean that the hypothesis agrees with every example in the training sample. An algorithm that always finds such a hypothesis (when one exists) is called a *consistent algorithm.* It should be noted that this PAC usage of the term "consistent" has no relation to the notion of consistency for methods of parameter estimation, etc. in statistics.

As the size of the hypothesis space increases, it may become easier to find a consistent hypothesis, but it will require more random training examples to insure that this hypothesis is accurate with high probability. In the limit, when any subset of the instance space is allowed as a hypothesis, it becomes trivial to find a consistent hypothesis, but a sample size proportional to the size of the entire instance space will be required to insure that it is accurate. Hence, there is a fundamental tradeoff between the computational complexity and the sample complexity of learning.

Restriction to particular hypothesis spaces of limited size is one form of *bias* that has been explored to facilitate learning [85]. In addition to the cardinality of the hypothesis space, a parameter known as the Vapnik-Chervonenkis (VC) dimension of the hypothesis space has been shown to be useful in quantifying the bias inherent in a restricted hypothesis space [127, 99, 49]. The VC dimension of a hypothesis space $H$, denoted $dim_{VC}(H)$, is defined to be the maximum number $d$ of instances that can be labeled as positive and negative examples in all $2^d$ possible ways, such that each labeling is consistent with some hypothesis in $H$ [30, 127, 128]. Let $\mathbf{H} = \{H_n\}_{n\geq 1}$ be a hypothesis space and $\mathbf{C} = \{C_n\}_{n\geq 1}$ be a target class, where $C_n \subseteq H_n$ for $n \geq 1$. Then it can be shown [12] that any consistent algorithm for learning $\mathbf{C}$ by $\mathbf{H}$ will have sample complexity at most

$$\frac{1}{\epsilon(1-\sqrt{\epsilon})}\left(2dim_{VC}(H_n)\mathbf{ln}\frac{6}{\epsilon} + \mathbf{ln}\frac{2}{\delta}\right).$$

This improves on earlier bounds given in [24], but may still be a considerable overestimate.

In terms of the cardinality of $H_n$, denoted $|H_n|$, it can be shown [128, 89, 23] that the sample complexity is at most

$$\frac{1}{\epsilon}\left(\mathbf{ln}|H_n| + \mathbf{ln}\frac{1}{\delta}\right).$$

For most hypothesis spaces on Boolean domains, the second bound gives the better bound. In contrast, most hypothesis spaces on real-valued attributes are infinite, so only the first bound is applicable. Neural networks, and in particular linear threshold functions are a notable exception. The VC dimension of the class of linear threshold functions on $n$ Boolean inputs is $n+1$, while the logarithm of the cardinality of this class is quadratic in $n$ [24], so it is better to use the first bound in this case, even though the hypothesis space is finite. In general, the VC dimension of the class of functions represented by multilayer perceptrons with a fixed architecture but variable weights is upper bounded by a quantity that is close to (but slightly higher than) the number of variable weights, whether the inputs are real-valued or Boolean [31, 17]. If the weights are represented with very high or infinite precision, then it may be better to use these estimates of the VC dimension in conjunction with the first bound. However, when concepts are represented by neural networks of a fixed size with only a few bits of precision per weight, the hypothesis space is finite even for real-valued inputs, so the second bound is applicable, and it is often better. Unfortunately, in either case the bounds are likely to be overly pessimistic in practice.

Extensions of these results for the case of noisy training data and more complex learning problems are described in the second part of this chapter, and in the chapter by Vapnik.

## Criticisms of the PAC Model

The two criticisms most often leveled at the PAC model by AI researchers interested in empirical machine learning are

1. the worst-case emphasis in the model makes it unusable in practice (e.g. [26, 114]) and
2. the notions of target concepts and noise-free training data are unrealistic in practice (e.g. [4, 19]).

We take these in turn.

There are two aspects of the worst case nature of the PAC model that are at issue. One is the use of the worst case model to measure the computational complexity of the learning algorithm, the other is the definition of the sample complexity as the worst case number of random examples needed over all target concepts in the target class and all distributions on the instance space. Little work has been done on the former issue (with the notable exception of [131]), so here we address only the latter issue.

As pointed out in the section "Definition of PAC Learning" above, the worst case definition of sample complexity means that even if we could calculate the sample complexity of a given algorithm exactly, we would still expect it to overestimate the typical error of the hypothesis produced as a function of the training set size on any particular target concept and particular distribution on the instance space. This is compounded by the fact that we usually cannot calculate the sample complexity of a given algorithm exactly even when it is a relatively simple consistent algorithm. Instead we are forced to fall back on the upper bounds on the sample complexity that hold for any consistent algorithm, given in the previous section, which themselves may contain overblown constants.

The upshot of this is that the basic PAC theory is not good for predicting learning curves. Some variants of the PAC model come closer, however. One simple variant is to make it distribution specific, i.e. define and analyze the sample complexity of a learning algorithm for a specific distribution on the instance space, e.g. the uniform distribution on a Boolean space [18, 114]. There are two potential problems with this. The first is finding distributions that are both analyzable and indicative of the distributions that arise in practice. The second is that the bounds obtained may be very sensitive to the particular distribution analyzed, and not be very reliable if the actual distribution is slightly different.

A more refined, Bayesian extension of the PAC model is explored in [26]. Using the Bayesian approach involves assuming a prior distribution over possible target concepts as well as training instances. Given these distributions, the average error of the hypothesis as a function of training sample size, and even as a function of the particular training sample, can be defined. Also, $1-\delta$ confidence intervals like those in the PAC model can be defined as well. Experiments with this model on small learning problems are encouraging, but further work needs to be done on sensitivity analysis, and on simplifying the calculations so that larger problems can be analysed. Some success in tackling the difficult computations involved in certain Bayesian approaches to learning theory has been obtained by using the tools from statistical physics [32, 121, 47, 117, 97]. This work, and the other distribution specific learning work, provides an increasingly important counterpart to PAC theory[1].

Another variant of the PAC model designed to address these issues is the "probability of mistake" model explored in [57] [56] and [97]. This model is designed specifically to help understand some of the issues in incremental learning. Instead of looking at sample complexity as defined above, the measure of performance here is the probability that the learning algorithm incorrectly guesses the label of the $m^{\text{th}}$ training example in a sequence of $m$ random examples. Of course, the algorithm is allowed to update its hypothesis after each new training example is processed, so as $m$ grows, we expect the probability of a mistake on example $m$ to decrease. For a fixed target concept and a fixed distribution on the instance space, it is easy to see that the probability of a mistake on example $m$ is the same as the average error of the hypothesis produced by the algorithm from $m-1$ random training examples. Hence, the probability of mistake on example $m$ is exactly what is plotted on empirical learning curves that plot error versus sample size and average several runs of the learning algorithm for each sample size.

In [57] the focus is on the worst case probability of mistake on the $m^{\text{th}}$ example, over all possible target concepts and distributions on the training examples. In [56] and [97] the probability of mistake on the $m^{\text{th}}$ example is examined when the target concept is selected at random according to a prior distribution on the target class and the examples are drawn at random from a certain fixed distribution. This is a Bayesian approach. The former we will call the *worst case probability of mistake* and the latter we will call the *average case probability of mistake*. The results can be summarized as follows. Let $\mathbf{C} = \{C_n\}_{n\geq 1}$ be a concept class and $d_n = dim_{VC}(C_n)$ for all $n \geq 1$.

First, for any concept class $\mathbf{C}$ and any consistent algorithm for $\mathbf{C}$ using hypothesis space $\mathbf{C}$, the worst case probability of mistake on example $m$ is at most $O((d_n/m)\mathbf{ln}(m/d_n))$, where $m > d_n$. Furthermore, there are particular consistent algorithms and concept classes where the worst case probability of mistake on example $m$ is at least $\Omega((d_n/m)\mathbf{ln}(m/d_n))$, hence this is the best that can be said in general of arbitrary consistent algorithms.

Second, for any concept class $\mathbf{C}$ there exists a (universal) learning algorithm for $\mathbf{C}$ (not necessarily consistent or computationally efficient) with worst case probability of mistake on example $m$ at most $d_n/m$. On the other hand, any learning algorithm for $\mathbf{C}$ must have worst case probability of mistake on example $m$ at least $\Omega(d_n/m)$, so this universal algorithm is essentially optimal.

[1]There has been a recent surge of interest in Bayesian approaches to neural network learning; see e.g. [82, 27].

Third, if we focus on average case behavior, then there is a different universal learning algorithm, which is called *Bayes optimal learning algorithm* (or the *weighted majority algorithm* [80]) and there is a closely related, more efficient algorithm called the *Gibbs* (or *randomized weighted majority*) algorithm that have average case probability of mistake on example $m$ at most $d_n/m$ and $2d_n/m$, respectively. Furthermore, there are particular concept classes $\mathbf{C}$, particular prior probability distributions on the concepts in these classes, and particular distributions on the instance spaces of these classes, such that the average case probability of mistake on example $m$ is at least $\Omega(d_n/m)$ for any learning algorithm (with constant $\approx 1/2$). This indicates that the above general bounds are tight to within a small constant. Even better forms of these upper and lower bounds can be given for specific distributions on the examples, specific target concepts, and even specific sequences of examples.

These results show two interesting things. First, certain learning algorithms perform better than arbitrary consistent learning algorithms in the worst case and average case, therefore, even in this restricted setting there is definitely more to learning than just finding any consistent hypothesis in an appropriately biased hypothesis space. Second, the worst case is not always much worse than the average case. Some recent experiments in learning perceptrons and multilayer perceptrons have shown that in many cases $d_n/m$ is a rather good predictor of actual (i.e. average case) learning curves for backpropagation on synthetic random data [16, 120]. However, it is still often an overestimate on natural data [112], and in other domains such as learning conjunctive concepts on a uniform distribution [114]. Here the distribution (and algorithm) specific aspects of the learning situation must also be taken into account. Thus, in general we concur that extensions of the PAC model are required to explain learning curves that occur in practice. However, no amount of experimentation or distribution specific theory can replace the security provided by a distribution independent bound.

The second criticism of the PAC model is that the assumptions of well-defined target concepts and noise-free training data are unrealistic in practice. This is certainly true. However, it should be pointed out that the computational hardness results for learning described above, having been established for the simple noise-free case, must also hold for the more general case. The PAC model has the advantage of allowing us to state these negative results simply and in their strongest form. Nevertheless, the positive learnability results have to be strengthened before they can be applicable in practice, and some extensions of the PAC model are needed for this purpose. Many have been proposed (see e.g. [10, 62]).

Since the definitions of target concepts, random examples and hypothesis error in the PAC model are just simplified versions of standard definitions from statistical pattern recognition and decision theory, one reasonable thing to do is to go back to these well-established fields and use the more general definitions that they have developed. First, instead of using the probability of misclassification as the only measure of error, a general *loss function* can be defined that for every pair consisting of a guessed value and an actual value of the classification, gives a non-negative real number indicating a "cost" charged for that particular guess given that particular actual value. Then the error of a hypothesis can be replaced by the average loss of the hypothesis on a random example. If the loss is 1 if the guess is wrong and 0 if it is right (*discrete loss*), we get the PAC notion of error as a special case. However, using a more general loss function we can also choose to make false positives more expensive than false negatives or vice-versa, which can be useful. The use of a loss function also allows us to handle cases where there are more than two possible values of the classification. This includes the problem of learning real-valued functions, where we might choose to use $|guess - actual|$ or $(guess - actual)^2$ as loss functions.

Second, instead of assuming that the examples are generated by selecting a target concept and then generating random instances with labels agreeing with this target concept, we might assume that for each random instance, there is also some randomness in its label. Thus, each instance will have a particular probability of being drawn and, given that instance, each possible classification value will have a particular probability of occurring. This whole random process can be described as making independent random draws from a single joint probability distribution on the set of all possible labeled instances. Target concepts with attribute noise, classification noise, or both kinds of noise can be modeled in this way. The target concept, the noise, and the distribution on the instance space are all bundled into one joint probability measure on labeled examples. The goal of learning is then to find a hypothesis that minimizes the average loss when the examples are drawn at random according to this joint distribution.

The PAC model, disregarding computational complexity considerations, can be viewed as a special case of this set-up using the discrete loss function, but with the added twist that learning performance is measured with respect to the worst case over all joint distributions in which the entire probability measure is concentrated on a set of examples that are consistent with a single target concept of a particular type. Hence, in the PAC case it is possible to get arbitrarily close to zero loss by finding closer and closer approximations to this underlying target concept. This is not possible in the general case, but one can still ask how close the hypothesis produced by the learning algorithm comes to the performance of the best possible hypothesis in the hypothesis space. For an unbiased hypothesis space, the latter is known as Bayes optimal classifier [34].

Some recent PAC research has used this more general framework. By using the quadratic loss function mentioned above in place of the discrete loss, Kearns and Shapire investigate the problem of efficiently learning a real-valued regression function that gives the probability of a "+" classification for each instance [66]. A more general approach, which also links the learning model with the minimum description length framework of Rissanen [109], is given by Yamanishi in [140]. In the second part of this chapter we show how the VC dimension and related tools, originally developed by Vapnik, Chervonenkis, and others for this type of analysis, can be applied to the more general study of learning in neural networks. Here no restrictions whatsoever are placed on the joint probability distribution governing the generation of examples, i.e. the notion of a target concept or target class is eliminated entirely. Using this method, specific sample complexity bounds are obtained for learning with feedforward neural networks under various loss functions. Further results along these and related lines are given in the chapters by Vapnik and White.

Indeed, viewed in the larger context of statistics, the basic PAC model is really only a "toy" model; useful as an introduction to some of the important ideas in machine learning, and as a clean framework in which questions of computational complexity can be addressed (and answered negatively in their strongest form), but not nearly rich enough to capture the range of machine learning practice. Some other, more sophisticated union of statistics, computer science and possibly other disciplines will be required for this task. While no comprehensive model of this type has been suggested in the computational learning theory literature, a number of other aspects of, and approaches to machine learning have been explored. A few of these are discussed in the next section.

## Other Theoretical Learning Models

A number of other theoretical approaches to machine learning are flourishing in recent computational learning theory work. One of these is the *total mistake bound* model [76]. Here an arbitrary sequence of examples of an unknown target concept is fed to the learning algorithm, and after seeing each instance the algorithm must predict the label of that instance. This is an incremental learning model like the probability of mistake model described above, however here it is not assumed that the instances are drawn at random, and the measure of learning performance is the *total* number of mistakes in prediction in the worst case over all sequences of training examples (arbitrarily long) of all target concepts in the target class. We will call this latter quantity the *(worst case) mistake bound* of the learning algorithm. Of interest is the case when there exists a polynomial time learning algorithm for a concept class $\mathbf{C} = \{C_n\}_{n\geq 1}$ with a worst case mistake bound for target concepts in $C_n$ that is polynomial in $n$. As in the PAC model, mistake bounds can also be allowed to depend on the syntactic complexity of the target concept.

The perceptron algorithm for learning linear threshold functions in the Boolean domain is a good example of a learning algorithm with a worst case mistake bound. This bound comes directly from the bound on the number of updates given in the perceptron convergence theorem (see e.g. [34]). The worst case mistake bound of the perceptron algorithm is polynomial (and at least linear) in the number $n$ of Boolean attributes when the target concepts are conjunctions, disjunctions, or any concept expressible with 0-1 weights and an arbitrary threshold [48]. A variant of the perceptron learning algorithm with multiplicative instead of additive weight updates was developed that has a significantly improved mistake bound for target concepts with small syntactic complexity [76]. The performance of this algorithm has also been extensively analysed in the case when some of the examples may be mislabeled [78].

It can be shown that if there is a polynomial time learning algorithm for a target class $\mathbf{C}$ with a polynomial worst case mistake bound, then $\mathbf{C}$ is PAC learnable. General methods for converting a learning algorithm with a good worst case mistake bound into a PAC learning algorithm with a low sample complexity are given in [77]. Hence, the total mistake bound model is actually not unrelated to the PAC model.

Another fascinating transformation of learning algorithms is given by the *weighted majority method* [80]. (See also the general methods in [132].) This is a method of combining several incremental learning algorithms into a single incremental learning algorithm that is more powerful and more robust than any of the component algorithms. This method extends the Bayesian-style weighted majority algorithm mentioned in the previous section. The idea is simple. All the component learning algorithms are run in parallel on the same sequence of training examples. For each example, each algorithm makes a prediction and these predictions are combined by a weighted voting scheme to determine the overall prediction of the "master" algorithm. After receiving feedback on its prediction, the master algorithm adjusts the voting weights for each of the component algorithms, increasing the weights of those that made the correct prediction, and decreasing the weights of those that guessed wrong, in each case by a multiplicative factor. It can be shown that this method of combining learning algorithms is very robust with regard to mislabeled examples. More importantly, the method produces a master algorithm with a worst case mistake bound that approaches the worst case mistake bound of the best component learning algorithm [80]. Thus the performance of the master algorithm is almost as good as that of the best component algorithm. This is particularly useful when a good learning algorithm is known but a parameter of the algorithm has to be tuned for the particular application [61]. In this case the weighted majority method is applied to a pool of component algorithms, each of which is a version of the original learning algorithm with a different setting of the parameter. The master algorithm's performance approaches the performance of the component algorithm with the best setting of the parameter.

The weighted majority method can also be adapted to the case when the predictions of the component algorithms are continuous [80]. This leads to a method for designing a master algorithm whose worst case loss approaches the worst case loss of the best linear combination of the component learning algorithms [79]. Here instead of the total number of mistakes, the loss is the total squared prediction error. Finally, a version of the weighted majority method can also be used to obtain good mistake bounds in the case when the best component algorithm changes in various sections of the trial sequence. More general learning problems for "drifting" target concepts have been investigated as well [60, 69]. This represents an interesting new direction in computational learning theory research.

Both the PAC and total mistake bound models can be extended significantly by allowing learning algorithms to perform experiments or make queries to a teacher during learning [6]. The simplest type of query is a *membership query*, in which the learning algorithm proposes an instance in the instance space and then is told whether or not this instance is a member of the target concept. The ability to make membership queries can greatly enhance the ability of an algorithm to efficiently learn the target concept in both the mistake bound and PAC models. It has been shown that there are polynomial time algorithms that make polynomially many membership queries and have polynomial worst case mistake bounds for learning

1. monotone DNF concepts (Disjunctive Normal Form with no negated variables) [6],

2. $\mu$-formulae (Boolean formulae in which each variable appears at most once) [8],

3. deterministic finite automata [5], and

4. Horn sentences (propositional PROLOG programs) [7].

In addition, there is a general method for converting an efficient learning algorithm that makes membership queries and has a polynomial worst case mistake bound into a PAC learning algorithm, as long as the PAC algorithm is also allowed to make membership queries. Hence, all of the concept classes listed above are PAC learnable when membership queries are allowed. This contrasts with the evidence from cryptographic assumptions that classes (2) and (3) above are not PAC learnable from random examples alone [65].

Surprisingly, it can be shown, based on cryptographic assumptions, that slightly richer classes than those listed above list are not PAC learnable even with membership queries [9]. These include:

1. non-deterministic finite automata and

2. intersections of deterministic finite automata.

This is shown by generalizing the notion of polynomial-time learning-preserving reduction from [102] to the case when membership queries are allowed, and then reducing known cryptographically secure problems to the above learning problems.

## Conclusion

In this brief survey we were able to cover only a small fraction of the results that have been obtained recently in computational learning theory. For a glimpse at some of these further results we refer the reader to [58, 110, 44, 126]. However, we hope that we have at least convinced the reader that the insights provided by this line of investigation, such as those about the difficulty of searching hypothesis spaces, the notion of bias and its effect on required training size, the effectiveness of majority voting methods, and the usefulness of actively making queries during learning, have made this effort worthwhile.

## Part 2: Decision Theoretic Generalizations of the PAC Model for Neural Net Applications

Summary of Part 2: We describe a generalization of the PAC learning model that is based on statistical decision theory. In this model the learner receives randomly drawn examples, each example consisting of an instance $x \in X$ and an outcome $y \in Y$, and tries to find a decision rule $h : X \to A$, where $h \in \mathcal{H}$, that specifies the appropriate action $a \in A$ to take for each instance $x$, in order to minimize the expectation of a loss $\mathbf{L}(y, a)$. Here $X$, $Y$, and $A$ are arbitrary sets, $\mathbf{L}$ is a real-valued function, and examples are generated according to an arbitrary joint distribution on $X \times Y$. Special cases include the problem of learning a function from $X$ into $Y$, the problem of learning the conditional probability distribution on $Y$ given $X$ (regression), and the problem of learning a distribution on $X$ (density estimation).

We give theorems on the uniform convergence of empirical loss estimates to true expected loss rates for certain decision rule spaces $\mathcal{H}$, and show how this implies learnability with bounded sample size, disregarding computational complexity. As an application, we give distribution-independent upper bounds on the sample size needed for learning with feedforward neural networks. Our theorems use a generalized notion of VC dimension that applies to classes of real-valued functions, adapted from Vapnik and Pollard's work, and a notion of *capacity* and *metric dimension* for classes of functions that map into a bounded metric space. A fuller version of this part of the chapter appears in [54].

# 1 Introduction

The introduction of the Probably Approximately Correct (PAC) model [124] [6] of learning from examples has done an admirable job of drawing together practitioners of machine learning with theoretically oriented computer scientists in the pursuit of a solid and useful mathematical foundation for applied machine learning work. These practitioners include both those in mainstream artificial intelligence and in neural net research. However, in attempting to address the issues that are relevant to this applied work in machine learning, a number of shortcomings of the model have cropped up repeatedly. Among these are the following:

1. The model is defined only for $\{0, 1\}$-valued functions. Practitioners would like to learn functions on an instance space $X$ that take values in an arbitrary set $Y$, e.g. multi-valued discrete functions, real-valued functions and vector-valued functions.

2. Some practitioners are wary of the assumption that the examples are generated from an underlying "target function", and are not satisfied with the noise models that have been proposed to weaken this assumption (e.g. [10] [116] [115]). They would like to see more general regression models investigated in which the $y$ component in a training example $(x, y) \in X \times Y$ is randomly specified according to a conditional distribution on $Y$, given $x$. Here the general goal is to approximate this conditional distribution for each instance $x \in X$. In the computational learning theory literature, a model of this type is investigated in [66], with $Y = \{0, 1\}$, and in a more general case in [139].

3. Many learning problems are unsupervised, i.e. the learner has access only to randomly drawn, unlabeled examples from an instance space $X$. Here learning can often be viewed as some form of approximation of the distribution that is generating these examples. This is usually called *density estimation* when the instance space $X$ is continuous and no specific parametric form for the underlying distribution on $X$ is assumed. It is often called *parameter estimation* when specific parametric probability models are used. One example of this in the computational learning theory literature is the recent investigation of Abe and Warmuth into the complexity of learning the parameters in a hidden Markov model [1].

Our purpose here is twofold. First, we propose an extension of the PAC model, based on the work of Vapnik and Chervonenkis [129] and Pollard [104, 106], that addresses these and other issues. Second, we use this extension to obtain distribution-independent upper bounds on the size of the training set needed for learning with various kinds of feedforward neural networks.

## 1.1 Overview of the proposed framework

To extend the PAC model, we propose a more general framework based on statistical decision theory (see e.g. Ferguson [43], Kiefer [67] or Berger [20]). In this general framework we assume the learner receives randomly drawn training examples, each example consisting of an instance $x \in X$ and an outcome $y \in Y$, where $X$ and $Y$ are arbitrary sets called *instance* and *outcome spaces*, respectively. These examples are generated according to a joint distribution on $X \times Y$, unknown to the learner. This distribution comes from a (known) class $\mathcal{P}$ of joint distributions on $X \times Y$, representing possible "states of nature." After training, the learner will receive further random examples drawn from this same joint distribution. For each example $(x, y)$, the learner will be shown only the instance $x$. Then he will be asked to choose an action $a$ from a set of possible actions $A$, called the *decision space*. Following this, the outcome $y$ will be revealed to the learner. In the case that we examine here, the outcome $y$ depends only on the instance $x$ and not on the action $a$ chosen by the learner. For each action $a$ and outcome $y$, the learner will suffer a loss, which is measured by a fixed real-valued *loss function* $\mathbf{L}$ on $Y \times A$. We assume that the loss function is known to the learner. The learner tries to choose his actions so as to minimize his loss.

Here we look at the case in which, based on the training examples, the learner develops a deterministic strategy that specifies what he believes is the appropriate action $a$ for each instance $x$ in $X$. He then uses this strategy on all future examples. Thus we look at "batch" learning rather than "incremental" or "on-line" learning [76]. The learner's strategy, which is a function from the instance space $X$ into the decision space $A$, will be called a *decision rule*. We assume that the decision rule is chosen from a fixed *decision rule space* $\mathcal{H}$ of functions from $X$ into $A$. For example, instances in $X$ may be encoded as inputs to a neural network, and outputs of the network may be interpreted as actions in $A$. In this case the network represents a decision rule, and the decision rule space $\mathcal{H}$ may be all functions represented by networks obtained by varying the parameters of a fixed underlying network. The goal of learning is to find a decision rule in $\mathcal{H}$ that minimizes the expected loss, when examples are drawn at random from the unknown joint distribution on $X \times Y$.

This learning framework can be applied in a variety of situations. We now give several illustrations. For further discussion, we refer the reader to the excellent surveys of White [138], Barron [13], Devroye [33], and Vapnik [129], to which we are greatly indebted. Further discussion also appears in the chapters of Vapnik and White. We also recommend the text by Kiefer [67] for a general introduction to statistical inference and decision theory.

### 1.1.1 Betting Example

For our first example, consider the problem of learning to maximize profit (or minimize loss!) at the horse races. Here an instance $x$ in $X$ is a race, an action $a$ in $A$ consists of placing or not placing a certain bet, and an outcome $y$ in $Y$ is determined by the winner and the second and third place finishers. The loss $\mathbf{L}(y, a)$ is the amount of money lost when bet $a$ is placed and the outcome of the race is $y$. A negative loss is interpreted as gain. The joint distribution on $X \times Y$ represents the probability of various races and outcomes. This joint distribution is unknown to the learner; he only has random examples $(x_1, y_1), \ldots, (x_m, y_m)$, each consisting of a race/outcome pair generated from this distribution. From these examples, the learner develops a deterministic betting strategy (decision rule). The best decision rule $h$ is one that specifies a bet $a$ for each race $x$ that minimizes the expectation of the loss $\mathbf{L}(y, a)$, when $y$ is chosen randomly from the unknown conditional distribution on $Y$ given $x$, which is determined by the underlying joint distribution on $X \times Y$. This (not necessarily unique) best decision rule minimizes the expected loss on a random example $(x, y)$. It is known as *Bayes optimal decision rule.* The learner tries to approximate Bayes optimal decision rule as best he can using decision rules from a given decision rule space $\mathcal{H}$ (e.g. decision rules that can be represented by a particular kind of neural network).

### 1.1.2 classification

As a second example, consider the problem of medical diagnosis. Here an instance $x$ is a vector of measurements from medical tests conducted on the patient, an action $a$ is a diagnosis of the patient's disease state, and an outcome $y$ may be defined as the actual disease state of the patient. Here $A = Y$, i.e. the possible diagnoses are the same as the possible disease states. To specify the loss function $\mathbf{L}$, we may stipulate that there is zero loss for the correct diagnosis $a = y$, but for each pair $(y, a)$ with diagnosis $a$ differing from disease state $y$ there is some positive real loss $\mathbf{L}(y, a)$, depending on the severity of the consequences of that particular misdiagnosis. Here a decision rule is a diagnostic method, and Bayes optimal decision rule is the one that minimizes the expected loss from misdiagnosis when examples $(x, y)$ of test results and associated disease states occur randomly according to some unknown "natural" joint distribution.

This medical diagnosis situation is a typical example of a *classification learning* problem in the field of pattern recognition (see e.g. [34]). The problem of learning a Boolean function from noise-free examples, as investigated in the PAC model, is a special case of classification learning. Here the outcome space $Y$ is $\{0, 1\}$ and only the instance $x$ in an example $(x, y)$ is drawn at random. The outcome $y$ is $f(x)$ for some unknown Boolean *target function* $f$, rather than being determined stochastically. As above, the decision space $A$ is the same as the outcome space $Y$, and the action $a$ can be interpreted as a prediction of the outcome $y$. Hence, a decision rule $h$ maps from the instance space $X$ into the outcome space $Y$, just as the target function does. In much of AI, and in PAC learning in particular, it is common to refer to $h$ as a *hypothesis* in this case, and to $\mathcal{H}$ as the *hypothesis space.*

This same setup, where the outcome $y$ is a function of the instance $x$, can be applied to any function learning problem by letting $X$ and $Y$ be arbitrary sets. In the general function learning problem, the loss function $\mathbf{L}(y, a)$ usually measures the distance between the prediction $a$ and the actual value $y$ in some metric. In the PAC model, $\mathbf{L}$ is the discrete metric: $\mathbf{L}(y, a) = 0$ if $a = y$, else $\mathbf{L}(y, a) = 1$. Thus the expected loss of the decision rule (or hypothesis) is just the probability that it predicts incorrectly, the usual PAC notion of the *error* of the hypothesis. In general, $Y$ may be a set of strings, graphs, real vectors, etc., in which case other distance metrics or more general kinds of loss functions may be more appropriate.

### 1.1.3 regression

The general problem of regression has a different character from that of classification learning, but can also be addressed in the decision theoretic learning framework. To illustrate this, as a third example consider a variant of the medical diagnosis situation in which the doctor provides an estimate of the probability that the patient has each of several diseases, rather than predicting that he has one specific disease or asserting that he is healthy. (Here we assume that the actual disease state includes at most one disease.) For example, the doctor may say "Given these test results $x$, I would say you have disease 1 with probability 55%, disease 2 with probability 5%, and no disease at all with probability 40%." Here the doctor is actually trying to estimate the conditional distribution on disease states $Y$ given the test results $x$. Her action $a$ entails providing a vector of parameters that determine that estimated distribution, e.g. $(0.55, 0.05, 0.4)$. The decision space $A$ is the set of all such parameter vectors.

Now let $Y$ be an arbitrary discrete outcome space. Keeping the instance $x$ fixed, for each parameter vector $a$ in $A$ and outcome $y$ in $Y$ let $\widehat{P}(y; a)$ denote the probability of outcome $y$ with respect to the distribution on $Y$ defined by the parameter vector $a$. Thus when we take action $a$ on instance $x$, we are asserting that, given the instance $x$, we estimate the conditional probability of outcome $y$ to be $\widehat{P}(y; a)$ for each outcome $y$ in $Y$. Let $P(y)$ denote the actual conditional probability of outcome $y$, given the instance $x$, with respect to the unknown joint distribution on $X \times Y$. (The distributions $P$ and $\widehat{P}$ can be replaced by densities when $Y$ is continuous.) Let us define[2] the loss function $\mathbf{L}$ by setting $\mathbf{L}(y, a) = -\mathbf{log}\widehat{P}(y; a)$. This is called the (negative) *log likelihood* loss function. If we define loss in this way, then the expected loss resulting from action $a$ has a natural information theoretic interpretation[3]: it is the *Kullback-Leibler divergence* [71] (or

[2]We assume $\widehat{P}(y; a) > 0$ for all $y$ in $Y$.

[3]The Kullback-Leibler divergence from $P$ to $\widehat{P}$, denoted $I(P||\widehat{P})$, is defined as $\sum_{y \in Y} P(y) \mathbf{log} \frac{P(y)}{\widehat{P}(y;a)}$ for countable

*information gain* [108]) from the actual conditional probability distribution $P$ to the estimated conditional distribution $\widehat{P}$, plus the entropy of $P$.

For a given $x$, the entropy of the true conditional distribution $P$ is a constant, independent of the action $a$. Thus choosing the action $a$ for each instance $x$ that minimizes the expected log likelihood loss is equivalent to choosing the action $a$ that gives the closest estimate $\widehat{P}$ to the true conditional distribution $P$ over possible outcomes in $Y$ as measured by the Kullback-Leibler divergence, given that instance $x$. It is well known that the Kullback-Leibler divergence is minimized when $\widehat{P} = P$. This is Bayes optimal decision rule in regression.

In the regression version of our medical diagnosis situation, the definition of the log likelihood loss function depends on the interpretation of the components of the parameter vector $a$. If there are $k$ possible diseases and the patient can have at most one of these, then we might have $k+1$ possible mutually exclusive disease states $y_1, \ldots, y_{k+1}$, where $y_{k+1}$ means healthy. Hence $Y = \{y_1, \ldots, y_{k+1}\}$. Then we might specify that an action $a$ takes the form

$$a = (a_1, \ldots, a_{k+1}),$$

where $a_i = \widehat{P}(y_i; a)$, the estimated probability of disease state $y_i$. Here the components of the vector $a$ must be positive and sum to one. In this case the log likelihood loss would be $\mathbf{L}(y_i, a) = -\mathbf{log} a_i = -\mathbf{log}\widehat{P}(y_i; a)$.

Often the constraints on the components of $a$ are a nuisance, so other interpretations of $a$ are used, e.g. that $a_i = \mathbf{log}\widehat{P}(y_i; a) - \mathbf{log}\widehat{P}(y_{k+1}; a)$ for each $i$, $1 \leq i \leq k+1$. In this case the $a_1, \ldots, a_k$ are arbitrary real numbers and $a_{k+1} = 0$, and hence can be ignored. Since $\widehat{P}(y_i; a) = e^{a_i} / \sum_{j=1}^{k+1} e^{a_j}$, the log likelihood loss is $\mathbf{L}(y_i, a) = -a_i + \mathbf{log} \sum_{j=1}^{k+1} e^{a_j} = -a_i + \mathbf{log}(1 + \sum_{j=1}^{k} e^{a_j})$. This is known as the *logistic loss* [84, 13]. A third interpretation would be to allow the possibility that the patient may have more than one disease, and assume, for the purposes of estimation, that diseases occur independently. Then the disease state $y$ might be defined as a binary vector of length $k$, where the $i^{th}$ bit $y_i$ is 1 if and only if the $i$th disease is present. Hence $Y = \{0,1\}^k$. Similarly, the vector $a$ would be a vector of independent probabilities $(a_1, \ldots, a_k)$, where $a_i$ is the estimated probability

---

$Y$. The entropy of $P$, denoted $H(P)$, is $-\sum_{y \in Y} P(y)\mathbf{log}P(y)$. Thus $I(P||\widehat{P}) + H(P) = -\sum_{y \in Y} P(y)\mathbf{log}\widehat{P}(y; a)$, which is the expectation of the (negative) log likelihood loss. Analogous results hold for densities when the relevant quantities are finite [71] (see the chapter by White).

of the patient having the $i$th disease. In this case

$$\hat{P}(y;a) = \prod_{i=1}^{k} a_i^{y_i}(1-a_i)^{(1-y_i)}$$

and the log likehood loss is

$$\mathbf{L}(y,a) = -\sum_{i=1}^{k}\left(y_i \mathbf{log} a_i + (1-y_i)\mathbf{log}(1-a_i)\right),$$

which we will call the *cross entropy loss.*

In the medical diagnosis example, the outcome space $Y$ is discrete. However, in most uses of regression $Y$ is real valued, e.g. the outcome $y$ is the measurement of some real valued quantity, and the instance $x$ represents the experimental conditions under which this quantity was measured. In this case regression is usually defined as estimating the conditional expectation of $Y$ given the instance $x$. Thus $A \subset \Re$, and the action $a \in A$ for a given instance $x$ consists of an estimate of the mean of the various outcomes $y$ that would typically be observed for that instance $x$. It is easy to show that by using the *quadratic* loss function $\mathbf{L}(y,a) = (a-y)^2$, the expected loss is minimized when $a$ is the true mean, and hence this version of regression also fits naturally[4] into the decision theoretic framework. An alternate approach is to use the $L_1$ loss function $\mathbf{L}(y,a) = |a-y|$, in which case the expected loss is minimized when $a$ is the median of the conditional distribution $Y$ given the instance $x$. (See e.g. [138], [51] and the chapters by White and Vapnik for further discussion.)

### 1.1.4 density and parameter estimation

Finally, the problems of parameter estimation and density estimation can also be viewed as special cases of this decision theoretic framework. For parameter estimation, note that when the instance

[4] In fact, the standard version of regression, defined as estimating the conditional mean of $Y$ given instance $x$ using the quadratic loss function, is actually a special case of the general version of regression defined above, where for continuous outcome spaces $Y$, the object is to estimate the parameters specifying the conditional density of $Y$ given instance $x$, using the log likelihood loss function. To see this, assume that we represent the conditional density on $Y$ with a Gaussian density $\hat{p}(y;\mu,\sigma) = (2\pi\sigma^2)^{-1/2}e^{-(\mu-y)^2/2\sigma^2}$, where $\mu$ is the mean and $\sigma^2$ the variance. Let the variance be fixed, independent of $x$, so that the estimate $\hat{p}(y;\mu,\sigma)$, of the conditional density on $Y$ given $x$ is completely determined by the mean $\mu$. Thus the decision space $A \subset \Re$, and each action $a$ in $A$ is interpreted as specifying the mean of a Gaussian density. Substituting $\mu = a$ and evaluating $-\mathbf{log}\hat{p}(y;\mu,\sigma)$, the log likelihood loss is seen to be $\mathbf{L}(y,a) = \frac{1}{2\sigma^2}(a-y)^2 + \frac{1}{2}\mathbf{log}(2\pi\sigma^2)$. For fixed variance $\sigma^2$, this is equivalent, for learning, to the quadratic loss $(a-y)^2$, since additive and multiplicative constants in the definition of $\mathbf{L}$ only rescale it without changing the value of $a$ that minimizes its expectation. A more general treatment of these ideas appears in the chapter of White. There he discusses how both $\mu$ and $\sigma^2$ can be estimated conditionally for the Gaussian density and other members of the exponential family of densities.

space $X$ has only one element then the particular instance $x$ can be ignored entirely. Thus the regression problem reduces to the problem of estimating the parameters of a single distribution on the outcome space $Y$ from a sample of random outcomes $y$ from $Y$, i.e. to the simpler problem of parameter estimation. Here the decision rule is not a function but merely a single vector of parameters, and the decision rule space $\mathcal{H}$ is the same as the decision space $A$.

For density estimation, we can consider the dual case in which the outcome space $Y$ has only one element, and hence can be ignored. Thus examples are unlabeled instances $x$ drawn randomly from some density $p(x)$ on $X$. Let the decision set $A$ be the positive real numbers and each decision rule $h$ in $\mathcal{H}$ be a density on $X$. Then, as above, information theoretic considerations suggest the the loss function $\mathbf{L}(y, a) = \mathbf{L}(a) = -\mathbf{log} a$. Again, as above, the expected loss of $h$ is minimized when $h$ is the true density $p$. Further, if $p$ is not a member of $\mathcal{H}$, then the best decision rule in $\mathcal{H}$, in terms of minimizing the expected loss, is the one with the smallest Kullback-Leibler divergence from the true density $p$ [71]. In the chapter of White (section 4.2 on unsupervised networks), a more detailed exposition of this approach is given, along with some indication of how neural networks can be used to represent densities. Here, perhaps the simplest example is the representation of a mixture of Gaussian densities by a neural network with one hidden layer, as described in [103, 95]. Many more methods of density estimation are discussed in Vapnik's work.

When the instance space $X$ is discrete, we are not estimating a density on $X$ but rather a probability distribution. The same ideas as above carry over, except that we let the decision space $A = (0, 1)$ and each decision rule $h$ in $\mathcal{H}$ represent a probability distribution on $X$. Here we can also use the same loss function, and it has the same properties.

These examples illustrate the diversity of the learning problems that can be cast in the proposed decision theoretic framework, even under the restrictive assumptions we make here, i.e. that the outcome $y$ does not depend on the action $a$, and that the learner always observes both the outcome and the loss. By weakening these assumptions, we can model other types of learning as well, including *associative reinforcement learning* [15, 46] and the theory of *learning automata* (with static environment) [87]. However, we will not pursue this here.

## 1.2 Summary and discussion of the results presented here

There are three major practical issues in this decision theoretic view of learning. The first is the number of random examples needed in order to be able to produce a good decision rule in the decision rule space $\mathcal{H}$, i.e. a decision rule whose expected loss is near the minimum of all decision rules in $\mathcal{H}$. If too few examples are used, we run into the problem of *overfitting*, where the decision rule produced performs well on the training data, but not on further random examples drawn from the same joint distribution that generated this training data. The second is the adequacy of the decision rule space $\mathcal{H}$. If $\mathcal{H}$ does not contain any decision rule with expected loss close to that of Bayes optimal decision rule for the particular joint distribution we are dealing with, then we can never hope to achieve near optimal performance using this decision rule space. Choosing the right decision rule space often requires considerable insight into the particular problem domain. Finally, the third practical problem is the computational complexity of the method we use to produce our decision rule from the training examples. This issue has been addressed extensively in the PAC literature, and is also addressed in [66, 1]. Of these three important issues, here we examine only the first. This issue is referred to as the problem of estimating the "sample complexity" of the learning problem in the PAC literature [40].

The number of random training examples needed to avoid overfitting depends critically on the nature of the decision rule space used. Different kinds of decision rule spaces are used in different areas of learning research, partly because different kinds of instance and outcome spaces are used. In pattern recognition and statistics, the instance space $X$ is usually a finite dimensional real vector space, i.e. each instance consists of a vector of real valued measurements of some attributes. In density estimation, a decision rule represents a density on $X$, and many choices are possible. One common choice is a mixture of Gaussian densities (e.g. [34][95]). In standard regression, the outcome and decision spaces $Y$ and $A$ are identical and real valued, and linear functions are most often used as decision rules. For more complex outcome spaces such as those in the medical diagnosis example given above, the decision rule space for regression is usually defined using a *generalized linear model* [84]. Similarly, in binary classification, where there are only two possible outcomes in $Y$ as in the PAC model, linear threshold functions are most often used as decision rules, and there are straightforward generalizations for the case of $k$-ary classification (see e.g. [34]). This "linear bias" in pattern recognition and statistics is in contrast to that in the PAC model and other AI areas, including work in neural networks, in which a rich variety of decision rule spaces are used (see e.g. [122, 123, 49, 50]). Our main goal here is to develop analytic tools to help understand the problem of overfitting in these more complex decision rule spaces.

In order to focus on the problem of overfitting, we take a simplified view of learning, in which the learner chooses a decision rule space $\mathcal{H}$, and then tries to find a decision rule in $\mathcal{H}$ with near minimal expected loss. To do this, the learner looks for a decision rule that minimizes the observed average loss on the training examples, which is called *empirical loss* or *empirical risk* (see chapter by Vapnik.) For example, in standard linear regression[5] the learning algorithm is the method of least squares, i.e. we find the linear function $h$ that minimizes the average of $\mathbf{L}(y, h(x)) = (h(x)-y)^2$ over all examples $(x, y)$ in our training set. It is well known that if we have too few training examples, then we tend to overfit them, and the function we find does not come close to minimizing the actual expected quadratic loss, which would be obtained by integrating over all possible (mostly unseen) examples with respect to the unknown joint distribution on them. This same situation occurs with all nontrivial decision rule spaces, including the nonlinear regression models defined by feedforward neural nets.

Using certain measures of the "dimension" or "capacity" of the decision rule space $\mathcal{H}$ and classes derived from $\mathcal{H}$ (see below), we obtain general upper bounds on the number of random training examples needed so that with high probability, any decision rule in $\mathcal{H}$ that has small empirical loss on the training examples will have small actual expected loss, i.e. we get uniform convergence results for empirical estimates like those in [128][36][104, 106]. We show how these give upper bounds on sufficient training sample size like those derived in [24] and elsewhere using the notion of the VC dimension, and generalize those results.

As an application, we give specific bounds on the number of training examples needed to avoid overfitting when learning with the decision rule space of feedforward neural nets [113], extending previous work in [17] and [137] (see also related work in [12]). These are the nets most widely used in current neural net learning research. Our model for feedforward neural nets is quite general in that it allows many types of units in the nets, including quasi-linear units [113], radial basis units [103], and product units [38].

In our general setting, successful learning means finding a decision rule with average loss close to minimal over all decision rules in the given decision rule space, rather than loss close to zero as in the PAC model. In addition to using an additive model as in [75], we also define "close to" using a measure of relative difference (the $d_\nu$ *metric*) similar to the standard multiplicative measure of approximation used in combinatorial optimization. This allows us to state the relevant uniform convergence bounds as generalized "Chernoff-style" [11] bounds, as in [105],[25] (chapter

[5]For general regression with the negative log likelihood loss function, the principle of minimizing empirical loss is the same as the principle of *maximum likelihood* [20, 67].

12), rather than "Hoeffding-style" bounds (as in Pollard's results [104]), giving better bounds on sufficient training sample size in some important cases. These two types of bounds are analogous to the two types of bounds that Vapnik gives his work in that one uses a measure of absolute difference and the other a measure of relative difference. However, both of our bounds are "two-sided", i.e. they bound deviations both above and below the mean.

We give these upper bounds on required sample size only to give some indication of the order-of-magnitude dependence of sample size on certain critical parameters of the learning problem, and to illustrate the theory. They are still too crude to be used directly in practice, e.g. as explicit formulae for choosing an appropriate sample size. Cross validation techniques, in which some of the training examples are held in reserve and used instead to test the performance of the decision rules produced by the learning algorithm, are likely to perform better for this task in practice (see e.g. [137, 134])[6]. Nevertheless, cross validation is only a means of estimating the amount of overfitting in the learning method in particular cases, i.e. it is only an engineering trick and provides no scientific explanation of the phenomenon. Our goal is to understand and explain overfitting in general decision rule spaces, from a scientific rather than an engineering viewpoint.

Finally, we should note that in practice, many learning algorithms do more than just search for a decision rule in a fixed decision rule space that minimizes empirical loss. For example, it is common to let the decision rule space depend on the number of training examples available, using richer and richer decision rule spaces as more examples become available (see e.g. [137, 24]). This can allow the learning algorithm to produce a sequence of decision rules with expected losses that approach the loss of Bayes optimal decision rule in the limit of infinite training sample size for a large class of possible joint distributions. The results given here can be used to estimate the appropriate rate at which the decision rule space should grow relative to the sample size to avoid overfitting. Other approaches, e.g. the method of *structural risk minimization* introduced by Vapnik [128], and the *Bayesian* [20, 82, 27] and *minimum description length* (MDL) approaches [14, 109], try to find a decision rule that minimizes some function of empirical loss and decision rule complexity. These can also achieve expected loss approaching that of Bayes optimal decision rule in the limit, and may be more effective in practice. Although uniform convergence results such as those we develop here are also used in the analysis of such methods [128] (and in the analysis

[6]In Vapnik's work, he argues that by reducing the constants on his bounds, criteria are obtained that can provide useful guidance for the practitioner in choosing sample or network size, either in place of, or in conjunction with cross validation. Recent empirical studies from his group at Bell Labs support this claim. Hence these types of bounds may find direct use in practice after all.

of cross-validation methods [94]), the full treatment of such approaches is beyond the scope of the present chapter. These issues are dealt with more fully in Vapnik's chapter. It should also be noted that Bayesian methods and structural risk minimization can be applied even when the decision rule space includes only neural networks of a fixed size. An example is the recent work using weight penalty functions in neural net training [133, 72, 96, 82, 27]. Such approaches may significantly reduce the training sample size needed to avoid overfitting in practice.

### 1.3 Overview of methods used

We now briefly discuss the methodology and previous work used in obtaining our results. Our work builds directly on the work of Vapnik and Chervonenkis, Pollard, and Dudley on the uniform convergence of empirical estimates [128][104][36] and its application to pattern recognition [128, 129] [33]. It also builds on the work of Benedek and Itai on PAC learnability with respect to specific probability distributions [18], and is related to the work of Natarajan and Tadepalli on extensions of the VC dimension to multi-valued functions [92] [91] and PAC learnability with respect to classes of probability distributions [88] [90]. In addition, Quiroz and Kulkarni have each independently generalized the PAC model in a related manner [107, 70].

One of the key ideas we use is the notion of an $\epsilon$-cover of a metric space [36] [104] [18] [90] [107] and the associated idea of *metric dimension* [68] (also called the *fractal* dimension [41]). This notion of dimension has played an important role in the now very active study of fractals in nature [83], especially in connection with chaos in dynamical systems [41][42]. Here we build further on the beautiful results of Vapnik and Chervonenkis [128], Dudley [35] and Pollard [104], which relate a type of generalized VC dimension for a decision rule space to the number of balls of radius $\epsilon$ required to cover the space, with respect to certain metrics. The sizes of the smallest such covers determine the *metric dimension* of the space. Our treatment closely parallels the approach given in [106]. It is interesting to note that related results connecting $\epsilon$-covers with the VC dimension have also been independently developed in [18] and in recent computational geometry work [135][7]. This work seems to lead to a potentially rich area of investigation that combines elements of combinatorics, topology and geometry, and probability and measure in a novel framework. We feel that this area is not only fascinating from a purely mathematical standpoint, but also potentially very useful in

[7] Specifically, Lemma 7.13 of [35] is nearly equivalent to Lemma 4.1 of [135] (using the primal space instead of the dual). This result also gives a stronger version of Theorem 4, part(3) of [18]. We give a still stronger version of this result in Theorem 10 below.

machine learning and other applied fields.

## 1.4 Organization of the results

The remainder of the chapter is organized as follows. The learning framework we have described above in section 1.1 is defined more formally in section 2. There we also look at the question of evaluating the performance of learning algorithms in terms of the number of training examples they use. This question is also formalized from a decision theory perspective. We then provide a lemma (Lemma 1) that can be used to evaluate the performance of learning algorithms that work by minimizing empirical loss. To use this lemma, we need bounds on the rate of uniform convergence of empirical loss estimates to true expected losses. These are given in section 3. The key bound is given in Theorem 2 in section 3, and in a more general version in Theorem 3.

To use the bound from Theorem 2 we need bounds on the "random covering numbers" associated with the decision rule space $\mathcal{H}$, the loss function $\mathbf{L}$ and the distribution $P$. These are related to the idea of an $\epsilon$-cover described above. Tools for bounding the random covering numbers in the worst case over all distributions $P$ are developed in section 4. Here we introduce the notion of the capacity of the decision rule space $\mathcal{H}$ (for a particular loss function $\mathbf{L}$), and the related notion of the metric dimension of $\mathcal{H}$. In section 5 we use these notions, along with Pollard's generalization of the VC dimension (the *pseudo dimension*), described in section 8.2 of the appendix, to obtain bounds on the performance (in terms of the number of training examples used) of learning algorithms that use multilayer feedforward neural networks, and work by minimizing empirical loss (Corollary 2). Finally, some further discussion of our results is given in the conclusion, section 6. Many of the more technical proofs and definitions have been omitted to simplify the presentation. They can be found in [54].

## 1.5 Notational conventions

We denote the real numbers by $\Re$ and the non-negative real numbers by $\Re^+$. By **log** and **ln** we denote the logarithm base 2 and the natural logarithm, respectively. We use $\mathbf{E}(\cdot)$ to denote the expectation of a random variable, and $\mathbf{Var}(\cdot)$ to denote the variance of a random variable. When the probability space is defined implicitly from the context, we use $\mathbf{Pr}(\cdot)$ to denote the probability of a set. However, usually the measure on the underlying probability space will be defined explicitly using the symbol $P$.

Here, $P$ will usually denote a probability measure on some appropriate[8] $\sigma$-algebra over the set $Z = X \times Y$, where $X$ is the instance space and $Y$ is the outcome space. We use $P^m$ to denote the $m$-fold product measure on $Z^m$. Functions on $Z$ and subsets of $Z$ mentioned in what follows will be assumed to be measurable without explicit reference. Alternately, we will also view $X$ and $Y$ as random variables on some other, unspecified, probability space, e.g. when they are viewed as real valued measurements. In this case $P$ is viewed as a joint distribution on $X$ and $Y$. In either case, the probability of a set $T \subset Z$ is defined by

$$P(T) = \int_T dP(z)$$

(where $z = (x, y)$ with $x \in X$ and $y \in Y$) and the expectation of function $f$ on $Z$ is denoted by

$$\mathbf{E}(f) = \int_Z f(z)dP(z).$$

When $Z$ is countable we will, with some abuse of notation, also use $P$ for the probability mass function, i.e. for $z \in Z$, $P(z)$ denotes $P(\{z\})$. Hence $P(T) = \sum_{z \in T} P(z)$ and $\mathbf{E}(f) = \sum_{z \in Z} f(z)P(z)$ in this case. When $Z$ is continuous, a density associated with $P$ (if it exists) is denoted by $p$.

When $Z$ is countable we use $P(y|x)$ to denote the probability that $Y = y$ given that $X = x$ (viewing $X$ and $Y$ as random variables) and similarly for $P(x|y)$. Hence $P(\cdot|x)$ denotes the conditional distribution on $Y$, given $X = x$. The marginal distribution in $X$ is defined by[9] $P_{|X}(x) = \sum_{y \in Y} P(x, y)$. Here and elsewhere, we abbreviate $P((x, y))$ by $P(x, y)$.

Finally, we list some other notation that is used several places in the text, indicating which section it is defined in.

---

[8] If $Z$ is countable then we assume this $\sigma$-algebra contains all subsets of $Z$, otherwise we assume that $Z$ is a complete, separable metric space (see section 8.1) and that this $\sigma$-algebra is the smallest $\sigma$-algebra that contains the open sets of $Z$ (i.e. the $\sigma$-algebra of Borel sets).

[9] When $Z$ is uncountable, the marginal and conditional distributions are defined so that

$$\int_Z \mathbf{f}(x, y)dP(x, y) = \int_X \left( \int_Y \mathbf{f}(x, y)dP(y|x) \right) dP_{|X}(x)$$

for every bounded measurable function $f$.

| | |
|---|---|
| $X$,$Y$,$A$,$\mathcal{H}$ and $\mathbf{L}$ | sections 1.1 and 2.1 |
| $\mathcal{P}$ | section 2.1 |
| $\mathbf{r}_{h,\mathbf{L}}(P)$, $\mathbf{r}_h(P)$ (true risk) | section 2.2 |
| $\mathbf{r}^*_{\mathbf{L}}(P)$, $\mathbf{r}^*(P)$ (optimal risk) | section 2.2 |
| $\widehat{\mathbf{r}}_h(\vec{z})$ (empirical risk) | section 2.2 |
| $\widehat{\mathbf{r}}^*(\vec{z})$ (optimal empirical risk) | section 2.2 |
| $d_\nu$ | section 2.2 |
| $L$, $L_\epsilon$, $L_{\alpha,\nu}$ (regret functions) | section 2.3 |
| $R$ (big 'L' risk) | section 2.3 |
| $m(\epsilon, \delta)$, $m(\alpha, \nu, \delta)$ (sample complexity) | section 2.4 |
| $\mathcal{N}$ (covering number) | sections 8.1 and 3.2 |
| $\mathcal{M}$ (packing number) | section 8.1 |
| $\mathbf{dim}$ (metric dimension) | section 8.1 |
| $\mathbf{dim_C}$ (pseudo dimension) | section 8.2 |
| $\mathcal{C}$ (capacity) | section 4 |
| $\rho_{\mathbf{L}}$ | section 4 |
| $l_{\mathcal{H}}$ | section 3 |
| $\mathbf{F}_{\vert_{\vec{z}}}$ | section 3 |
| $\widehat{\mathbf{E}}$ (empirical expectation) | section 3 |
| $d_{L^1}$ ($L^1$ distance for vectors) | section 3.2 |
| $d_{L^1(P)}$ ($L^1$ distance for functions) | section 8.2 |
| $d_{L^1(P,\rho)}$ ($L^1$ distance for functions) | section 4 |

## 2 Learning and optimization

We now further formalize the basic problem of learning, as introduced in section 1.1. We will introduce a formal notion of a learning algorithm, and a higher level loss function, which we will call a *regret function*, that measures how well the learning algorithm performs. The regret function will be defined in terms of the low level loss function $\mathbf{L}$ discussed in the previous section. Finally, we will show how an algorithm can solve the learning problem by solving a related optimization problem.

## 2.1 The basic components $X, Y, A, \mathcal{H}, \mathcal{P}$ and $\mathbf{L}$

We first review and further formalize the six components of the basic learning problem introduced in the previous section: $X$, $Y$, $A$, $\mathcal{H}$, $\mathcal{P}$ and $\mathbf{L}$. The first four components are the instance, outcome, decision and decision rule spaces, respectively. The first three of these are arbitrary sets, and the fourth, $\mathcal{H}$ is a family of functions from $X$ into $A$. These have been discussed extensively in the previous section.

The fifth component, $\mathcal{P}$, is a family of joint probability distributions on $X \times Y$. These represent the possible "states of nature" that might be governing the generation of examples. The set $Z = X \times Y$ will be called the *sample space*. We assume that examples are drawn independently at random according to some probability distribution $P \in \mathcal{P}$ on the sample space $Z$. A sequence of examples will be called a *sample*. In what follows[10] we will usually assume that $\mathcal{P}$ includes all probability distributions on $Z$. Hence our results will be distribution independent.

The last component, the loss function $\mathbf{L}$, is a mapping from $Y \times A$ into $\Re$. In this paper we will assume that $\mathbf{L}$ is bounded and nonnegative, i.e. $0 \leq \mathbf{L} \leq M$ for some real $M$. When $Y$ and $A$ are finite it is always possible to enforce this condition by simply adding a constant to $\mathbf{L}$, which doesn't change the learning problem in any essential way. When either $Y$ or $A$ is infinite, the learning problem sometimes needs to be restricted to meet this condition. For example, in regression[11] we might restrict the possible parameter vectors in $A$ and/or the possible outcomes in $Y$ such that for every $y \in Y$ and $a \in A$, $\widehat{P}(y;a) \geq b$ for some constant $b$. We can then take $M = -\mathbf{log}b$. In density estimation, the same thing can be accomplished by restricting the instance space $X$ to a bounded subset of $\Re^n$ on which all densities in $\mathcal{H}$ have values uniformly greater than $b$ and less than $B$ for constants $0 < b < B$. We can then add $\mathbf{log}B$ to the loss function to make it positive. The same method works for estimating distributions on discrete spaces: we restrict ourselves to a finite instance space $X$ and demand that for all $x \in X$ and all probability distributions $h \in \mathcal{H}$, $h(x) \geq b > 0$ (see e.g. [1, 139]). These restrictions are often reasonable in practice, e.g. most

[10] It is, however, possible and in fact common to assume that $\mathcal{P}$ is a very specific class of probability distributions on $Z$. For example, let $X = \Re^n$. Then if we are doing classification learning and $Y$ is discrete we may assume that $y$ is selected according to an arbitrary distribution on $Y$, and for each $y$, $P(x|y)$ is a multi-variate Gaussian distribution on $X$ [34]. On the other hand, if we are doing linear regression, then $Y$ is real-valued and we might assume that $x$ is selected according to an arbitrary distribution on $X$, and $y$ is a linear function of $x$ with additive Gaussian noise. In PAC learning theory we have a discrete analog of the latter case. Here we usually have $X = \{0,1\}^n$, $Y = \{0,1\}$, and $y$ a Boolean function of $x$ of a particular type (e.g. defined by a small disjunctive normal form formula), possibly plus random noise.

[11] Note that to get bounded loss in linear regression, $X$ must a bounded subset of $\Re^n$ as well, since we can't bound $Y$ without bounding $X$. The coefficients of the functions in $\mathcal{H}$ must also be bounded.

measurements naturally have bounded ranges, but they can be annoying (see [129], [104], [106] for alternative approaches for unbounded loss functions).

## 2.2 Measuring distance from optimality with the $d_\nu$ metric

For a given decision rule $h \in \mathcal{H}$ and distribution $P$ on the sample space $Z$, the expected loss of $h$ is the average value of $\mathbf{L}(y, h(x))$, when the example $(x, y)$ is drawn at random according to $P$. It is defined by

$$\mathbf{r}_{h,\mathbf{L}}(P) = \mathbf{r}_h(P) = \mathbf{E}(\mathbf{L}(y, h(x))) = \int_Z \mathbf{L}(y, h(x)) dP(x, y)$$

(the subscript $\mathbf{L}$ will be omitted when the loss function is clear from the context.) Since $\mathbf{L}$ is bounded this expectation is finite for every distribution $P$. In decision theory the expected loss $\mathbf{r}_h(P)$ is called the *risk* of $h$ when $P$ is the true underlying distribution. This quantity generalizes the notion of the *error* of $h$ used in computational learning theory.

In section 1.1 we stated the goal of learning quite informally: Given examples chosen independently at random from some unknown probability distribution $P \in \mathcal{P}$, find a decision rule $\hat{h}$ in $\mathcal{H}$ that comes "close to" minimizing the risk $\mathbf{r}_h(P)$ over all $h \in \mathcal{H}$. Let $\mathbf{r}^*_{\mathbf{L}}(P)$ (or $\mathbf{r}^*(P)$ when $\mathbf{L}$ is clear from the context) denote the infimum of $\mathbf{r}_h(P)$ over all $h$ in the decision rule space $\mathcal{H}$. To formalize our notion of a basic learning problem, we first need to say what we mean that $\mathbf{r}_{\hat{h}}(P)$ is "close to" $\mathbf{r}^*(P)$.

Let $r = \mathbf{r}_{\hat{h}}(P)$ and $s = \mathbf{r}^*(P)$. One natural interpretation is to demand that $|r - s| \le \epsilon$ for some small $\epsilon > 0$. However, we will see in section 3.1 that sometimes it is better to use a relative measure of distance. For any real $\nu > 0$, let $d_\nu$ be the function defined by

$$d_\nu(r, s) = \frac{|r - s|}{\nu + r + s}$$

for any non-negative reals $r$ and $s$. It is straightforward but tedious to verify that $d_\nu$ is a metric on $\Re^+$. The $d_\nu$ metric is similar to the standard function

$$\frac{|r - s|}{s}$$

used to measure the difference between the quality $r$ of a given solution and the quality $s$ of an optimal solution in combinatorial optimization. However, our measure has been modified to be well-behaved when one or both of its arguments are zero, and to be symmetric in its arguments (so that it is a metric). Three other properties of $d_\nu$ are also useful.

1. For all non-negative reals $r$ and $s$, $0 \le d_\nu(r,s) < 1$.

2. For all non-negative $r \le s \le t$, $d_\nu(r,s) \le d_\nu(r,t)$ and $d_\nu(s,t) \le d_\nu(r,t)$.

3. For $0 \le r, s \le M$, $\frac{|r-s|}{\nu+2M} \le d_\nu(r,s) \le \frac{|r-s|}{\nu}$.

We will refer to the second property by saying that $d_\nu$ is *compatible with the ordering on the reals.*

## 2.3 The regret function $L$ and the big 'L' risk $R$

Once we have specified how we measure closeness to optimality, we still need to specify our criteria for a successful learning algorithm. Do we need to have the risk of the decision rule found close to the optimum $\mathbf{r}^*_{\mathbf{L}}(P)$ with high probability, or should its average distance from $\mathbf{r}^*_{\mathbf{L}}(P)$ be small? Do we measure success in terms of the performance of the algorithm on the worst case distribution in $\mathcal{P}$, or do we use some average case analysis over distributions in $\mathcal{P}$? These questions lead us right back to decision theory again, but this time at a higher level in the analysis of learning.

To see this, consider the structure of a learning algorithm $\mathcal{L}$. For any sample size $m$, the algorithm $\mathcal{L}$ may be given a sample $\bar{z} = ((x_1,y_1),\ldots,(x_m,y_m))$ drawn at random from $Z^m$ according an unknown product distribution $P^m$, where $P \in \mathcal{P}$. For any such $\bar{z}$ it will choose a decision rule $\mathcal{L}(\bar{z}) \in \mathcal{H}$. Thus abstractly, the algorithm defines a function $\mathcal{L}$ from the set of all samples over $Z$ into $\mathcal{H}$, i.e. $\mathcal{L} : \bigcup_{m\ge 1} Z^m \to \mathcal{H}$. Since we are not requiring computability here, we will call such $\mathcal{L}$ a *learning method.* When $P \in \mathcal{P}$ is the actual "state of nature" governing the generation of examples, and the algorithm produces the decision rule $h \in \mathcal{H}$, let us say that we suffer a nonnegative real-valued *regret* $L(P,h)$. Thus, formally $L : \mathcal{P} \times \mathcal{H} \to \Re^+$. In our treatment here, the regret function $L$ will be derived from the loss function $\mathbf{L}$, and will measure the extent to which we have failed to produce a near optimal decision rule, assuming $P$ is the true state of nature (i.e. the amount of "regret" we feel for not having produced the optimal decision rule). Finally, for each possible state of nature $P$, the average regret suffered by the algorithm, over all possible training samples $\bar{z} \in Z^m$, is the *big 'L' risk* of that algorithm under $P$ for sample size $m$. This big 'L' risk is defined formally by

$$R_{L,\mathcal{L},m}(P) = \int_{\bar{z}\in Z^m} L(P,\mathcal{L}(\bar{z}))dP^m(\bar{z}).$$

The goal of learning is to minimize big 'L' risk.

We illustrate these definitions with a few examples. First suppose we want to capture the notion of successful learning that is used in the PAC model. Then one possibility is to introduce an *accuracy parameter* $\epsilon > 0$ and define the regret function $L = L_\epsilon$ by letting $L_\epsilon(P,h) = 1$ if $\mathbf{r}_{h,\mathbf{L}}(P) - \mathbf{r}^*_{\mathbf{L}}(P) > \epsilon$, and $L_\epsilon(P,h) = 0$ otherwise. This we suffer regret only when the decision rule $h$ produced by the learning algorithm has risk that is more than $\epsilon$ from optimal, measured by the absolute difference metric. For this definition of regret, the big 'L' risk $R_{L_\epsilon,\mathcal{L},m}(P)$ measures the probability that the decision rule produced by $\mathcal{L}$ has risk more than $\epsilon$ from optimal, when $\mathcal{L}$ is given $m$ random training examples drawn according to $P$. We then demand that this big 'L' risk be small, i.e. smaller than some given *confidence parameter* $\delta > 0$.

In the PAC model it is commonly assumed that the examples given to the algorithm $\mathcal{L}$ are noise-free examples of some underlying target function $f \in \mathcal{H}$. In this case the risk $\mathbf{r}^*(P)$ of the optimal decision rule in $\mathcal{H}$ is zero, and hence $L_\epsilon(P,h) = 1 \Leftrightarrow \mathbf{r}_{h,\mathbf{L}}(P) > \epsilon$. Hence demanding big 'L' risk at most $\delta$ gives the usual PAC criterion that the risk of the decision rule (or "hypothesis") produced by $\mathcal{L}$ be greater than $\epsilon$ with probability at most $\delta$.

The regret function $L$ can also be defined similarly, but using the $d_\nu$ metric to measure distance from optimality, instead of the absolute difference. Specifically, for every $\nu > 0$ and $0 < \alpha < 1$, we can define the regret function $L_{\alpha,\nu}$ by letting $L_{\alpha,\nu}(P,h) = 1$ if $d_\nu(\mathbf{r}_{h,\mathbf{L}}(P), \mathbf{r}^*_{\mathbf{L}}(P)) > \alpha$, and $L_{\alpha,\nu}(P,h) = 0$ otherwise. In this case the big 'L' risk $R_{L_{\alpha,\nu},\mathcal{L},m}(P)$ measures the probability that the risk of the decision rule produced by the algorithm $\mathcal{L}$ has distance more than $\alpha$ from optimal in the $d_\nu$ metric, when the algorithm is given $m$ random training examples drawn according to $P$. We will see in sections 2.4 and 3.1 why this sometimes gives a more useful and flexible definition of regret. In fact, in this paper, we will give our main results in terms of the family $\{L_{\alpha,\nu} : \nu > 0 \textit{ and } 0 < \alpha < 1\}$ of regret functions, and show how corresponding results may be derived as corollaries for the family $\{L_\epsilon : \epsilon > 0\}$ of regret functions.

Other regret functions are also possible and lead to different learning criteria. For example, another, perhaps simpler, way to define regret is to let $L(P,h) = \mathbf{r}_{h,\mathbf{L}}(P) - \mathbf{r}^*_{\mathbf{L}}(P)$. When $\mathbf{r}^*_{\mathbf{L}}(P) = 0$, as it does in the standard noise-free PAC model, this definition makes the regret $L$ equal to the risk $\mathbf{r}_{\mathbf{L}}(P)$, i.e. the expectation of the underlying loss $\mathbf{L}$. In this case the big 'L' risk $R_{L,\mathcal{L},m}(P)$ measures the expectation of the loss incurred by the learning algorithm $\mathcal{L}$ when it is given $m$ random training examples drawn according to $P$, forms a decision rule $h$, and then uses $h$ to determine the action on one further independent random example drawn according to $P$. This gives a generalization of the learning criterion studied in [57]. When $\mathbf{r}^*_{\mathbf{L}}(P) \neq 0$, then the big 'L' risk gives the expectation of the amount of such loss above and beyond the expected loss that would be suffered if the optimal decision rule were used. In particular, in density estimation, where $P$ and $h$ are both densities on the instance space $X$, if $P \in \mathcal{H}$ then defining the regret by $L(P, h) = \mathbf{r}_{h,\mathbf{L}}(P) - \mathbf{r}^*_{\mathbf{L}}(P)$ makes it equal to the Kullback-Leibler divergence from $P$ to $h$. Hence the big 'L' risk is the expected Kullback-Leibler divergence of the decision rule $h$ returned by the algorithm from the true density (see sections 1.1.3 and 1.1.4).

It is also possible to define the regret function $L$ directly, without using an underlying loss function $\mathbf{L}$. For example, in density estimation it is possible to use other measures of the distance between two densities, e.g. the Hellinger distance or the total variational distance, as in [14, 139]. The criterion from [66] for inferring a good *model of probability* can also be defined using an appropriate regret function, without defining an underlying loss $\mathbf{L}$.

## 2.4 Full formalization of the basic learning problem

Having defined the regret function, and thereby the big 'L' risk function, we still face one last issue: do we want to minimize big 'L' risk in the worst case over all possible states of nature $P$ in $\mathcal{P}$, or do we want to assume a *prior distribution* on possible distributions in $\mathcal{P}$, so that we can define a notion of "average case" big 'L' risk to be minimized. The former goal is know as *minimax optimality*, and has been used in the PAC model. The later is the *Bayesian* notion of optimality [20, 67], and has been used in several approaches to learning in neural nets based on statistical mechanics [32, 121, 47, 117, 97, 98]. Unfortunately this last question has no clear cut answer, and leads us directly into a longstanding unresolved debate in statistics (see e.g. [74] and following discussion.). Since we have set out to generalize the PAC model, and since our results are best illustrated in the minimax setting, we will formalize the notion of a basic learning problem using the minimax criterion. In subsequent work we hope to further explore this Bayesian setting. (For recent work in Bayesian approaches to neural network learning see [82, 27], and for Bayesian versions of the PAC model see [56, 26].)

We can now define exactly what we mean by a basic learning problem, and what it means for a learning method to solve this problem in this minimax setting.

**Definition 1** *A basic learning problem is defined by six components $X$, $Y$, $A$, $\mathcal{H}$, $\mathcal{P}$, and $\mathcal{L}$, where the first five components are as defined in section 2.1, and the last component, $\mathcal{L}$, is a family of regret* *functions as defined in section 2.3 (e.g. $\mathcal{L} = \{L_{\alpha,\nu} : \nu > 0$ and $0 < \alpha < 1\}$, or $\mathcal{L} = \{L_\epsilon : \epsilon > 0\}$ for some loss function $\mathbf{L}$). Let $\mathcal{L}$ be a learning method as defined in section 2.3. We say that $\mathcal{L}$ solves the basic learning problem if for all $L \in \mathcal{L}$ and all $0 < \delta < 1$ there exists a finite sample size $m = m(L, \delta)$ such that*

$$\text{for all } P \in \mathcal{P},\ \ R_{L,\mathcal{L},m}(P) \leq \delta.$$

*The* sample complexity *of the learning method $\mathcal{L}$ is the smallest such integer valued function $m(L, \delta)$. When $\mathcal{L} = \{L_{\alpha,\nu} : \nu > 0$ and $0 < \alpha < 1\}$ we will denote $m(L_{\alpha,\nu}, \delta)$ by $m(\alpha, \nu, \delta)$ and when $\mathcal{L} = \{L_\epsilon : \epsilon > 0\}$ we will denote $m(L_\epsilon, \delta)$ by $m(\epsilon, \delta)$.*

As discussed above, this definition generalizes the PAC criterion, and several others as well. In fact, this definition is quite generous, in that sample size needed to get the big 'L' risk less than $\delta$ is only required to be finite for each $\delta > 0$. In particular, using property (3) of the $d_\nu$ metric from section 2.2, when the underlying loss function $\mathbf{L}$ is bounded, as we assume here, any algorithm $\mathcal{L}$ solves the basic learning problem using the $L_{\alpha,\nu}$ class of regret functions if and only if it solves it using the $L_\epsilon$ class. Thus it doesn't matter which of these two classes of regret functions we use. However, in practice it is the sample complexity of $\mathcal{L}$ that is critical, and this will depend on which class of regret functions are used.

The nature of this dependence is seen more clearly when we expand the condition

$$R_{L,\mathcal{L},m}(P) \leq \delta$$

for $L = L_\epsilon$ and $L = L_{\alpha,\nu}$. When $L = L_\epsilon$, this condition means that given $m$ random training examples drawn according to $P$, with probability at least $1 - \delta$, the decision rule $\hat{h}$ produced by the algorithm $\mathcal{L}$ satisfies

$$\mathbf{r}_{\hat{h},\mathbf{L}}(P) \leq \mathbf{r}^*_{\mathbf{L}}(P) + \epsilon,$$

i.e. the risk of $\hat{h}$ is at most $\epsilon$ greater than that of the optimal decision rule in $\mathcal{H}$. When $L = L_{\alpha,\nu}$, this condition is the same, except that we require

$$\mathbf{r}_{\hat{h},\mathbf{L}}(P) \leq \frac{1+\alpha}{1-\alpha}\mathbf{r}^*_{\mathbf{L}}(P) + \frac{\alpha\nu}{1-\alpha}.$$

Thus in the former case, the sample complexity is defined in terms of small additive deviations from optimality, and in the latter, we allow both additive and multiplicative deviations. These deviations are controlled by the parameters $\alpha$ and $\nu$.

For example, when $\mathbf{r}^*_{\mathbf{L}}(P) = 0$ as in the standard PAC model, then setting $\alpha = 1/2$ and $\nu = \epsilon$ makes the $L_\epsilon$ and $L_{\alpha,\nu}$ conditions equivalent; each reduces to the PAC condition

$$\mathbf{r}_{\hat{h},\mathbf{L}}(P) \leq \epsilon.$$

When $\mathbf{r}^*_{\mathbf{L}}(P) > 0$, then $L_{\alpha,\nu}$ condition approximates the $L_\epsilon$ condition when $\alpha$ is small and $\nu \approx \epsilon/\alpha$. In particular, since we are assuming that the underlying loss function $\mathbf{L}$ is bounded between 0 and $M$, we have $0 \leq \mathbf{r}_{\hat{h},\mathbf{L}}(P), \mathbf{r}^*_{\mathbf{L}}(P) \leq M$ and property (3) of the $d_\nu$ metric shows that the $L_{\alpha,\nu}$ condition with $\nu = 2M$ and $\alpha = \epsilon/4M$ implies the $L_\epsilon$ condition. This shows how the two parameter $L_{\alpha,\nu}$ condition is generally more flexible than the single parameter $L_\epsilon$ condition.

## 2.5 Relation between learning and optimization

Let us assume that the underlying loss function $\mathbf{L}$ is fixed, and we are using either the $L_\epsilon$ or $L_{\alpha,\nu}$ regret functions derived from $\mathbf{L}$. In order to solve a basic learning problem, we must find, with high probability, a decision rule $\hat{h}$ with risk close to optimal. As the true distribution $P$ is unknown, to do this we must rely on estimates of $\mathbf{r}_h(P)$ for the various $h \in \mathcal{H}$ which are derived from the given random training sample. For a given $h \in \mathcal{H}$ and training sample $\bar{z} = (z_1, \ldots, z_m)$, where $z_i = (x_i, y_i) \in Z$, let $\widehat{\mathbf{r}}_h(\vec{z})$ denote the empirical risk on $\bar{z}$, i.e. $\widehat{\mathbf{r}}_h(\vec{z}) = \frac{1}{m}\sum_{i=1}^m \mathbf{L}(y_i, h(x_i))$. Let $\widehat{\mathbf{r}}^*(\vec{z}) = \inf\{\widehat{\mathbf{r}}_h(\vec{z}) : h \in \mathcal{H}\}$. We can then define a natural optimization problem associated with the basic learning problem: given the training sample $\bar{z}$, find a decision rule $\hat{h} \in \mathcal{H}$ such that $\widehat{\mathbf{r}}_{\hat{h}}(\vec{z})$ is close to $\widehat{\mathbf{r}}^*(\vec{z})$, i.e. a decision rule whose empirical risk on the training sample is close to minimal.

Solving the optimization problem does not automatically solve the learning problem. We need to have good empirical risk estimates as well. Since $\mathbf{L}$ is bounded, for every $h \in \mathcal{H}$, as the sample size $m \to \infty$, $\widehat{\mathbf{r}}_h(\vec{z}) \to \mathbf{r}_h(P)$ with probability 1. We will say that the empirical risk estimates of decision rules in $\mathcal{H}$ converge *uniformly* to the true risk if for all $\epsilon$ and $\delta > 0$ there exists a sample size $m$ such that when the $z_i \in \bar{z}$, $1 \leq i \leq m$, are drawn independently at random from $Z$ according to the distribution $P$, with probability at least $1 - \delta$, we have $\rho(\widehat{\mathbf{r}}_h(\vec{z}), \mathbf{r}_h(P)) \leq \epsilon$ for all $h \in \mathcal{H}$. Here $\rho$ is some metric on $\Re^+$, e.g. either the absolute difference or the $d_\nu$ metric.

The following result shows that uniform convergence of the empirical risk estimates, along with a learning method $\mathcal{L}$ that gives a randomized solution to the optimization problem on the estimates, gives a solution to the basic learning problem. We state it for the $d_\nu$ metric, but the same argument works also for the absolute difference metric.

**Lemma 1** *Let $\nu > 0$ and $0 < \alpha, \delta < 1$. Suppose the sample size $m = m(\alpha, \nu, \delta)$ is such that for all probability distributions $P \in \mathcal{P}$*

$$\mathbf{Pr}(\exists h \in \mathcal{H} : d_\nu(\widehat{\mathbf{r}}_h(\vec{z}), \mathbf{r}_h(P)) > \alpha/3) \leq \delta/2,$$

*where the $z_i \in \bar{z}$, $1 \leq i \leq m$, are drawn independently at random from $Z$ according to the distribution $P$. Suppose also that the algorithm $\mathcal{L}$ is such that for all $P \in \mathcal{P}$*

$$\mathbf{Pr}(d_\nu(\widehat{\mathbf{r}}_{\mathcal{L}(\bar{z})}(\vec{z}), \widehat{\mathbf{r}}^*(\vec{z})) > \alpha/3) \leq \delta/2,$$

*where $\bar{z}$ is drawn randomly by $P$ as above. Then for all $P \in \mathcal{P}$*

$$\mathbf{Pr}(d_\nu(\mathbf{r}_{\mathcal{L}(\bar{z})}(P), \mathbf{r}^*(P)) > \alpha) \leq \delta,$$

*i.e. $\mathcal{L}$ solves the basic learning problem for the family of $L_{\alpha,\nu}$ regret functions and has sample complexity at most $m(\alpha, \nu, \delta)$.*

**Proof.** By the triangle inequality for $d_\nu$, if

1. $d_\nu(\mathbf{r}_{\mathcal{L}(\bar{z})}(P), \widehat{\mathbf{r}}_{\mathcal{L}(\bar{z})}(\vec{z})) \leq \alpha/3$,
2. $d_\nu(\widehat{\mathbf{r}}_{\mathcal{L}(\bar{z})}(\vec{z}), \widehat{\mathbf{r}}^*(\vec{z})) \leq \alpha/3$, and
3. $d_\nu(\widehat{\mathbf{r}}^*(\vec{z}), \mathbf{r}^*(P)) \leq \alpha/3$,

then

$$d_\nu(\mathbf{r}_{\mathcal{L}(\bar{z})}(P), \mathbf{r}^*(P)) \leq \alpha.$$

The second assumption of the lemma states that (2) holds with probability at least $1 - \delta/2$. The first assumption implies that both (1) and (3) hold with probability at least $1-\delta/2$. (If (3) fails then we can find a decision rule $h \in \mathcal{H}$ such that $d_\nu(\widehat{\mathbf{r}}_h(\vec{z}), \mathbf{r}_h(P)) > \alpha/3$. Here we use the compatibility of $d_\nu$ with the ordering on the reals.) Hence with probability at least $1 - \delta$ all of (1) - (3) hold. The result follows. □

In statistics, this type of result is called a *consistency theorem* about the "statistic" (i.e. the decision rule) computed by the learning method $\mathcal{L}$. This use of the term "consistency" differs sharply from that common in PAC learning research.

# 3 Uniformly good empirical estimates of means

In this section we concentrate on the problem of bounding the number of random examples needed to get good empirical estimates of the risk of each of the decision rules in a decision rule space $\mathcal{H}$. For each decision rule $h \in \mathcal{H}$ and example $z = (x, y) \in Z$, let $l_h(z) = \mathbf{L}(y, h(x))$. As in the previous section, we assume that $\mathbf{L}$ is a non-negative bounded loss function taking values in the interval $[0, M]$, thus for each decision rule $h$, $l_h$ defines a random variable taking values in $[0, M]$. The value of $l_h$ on an example $(x, y)$ is the loss incured when you use $h$ to determine the action to take for instance $x$, and the outcome is $y$. The risk of $h$ is just the expectation of $l_h$, i.e.

$$\mathbf{r}_h(P) = \mathbf{E}(l_h) = \int_Z l_h(z) dP(z).$$

Furthermore, if $\bar{z} = (z_1, \ldots, z_m)$ is a sequence of examples from $Z$, then the empirical risk of $h$ on $\bar{z}$ is the empirical estimate of the mean of $l_h$ based on the sample $\bar{z}$, which we denote by $\widehat{\mathbf{E}}_{\bar{z}}(l_h)$, i.e.

$$\widehat{\mathbf{r}}_h(\vec{z}) = \widehat{\mathbf{E}}_{\bar{z}}(l_h) = \frac{1}{m} \sum_{i=1}^{m} l_h(z_i).$$

Let $l_{\mathcal{H}} = \{l_h : h \in \mathcal{H}\}$. We need to draw enough random examples to get a uniformly good empirical estimate of the expectation of every random variable in $l_{\mathcal{H}}$.

The general problem of obtaining a uniformly good estimate of the expectation of every function in a class $\mathbf{F}$ of real-valued functions has been widely studied (see e.g. [128, 104, 36] and their references). If no assumptions at all are made about the functions in $\mathbf{F}$, we immediately run into the problem that some functions in $\mathbf{F}$ could take on arbitrarily large values with arbitrarily small probabilities, making it impossible to obtain uniformly good empirical estimates of all expectations with any finite sample size. This problem can be avoided by making assumptions about the moments of the functions in $\mathbf{F}$, as in [128], or by assuming that there exists a single non-negative function with a finite expectation (called an *envelope*) that lies above the absolute value of every function in $\mathbf{F}$, as in [104, 36] (see also Vapnik's chapter). In our case, when the loss takes only values in $[0, M]$, then the constant function $M$ serves as an envelope. This case is especially nice since this same envelope works for all distributions on the domain $Z$ of the functions in $\mathbf{F}$.

The usual measure of deviation of empirical estimates from true means is simply the absolute value of the difference. Thus we would say that the empirical estimates for the expectations of the functions in $\mathbf{F}$ converge uniformly to the true expectations if as the size of the random sample $\bar{z}$

grows,

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : |\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}) - \mathbf{E}(\mathbf{f})| > \epsilon\right)$$

goes to zero for any $\epsilon > 0$. (This is called *(uniform) convergence in probability*, see e.g. [21]). Vapnik, Dudley, Pollard and others have obtained general bounds on the sample size needed so that

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : |\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}) - \mathbf{E}(\mathbf{f})| > \epsilon\right) < \delta$$

for $\epsilon, \delta > 0$ [104] [36] [128]. Vapnik also obtains better bounds in some important cases by considering the relative deviation of empirical estimates from true expectations. He looks at bounds on the sample size needed so that

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : \frac{\mathbf{E}(\mathbf{f}) - \widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f})}{\mathbf{E}(\mathbf{f})} > \epsilon\right) < \delta$$

and also bounds on the sample size needed so that

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : \frac{\mathbf{E}(\mathbf{f}) - \widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f})}{\sqrt{\mathbf{E}(\mathbf{f})}} > \epsilon\right) < \delta.$$

(Anthony and Shawe-Taylor also obtain bounds of the latter form [12].) Note that these are one-sided bounds, in that they only bound the probability that the empirical mean is significantly smaller than the true mean. While extremely useful, as we mentioned in the previous section, these measures of deviation suffer from a discontinuity at $\mathbf{E}(\mathbf{f}) = 0$, and lack of convenient metric properties. As in [105], we will give bounds on the sample size needed so that

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : d_\nu(\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}), \mathbf{E}(\mathbf{f})) > \alpha\right) = \mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : \frac{|\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}) - \mathbf{E}(\mathbf{f})|}{\nu + \widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}) + \mathbf{E}(\mathbf{f})} > \alpha\right) < \delta,$$

i.e. the deviation measured using the $d_\nu$ metric[12]. By setting $\nu$ and $\alpha$ appropriately, we obtain results similar to those of [104] and [128] as special cases of our main theorem. However, our results

[12] In [105], Pollard also gives results that can be used to bound the sample size needed so that

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : \frac{|\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}) - \mathbf{E}(\mathbf{f})|}{\nu + \sqrt{\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f})} + \sqrt{\mathbf{E}(\mathbf{f})}} > \alpha\right) < \delta,$$

in analogy with the second type of bound given by Vapnik, except that these bounds are two-sided. We do not pursue these further here.

are restricted to the case that all functions in $\mathbf{F}$ are positive and uniformly bounded.

### 3.1 The case of finite F

Before considering the general case, it is useful to see what bounds we can get in the case that $\mathbf{F}$ is a finite set of functions. Here we can easily prove the following.

**Theorem 1** *Let $\mathbf{F}$ be a finite set of functions on $Z$ with $0 \leq \mathbf{f}(z) \leq M$ for all $\mathbf{f} \in \mathbf{F}$ and $z \in Z$. Let $\bar{z} = (z_1, \ldots, z_m)$ be a sequence of $m$ examples drawn independently from $Z$ according to any distribution on $Z$, and let $\epsilon > 0$. Then*

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : |\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}) - \mathbf{E}(\mathbf{f})| > \epsilon\right) \leq 2|\mathbf{F}|e^{-2\epsilon^2 m/M^2}.$$

*For $0 < \delta \leq 1$ and sample size*

$$m \geq \frac{M^2}{2\epsilon^2}\left(\mathbf{ln}|\mathbf{F}| + \mathbf{ln}\frac{2}{\delta}\right)$$

*this probability is at most $\delta$. Further, for any $\nu > 0$ and $0 < \alpha < 1$,*

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : d_\nu(\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}), \mathbf{E}(\mathbf{f})) > \alpha\right) \leq 2|\mathbf{F}|e^{-\alpha^2 \nu m/M}.$$

*For $0 < \delta \leq 1$ and sample size*

$$m \geq \frac{M}{\alpha^2 \nu}\left(\mathbf{ln}|\mathbf{F}| + \mathbf{ln}\frac{2}{\delta}\right)$$

*this probability is at most $\delta$.* $\square$

**Proof.** For the second part of the theorem, using Bernstein's inequality (see e.g. [104]) it is easy to show that for any single function $\mathbf{f}$ with $0 \leq \mathbf{f} \leq M$,

$$\mathbf{Pr}\left(d_\nu(\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}), \mathbf{E}(\mathbf{f})) > \alpha\right) < 2e^{-\alpha^2 \nu m/M}.$$

Details are given in Lemma 9, part (2) in the Appendix of [54]. It follows that the probability that there is any $\mathbf{f} \in \mathbf{F}$ with $d_\nu(\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}), \mathbf{E}(\mathbf{f})) > \alpha$ is at most $2|\mathbf{F}|e^{-\alpha^2 \nu m/M}$. Setting this bound to $\delta$ and solving for $m$ gives the result on the sample size. The proof of the first part of the lemma is similar, except we use Hoeffding's inequality (see e.g. [104]), which implies that for any single $\mathbf{f}$,

$$\mathbf{Pr}\left(|\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}) - \mathbf{E}(\mathbf{f})| > \epsilon\right) \leq 2e^{-2\epsilon^2 m/M^2}.$$

$\square$

By letting $\mathbf{F} = l_{\mathcal{H}}$, this theorem can be used in conjunction with Lemma 1 from the previous section to obtain bounds on the sample complexity of learning algorithms that minimize empirical risk. Here we can use either the $L_\epsilon$ or $L_{\alpha,\nu}$ family of regret functions. In the former case we get a sample complexity

$$m(\epsilon, \delta) = O\left(\frac{M^2}{\epsilon^2}\left(\mathbf{log}|l_{\mathcal{H}}| + \mathbf{log}\frac{1}{\delta}\right)\right). \tag{1}$$

In the latter case we get a sample complexity

$$m(\alpha, \nu, \delta) = O\left(\frac{M}{\alpha^2\nu}\left(\mathbf{log}|l_{\mathcal{H}}| + \mathbf{log}\frac{1}{\delta}\right)\right). \tag{2}$$

As shown in the previous section, a generalization of the PAC learning model can be obtained by using either the $L_\epsilon$ or $L_{\alpha,\nu}$ regret functions, in the latter case by setting $\alpha = 1/2$ and $\nu = \epsilon$. Note that plugging this latter setting into (2) gives a sample complexity

$$m(\epsilon, \delta) = O\left(\frac{M}{\epsilon}\left(\mathbf{log}|l_{\mathcal{H}}| + \mathbf{log}\frac{1}{\delta}\right)\right), \tag{3}$$

a significant improvement over (1), which is quadratic in $M/\epsilon$. Thus the generalization of the PAC model using the $d_\nu$ metric to measure distance from optimality, and the resulting $L_{\alpha,\nu}$ family of regret functions, offers new insight in this regard. (Vapnik's use of the relative difference between empirical estimates and true expectations [128] also has this advantage; see [12], the appendix of [24] and Vapnik's chapter.)

### 3.2 The general case

The main task of this section is to generalize Theorem 1 to infinite collections of uniformly bounded functions. The basic idea is simple: we replace the infinite class $\mathbf{F}$ of functions with a finite class $\mathbf{F}_0$ that "approximates" it, in the sense that each function in $\mathbf{F}$ is close to some function in $\mathbf{F}_0$, and argue that some type of uniform convergence of empirical estimates for $\mathbf{F}_0$ implies uniform convergence for $\mathbf{F}$. In the simplest version of this technique, the choice of $\mathbf{F}_0$ depends only on $\mathbf{F}$ and the distribution $P$, as in the "direct method" discussed in section II.2 of [104] (see also [128] section 6.6, [36] chapter 6, [18], [137], and White's chapter). However, more general results (apart from certain measurability constraints) are obtained by allowing $\mathbf{F}_0$ to depend on the particular random sample $\bar{z}$ (e.g. [104], chapter 2). Here $\mathbf{F}_0$ is called a "random cover", and its size is called a "random covering number". It is this type of result that we derive here.

We will need a few preliminary definitions to introduce the notion of $\epsilon$-covers and metric dimension. A more general treatment of these ideas is given in the appendix, section 8.1. This more general treatment will be used in the next section, but the following definitions suffice for this section.

For any real vectors $\bar{x} = (x_1, \ldots, x_m)$ and $\bar{y} = (y_1, \ldots, y_m)$ in $\Re^m$, let $d_{L^1}(\bar{x}, \bar{y}) = \frac{1}{m}\sum_{i=1}^{m} |x_i - y_i|$. Thus $d_{L^1}$ is the $L^1$ distance metric. Let $T$ be a set of points that lie in a bounded region of $\Re^m$. For any $\epsilon > 0$, an *$\epsilon$-cover* for $T$ is a finite set $N \subset \Re^m$ (not necessarily contained in $T$) such that for all $\bar{x} \in T$ there is a $\bar{y} \in N$ with $d_{L^1}(\bar{x}, \bar{y}) \leq \epsilon$. The function $\mathcal{N}(\epsilon, T)$ denotes the size of the smallest $\epsilon$-cover for $T$. We refer to $\mathcal{N}(\epsilon, T)$ as a *covering number*.

Following [68] we define the *upper metric dimension* of the set $T$ of points by

$$\overline{\mathbf{dim}}(T) = \mathbf{limsup}_{\epsilon \to 0} \frac{\mathbf{log} \mathcal{N}(\epsilon, T)}{\mathbf{log}(1/\epsilon)}.$$

The *lower metric dimension*, denoted by $\underline{\mathbf{dim}}$, is defined similarly using **liminf**. When $\overline{\mathbf{dim}}(T) = \underline{\mathbf{dim}}(T)$, then this quantity is denoted $\mathbf{dim}(T)$, and referred to simply as the *metric dimension* of $T$. Note that if $\mathcal{N}(\epsilon, T) = (g(\epsilon)/\epsilon)^n$, where $g(\epsilon)$ is polylogarithmic in $1/\epsilon$, then $\mathbf{dim}(T) = n$. Hence the metric dimension essentially picks out the exponent in the rate of growth of the covering number as a function of $1/\epsilon$.

Assume all functions in $\mathbf{F}$ map from $Z$ into $[0, M]$. For any sample $\bar{z} = (z_1, \ldots, z_m)$, with $z_i \in Z$, let

$$\mathbf{F}_{|\bar{z}} = \{(f(z_1), \ldots, f(z_m)) : f \in \mathbf{F}\}.$$

We will call $\mathbf{F}_{|\bar{z}}$ the *restriction of* $\mathbf{F}$ *to* $\bar{z}$. Note that $\mathbf{F}_{|\bar{z}}$ is a set of points in the $m$-cube $[0, M]^m$. We can consider the size of the covering number $\mathcal{N}(\epsilon, \mathbf{F}_{|\bar{z}})$ as giving some indication of the "richness at scale $\approx \epsilon$" of the class $\mathbf{F}$ of functions, restricted to the domain $z_1, \ldots, z_m$. The metric dimension of $\mathbf{F}_{|\bar{z}}$ gives some indication of the "number of essential degrees of freedom" in this restriction of $\mathbf{F}$.

When $z_1, \ldots, z_m$ are drawn independently at random from $Z$, the *random covering number* $\mathbf{E}(\mathcal{N}(\epsilon, \mathbf{F}_{|\bar{z}}))$ gives some indication of the "richness" of $\mathbf{F}$ on a "typical" set of $m$ points in the domain $Z$. Note that for finite $\mathbf{F}$, we have $\mathcal{N}(\epsilon, \mathbf{F}_{|\bar{z}}) \leq |\mathbf{F}|$ for all $\epsilon$ and all samples $\bar{z}$, and hence the random covering number $\mathbf{E}(\mathcal{N}(\epsilon, \mathbf{F}_{|\bar{z}})) \leq |\mathbf{F}|$ for all $\epsilon$, all sample sizes $m$, and all distributions on $Z$. The main result about uniform empirical estimates for infinite classes of functions is similar to Theorem 1 except that the random covering numbers are used in place of $|\mathbf{F}|$.

**Theorem 2** *([105]) Let* $\mathbf{F}$ *be a permissible*[13] *set of functions on* $Z$ *with* $0 \leq \mathbf{f}(z) \leq M$ *for all* $\mathbf{f} \in \mathbf{F}$ *and* $z \in Z$. *Let* $\bar{z} = (z_1, \ldots, z_m)$ *be a sequence of* $m$ *examples drawn independently from* $Z$ *according to any distribution on* $Z$. *Then for any* $\nu > 0$ *and* $0 < \alpha < 1$,

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : d_\nu(\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}), \mathbf{E}(\mathbf{f})) > \alpha\right) \leq 4\mathbf{E}\left(\mathcal{N}(\alpha\nu/8, \mathbf{F}_{|\bar{z}})\right) e^{-\alpha^2 \nu m/16M}. \ \Box$$

**Corollary 1** *([104]) Under the same assumptions as above, for all* $\epsilon > 0$,

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : |\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}) - \mathbf{E}(\mathbf{f})| > \epsilon\right) \leq 4\mathbf{E}\left(\mathcal{N}(\epsilon/16, \mathbf{F}_{|\bar{z}})\right) e^{-\epsilon^2 m/128M^2}.$$

**Proof of corollary.** This follows directly from the above result by setting $\nu = 2M$, and $\alpha = \epsilon/4M$. To see this, note that property (3) of the $d_\nu$ metric (section 2.2) implies that $|r - s| \leq \epsilon$ whenever $d_\nu(r, s) \leq \alpha$ for all $0 \leq r, s \leq M$ when this setting of $\nu$ and $\alpha$ is used. $\Box$

The constants in these results are only crude estimates. No serious attempt has been made to minimize them. (See the recent results of Talagrand [119] for much better constants for Corollary 1).

The bound in this latter result depends critically on the relative magnitudes of the negative exponent in $e^{-\epsilon^2 m/128M^2}$ and the exponent in the expectation of the covering number $\mathcal{N}(\epsilon/16, \mathbf{F}_{|\bar{z}})$, which reflects the extent to which $\mathbf{F}_{|\bar{z}}$ "fills up" the $m$-cube $[0, M]^m$. For example, if $\mathbf{F}_{|\bar{z}}$ has metric dimension at most $n$ for all $m$ and all $\bar{z}$, then there is a constant $c_0$ such that for any $\eta > 0$, $\mathcal{N}(\epsilon/16, \mathbf{F}_{|\bar{z}}) \leq (c_0 M/\epsilon)^{n+\eta}$ for suitably small $\epsilon$. In this case the negative exponential term eventually dominates the expected covering number, and beyond a critical sample size

$$m_0 = O\left(\frac{nM^2}{\epsilon^2} \mathbf{log} \frac{M}{\epsilon}\right),$$

the bound goes to zero exponentially fast. We will see examples of this in the following section, where we give bounds on the metric dimension of $\mathbf{F}_{|\bar{z}}$ in terms of a combinatorial parameter called the pseudo dimension of $\mathbf{F}$. The theorem actually shows that this exponential drop off occurs even if this metric dimension bound holds only for "most" $\bar{z}$.

[13] This is a measurability condition defined in [104] which need not concern us in practice. A detailed discussion is given in [54].

On the other hand, if with high probability $\mathbf{F}_{|\bar{z}}$ "fills up" the $m$-cube $[0, M]^m$ to the extent that $\mathcal{N}(\epsilon/16, \mathbf{F}_{|\bar{z}}) \approx (c_0/\epsilon)^m$, which is as large as possible, then the covering number dominates, and the bound is trivial. Results in [128] (Theorem A.2, page 220) indicate that uniform convergence does not take place in this case. Similar remarks apply to the bound given in Theorem 2, which uses the $d_\nu$ metric.

The proof of Theorem 2 follows the proof of Pollard's Theorem 24 ([104], p. 25) in general outline, and is given in [54]. However, the use of the $d_\nu$ metric necessitates a number of substantial modifications. The approach taken here is different from that taken (independently, but prior to this work) by Pollard in [105]. Still different, and more involved, techniques are used in the more general theory of weighted empirical processes developed by Alexander [2, 3].

Actually, we can prove a slightly stronger result than Theorem 2 (see [54], Theorem 12). This result is obtained by bounding the probability of uniform convergence on a sample of length $m$ in terms of the expected covering numbers associated with a sample of length $2m$, and by expanding the expectation to include the negative exponential term with a "truncation" at 1. It turns out that this saves us a factor of 1/2 in the negative exponential term.

**Theorem 3** *Let $\mathbf{F}$ be a permissible set of functions on $Z$ with $0 \leq \mathbf{f}(z) \leq M$ for all $\mathbf{f} \in \mathbf{F}$ and $z \in Z$. Assume $\nu > 0$, $0 < \alpha < 1$ and $m \geq 1$. Suppose that $\bar{z}$ is generated by $m$ independent random draws according to any probability measure on $Z$. Let*

$$p(\alpha, \nu, m) = \mathbf{Pr}\left\{\bar{z} \in Z^m : \exists \mathbf{f} \in \mathbf{F} \text{ with } d_\nu(\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}), \mathbf{E}(\mathbf{f})) > \alpha\right\}.$$

*Then*

$$p(\alpha, \nu, m) \leq 2\mathbf{E}(\min(2\mathcal{N}(\alpha\nu/8, \mathbf{F}_{|\bar{z}}, d_{L^1})e^{-\alpha^2\nu m/8M}, 1)),$$

*where the expectation is over $\bar{z}$ drawn randomly from $Z^{2m}$. If in addition $\mathbf{F}_{|\bar{z}}$ is finite for all $\bar{z} \in Z^{2m}$ then*

$$p(\alpha, \nu, m) \leq 2\mathbf{E}(min(2|\mathbf{F}_{|\bar{z}}|e^{-\alpha^2\nu m/2M}, 1)).$$

Theorem 2 is obtained as a corollary of this result by substituting $m/2$ for $m$ and not taking the minimum with 1 in the left hand side of the first bound for $p(\alpha, \nu, m)$. We will use Theorem 3 to obtain slightly better constants in some of the results in the sequel.

## 4 Capacity and Metric Dimension of Function Classes

In this section we develop a way of obtaining distribution independent bounds on the random covering numbers needed for Theorems 2 and 3. The key idea is to introduce a pseudo metric (see section 8.1) on the decision space $A$. The distance between two actions is the maximum difference in loss for these actions, over all possible outcomes.

**Definition 2** *For every loss function* $\mathbf{L} : Y \times A \to [0, M]$*, by* $\rho_{\mathbf{L}}$ *we denote the pseudo metric on* $A$ *defined by* $\rho_{\mathbf{L}}(a, b) = \sup_{y \in Y} |\mathbf{L}(y, a) - \mathbf{L}(y, b)|$ *for all* $a, b \in A$*.*

Note that $(A, \rho_{\mathbf{L}})$ is a bounded pseudo metric space: no two actions in $A$ are more than $M$ apart.

The notions of $\epsilon$-cover and metric dimension used in the previous section can be generalized to arbitrary pseudo metric spaces. This generalization is given in section 8.1 of the appendix. In the remainder of the paper we will use the concepts and notation given there without further special reference.

Since decision rules in $\mathcal{H}$ map from the instance space $X$ into $A$, the pseudo metric $\rho_{\mathbf{L}}$ on $A$ can be used to induce a pseudo metric on $\mathcal{H}$ in which two decision rules differ only to the extent that the actions that they proscribe differ with respect to $\rho_{\mathbf{L}}$. There are several ways to do this. The easiest is to use an $L^\infty$ function distance on $\mathcal{H}$, defining the distance between decision rules $f$ and $g$ as the supremum of $\rho_{\mathbf{L}}(f(x), g(x))$ over all $x \in X$. This works, and is a useful method of obtaining uniform convergence and learning results (see related techniques used in [137] and White's chapter). However, as we will see, the crucial issue is the size of the smallest $\epsilon$-cover of the resulting pseudo metric space $\mathcal{H}$. In some cases we can get smaller covers, and hence better results, by using an $L^1$ function distance instead. Since the $L^1$ distance is never more than the $L^\infty$ distance, the results are never worse. Thus we present this more powerful method here.

**Definition 3** *Let* $\mathcal{H}$ *be a family of functions from a set* $X$ *into a bounded pseudo metric space* $(A, \rho)$*. Let* $P$ *be a probability measure on* $X$*. Then* $d_{L^1(P,\rho)}$ *is the pseudo metric on* $\mathcal{H}$ *defined by*

$$d_{L^1(P,\rho)}(f, g) = \mathbf{E}(\rho(f(x), g(x))) = \int_X \rho(f(x), g(x)) dP(x)$$

*for all* $f, g \in \mathcal{H}$*. For every* $\epsilon > 0$ *let*

$$\mathcal{C}(\epsilon, \mathcal{H}, \rho) = sup\{\mathcal{N}(\epsilon, \mathcal{H}, d_{L^1(P,\rho)})\} \text{ over all probability measures } P \text{ on } X.$$

*If $\mathcal{N}(\epsilon, \mathcal{H}, d_{L^1(P,\rho)})$ is infinite for some measure $P$, or if the set in this supremum is unbounded, then $\mathcal{C}(\epsilon, \mathcal{H}, \rho) = \infty$. We call* [14] *$\mathcal{C}(\epsilon, \mathcal{H}, \rho)$ the* capacity *of $\mathcal{H}$. In analogy with the definition of metric dimension, we define the* upper metric dimension *of $\mathcal{H}$ by*

$$\overline{\mathbf{dim}}(\mathcal{H}) = \mathbf{limsup}_{\epsilon \to 0} \frac{\mathbf{log}\mathcal{C}(\epsilon, \mathcal{H}, \rho)}{\mathbf{log}(1/\epsilon)},$$

*and the* lower metric dimension*, denoted by $\underline{\mathbf{dim}}(\mathcal{H})$, is defined similarly using* **liminf**. *When $\overline{\mathbf{dim}}(\mathcal{H}) = \underline{\mathbf{dim}}(\mathcal{H})$, then this quantity is denoted $\mathbf{dim}(\mathcal{H})$, and referred to simply as the* metric dimension *of $\mathcal{H}$. If $\mathcal{C}(\epsilon, \mathcal{H}, \rho) = \infty$ for some $\epsilon > 0$ then $\mathbf{dim}(\mathcal{H}) = \infty$.*

We now show how bounds on the capacity of $\mathcal{H}$ lead to distribution-independent bounds on the rate of uniform convergence of empirical risk estimates for functions in $\mathcal{H}$ with respect to the loss function $\mathbf{L}$. As before, let $Z = X \times Y$, $P$ be a probability distribution on $Z$, and $l_{\mathcal{H}}$ be the family of functions on $Z$ defined by $l_{\mathcal{H}} = \{l_h : h \in \mathcal{H}\}$, where $l_h(x, y) = \mathbf{L}(y, h(x))$. Let $P_{|X}$ be the marginal on $X$ of the joint distribution $P$ on $X \times Y$ (see section 1.5).

**Lemma 2** *For all $\epsilon > 0$,*

$$\mathcal{N}(\epsilon, l_{\mathcal{H}}, d_{L^1(D)}) \leq \mathcal{N}(\epsilon, \mathcal{H}, d_{L^1(P_{|X}, \rho_{\mathbf{L}})}).$$

**Proof.** For every $h \in \mathcal{H}$ let $\psi(h) = l_h$. Hence $\psi$ maps from $\mathcal{H}$ onto $l_{\mathcal{H}}$. It suffices to show that $\psi$ is a contraction, i.e. that

$$\forall\, f, g \in \mathcal{H},\ d_{L^1(D)}(\psi(f), \psi(g)) \leq d_{L^1(P_{|X}, \rho_{\mathbf{L}})}(f, g).$$

Let $f$ and $g$ be any two functions in $\mathcal{H}$. Then

$$\begin{aligned} d_{L^1(D)}(\psi(f), \psi(g)) &= \int_Z |\mathbf{L}(y, f(x)) - \mathbf{L}(y, g(x))| dP(x, y) \\ &\leq \int_Z \rho_{\mathbf{L}}(f(x), g(x)) dP(x, y) \\ &= \int_X \rho_{\mathbf{L}}(f(x), g(x)) dP_{|X}(x) \end{aligned}$$

[14] The term *metric entropy* is often used for the quantities $\mathbf{log}\mathcal{N}(\epsilon, \mathcal{H}, d_{L^1(P,\rho)})$ and $\mathbf{log}\mathcal{C}(\epsilon, \mathcal{H}, \rho)$ [37, 107]. It is also used for an analogous, but fundamentally distinct, concept in the dynamical systems literature (e.g. [41]). The term *capacity* has also been used with many other related meanings [83, 128, 68, 42, 17]. Our usage here is taken from [36].

$$= \quad d_{L^1(P_{|X}, \rho_{\mathbf{L}})}(f,g).$$

□

This gives the following theorem about distribution-independent uniform convergence of risk estimates for learning.

**Theorem 4** *Assume that the decision rule space $\mathcal{H}$ and the loss function $\mathbf{L}$ are such that $l_{\mathcal{H}}$ is permissible. Let $P$ be any probability distribution on $Z = X \times Y$. Assume $m \geq 1$, $\nu > 0$ and $0 < \alpha < 1$. Let $\bar{z}$ be generated by $m$ independent draws from $Z$ according to $P$. Then*

$$\mathbf{Pr}\left(\exists h \in \mathcal{H} : d_\nu(\widehat{\mathbf{r}}_{h,\mathbf{L}}(\vec{z}), \mathbf{r}_{h,\mathbf{L}}(P)) > \alpha\right) \leq 4\mathcal{C}(\alpha\nu/8, \mathcal{H}, \rho_{\mathbf{L}})e^{-\alpha^2\nu m/8M}.$$

**Proof.** Let $\mathbf{F} = l_{\mathcal{H}}$. For any sequence $\bar{z}$ of points in $Z$ there is a trivial isometry between $(\mathbf{F}_{|\bar{z}}, d_{L^1})$ and $(\mathbf{F}, d_{L^1(P_{\bar{z}})})$, where $P_{\bar{z}}$ is the empirical measure induced by $\bar{z}$, in which each set has measure equal to the fraction of the points in $\bar{z}$ it contains. Thus by Lemma 2 above, we have

$$\mathcal{N}(\epsilon, \mathbf{F}_{|\bar{z}}, d_{L^1}) = \mathcal{N}(\epsilon, \mathbf{F}, d_{L^1(P_{\bar{z}})}) \leq \mathcal{N}(\epsilon, \mathcal{H}, d_{L^1(P_{\bar{z}|X}, \rho_{\mathbf{L}})}) \leq \mathcal{C}(\epsilon, \mathcal{H}, \rho_{\mathbf{L}})$$

for all $\bar{z} \in Z^{2m}$. Hence, setting $\epsilon = \alpha\nu/8$, the given probability is at most $4\mathcal{C}(\alpha\nu/8, \mathcal{H}, \rho_{\mathbf{L}})e^{-\alpha^2\nu m/8M}$ by Theorem 3. □

In order to apply the above theorem, we need tools for bounding the capacity of various decision rule spaces. Along these lines, we close this section by stating two basic lemmas, one about the capacity of the free product of a set of function classes, and the other about the capacity of compositions of functions classes. Proofs can be found in [54].

**Definition 4** *Let $(A_1, \rho_1), \ldots, (A_k, \rho_k)$ be bounded metric spaces. Let $A = A_1 \times \cdots \times A_k$ and $\rho$ be the metric on $A$ defined by*

$$\rho(\bar{u}, \bar{v}) = \frac{1}{k}\sum_{j=1}^{k} \rho_j(u_j, v_j)$$

*for any $\bar{u} = (u_1, \ldots, u_k)$ and $\bar{v} = (v_1, \ldots, v_k) \in A$. For each $j$, $1 \leq j \leq k$, let $\mathcal{H}_j$ be a family of functions from $X$ into $A_j$. The* free product *of $\mathcal{H}_1$ through $\mathcal{H}_k$ is the class of functions*

$$\mathcal{H} = \{(f_1, \ldots, f_k) : f_j \in \mathcal{H}_j, 1 \leq j \leq k\},$$

*where* $(f_1,\ldots,f_k): X \to A$ *is the function defined by*

$$(f_1,\ldots,f_k)(x) = (f_1(x),\ldots,f_k(x)).$$

**Lemma 3** *If* $\mathcal{H},\mathcal{H}_1,\ldots,\mathcal{H}_k$ *are defined as above then*

1. *for any probability measure* $P$ *on* $X$ *and* $\epsilon > 0$,

$$\prod_{j=1}^{k} \mathcal{N}(2k\epsilon,\mathcal{H}_j,d_{L^1(P,\rho_j)}) \leq \mathcal{N}(\epsilon,\mathcal{H},d_{L^1(P,\rho)}) \leq \prod_{j=1}^{k} \mathcal{N}(\epsilon,\mathcal{H}_j,d_{L^1(P,\rho_j)}),$$

2. $\overline{\mathbf{dim}}(\mathcal{H}) = \sum_{j=1}^{k} \overline{\mathbf{dim}}(\mathcal{H}_j)$, *and similarly for* $\underline{\mathbf{dim}}$ *and* $\mathbf{dim}$, *when the latter is defined.*

**Definition 5** *Let* $P$ *be a probability measure on* $X$ *and* $f$ *be a measurable function from* $X$ *into* $Y$. *Then* $P_f$ *denotes the probability measure on* $Y$ *induced by* $f$, *i.e.*

$$P_f(S) = P(f^{-1}(S)) \text{ for all measurable } S \subset Y.$$

**Definition 6** *Let* $f$ *be a function from a metric space* $(X,\rho)$ *into a metric space* $(Y,\sigma)$. *A* Lipschitz bound *on* $f$ *is a real number* $b > 0$ *such that for all* $x,y \in X$, $\sigma(f(x),f(y)) \leq b\,\rho(x,y)$. The *Lipschitz bound on* $f$ *is the smallest such* $b$. *If* $\mathcal{F}$ *is a class of functions from* $(X,\rho)$ *into* $(Y,\sigma)$ *then* $b$ *is a* uniform Lipschitz bound *on* $\mathcal{F}$ *if* $b$ *is a Lipschitz bound on* $f$ *for all* $f \in \mathcal{F}$.

**Lemma 4** *Let* $(X_1,\rho_1),\ldots,(X_{k+1},\rho_{k+1})$ *be metric spaces, where* $(X_j,\rho_j)$ *is bounded,* $2 \leq j \leq k$, *and* $\mathcal{H}_j$ *be a class of functions with* $f: X_j \to X_{j+1}$ *for all* $f \in \mathcal{H}_j$, $1 \leq j \leq k$. *Let* $b_j$ *be a uniform Lipschitz bound on* $\mathcal{H}_j$ *for all* $2 \leq j \leq k$. *Let* $\mathcal{H}$ *denote the class of all functions from* $X_1$ *into* $X_{k+1}$ *defined by compositions of functions in the* $\mathcal{H}_j$*'s, i.e.*

$$\mathcal{H} = \{f_k \circ f_{k-1} \circ \cdots \circ f_1 : f_j \in \mathcal{H}_j, 1 \leq j \leq k\}.$$

1. *For any* $\epsilon,\epsilon_1,\ldots,\epsilon_k > 0$ *such that*

$$\epsilon = \sum_{j=1}^{k} \left( \prod_{l=j+1}^{k} b_l \right) \epsilon_j$$

*we have*

$$\mathcal{C}(\epsilon, \mathcal{H}, \rho_{k+1}) \le \prod_{j=1}^{k} \mathcal{C}(\epsilon_j, \mathcal{H}_j, \rho_{j+1}).$$

*2.* $\overline{\mathbf{dim}}(\mathcal{H}) \le \sum_{j=1}^{k} \overline{\mathbf{dim}}(\mathcal{H}_j)$, *and similarly for* $\underline{\mathbf{dim}}$ *and* $\mathbf{dim}$, *when the latter is defined.*

# 5 Sample size bounds for learning with multi-layer neural nets

We now present some applications of the results of the previous section to learning with feedforward neural nets (see e.g. [113, 103]). The decision rule space $\mathcal{H}$ represented by a feedforward neural net consists of a family of functions from an instance space $X \subset \Re^n$ into a decision space $A \subset \Re^k$ for some $k, n \ge 1$. To apply Theorem 4 of the previous section, we will need to obtain an upper bound on the capacity $\mathcal{C}(\epsilon, \mathcal{H}, \rho_{\mathbf{L}})$ of such decision rule spaces for various loss functions $\mathbf{L}$.

For many loss functions, the metric $\rho_{\mathbf{L}}$ on $A \subset \Re^k$ can be bounded in terms of the $d_{L^1}$ metric, i.e. we can find a constant $c_{\mathbf{L}}$ such that for all $\vec{a} = (a_1, \ldots, a_k)$ and $\bar{b} = (b_1, \ldots, b_k)$ in $A$, $\rho_{\mathbf{L}}(\vec{a}, \bar{b}) \le c_{\mathbf{L}} d_{L^1}(\vec{a}, \bar{b}) = \frac{c_{\mathbf{L}}}{k} \sum_{i=1}^{k} |a_i - b_i|$. In this case it is clear that $\mathcal{C}(\epsilon, \mathcal{H}, \rho_{\mathbf{L}}) \le \mathcal{C}(\epsilon/c_{\mathbf{L}}, \mathcal{H}, d_{L^1})$. Thus our problem is reduced to obtaining an upper bound on the capacity $\mathcal{C}(\epsilon/c_{\mathbf{L}}, \mathcal{H}, d_{L^1})$.

We now give a few examples to illustrate this reduction. First consider the common case in which the outcome space $Y$ is also contained in $\Re^k$, e.g. we receive explicit feedback on each coordinate of our action $\vec{a} \in A \subset \Re^k$. This occurs when each coordinate $a_i$ of the action $\vec{a}$ is a prediction of the corresponding coordinate of the outcome $\bar{y}$. Here the loss function $\mathbf{L}$ may itself be a metric on $\Re^k$ which measures the distance between the predicted vector and the actual outcome vector. When $\mathbf{L}$ is a metric, we have for any actions $\vec{a}, \bar{b} \in A$

$$\rho_{\mathbf{L}}(\vec{a}, \bar{b}) = \sup_{\bar{y} \in Y} |\mathbf{L}(\bar{y}, \vec{a}) - \mathbf{L}(\bar{y}, \bar{b})| \le \mathbf{L}(\vec{a}, \bar{b})$$

by the triangle inequality for $\mathbf{L}$. Thus if the metric $\mathbf{L}$ is bounded with respect to $d_{L^1}$ metric, i.e. $\mathbf{L}(\vec{a}, \bar{b}) \le c_{\mathbf{L}} d_{L^1}(\vec{a}, \bar{b})$ for all $\vec{a}, \bar{b} \in A$, then we have $\rho_{\mathbf{L}}(\vec{a}, \bar{b}) \le c_{\mathbf{L}} d_{L^1}(\vec{a}, \bar{b})$. For example, if $\mathbf{L}(\bar{y}, \vec{a}) = d_{L^2}(\bar{y}, \vec{a}) = \frac{1}{k} \left( \sum_{i=1}^{k} (y_i - a_i)^2 \right)^{1/2}$ then we may take $c_{\mathbf{L}} = 1$, and similarly for the other $d_{L^q}$ metrics, for $q > 1$.

Note that the above trick does not apply to the mean squared loss $\mathbf{L}(\bar{y}, \vec{a}) = \frac{1}{k} \sum_{i=1}^{k} (y_i - a_i)^2$ since this loss does not satisfy the triangle inequality. However, in this case it is easy to show by direct calculation that if the outcome space $Y$ is bounded, e.g. $Y \subset [0, M]^k$, then $\rho_{\mathbf{L}}(\vec{a}, \bar{b}) \le$

$2Md_{L^1}(\vec{a},\bar{b})$, and hence we may take $c_{\mathbf{L}} = 2M$.

For our final example, consider the case when $Y = \{0,1\}^k$, $A \subset [0,1]^k$ and $\mathbf{L}$ is the cross entropy loss

$$\mathbf{L}(\bar{y},\vec{a}) = -\sum_{i=1}^{k} \left(y_i \mathbf{ln} a_i + (1-y_i)\mathbf{ln}(1-a_i)\right).$$

As discussed in section 1.1.3, this is the log likelihood loss for the regression problem in which the action $\vec{a}$ represents a vector of probabilities for independent Bernoulli variables, and the outcome $\bar{y}$ gives the observed values of these variables. This loss is bounded if we restrict the probabilities in $\vec{a}$ to be between $B$ and $1-B$ for some $0 < B \leq 1/2$. In this case

$$\begin{aligned}
\rho_{\mathbf{L}}(\vec{a},\bar{b}) &= sup_{\bar{y}\in Y} \left| \sum_{i=1}^{k} \left( y_i \mathbf{ln}\frac{b_i}{a_i} + (1-y_i)\mathbf{ln}\frac{(1-b_i)}{(1-a_i)} \right) \right| \\
&\leq \sum_{i=1}^{k} |\mathbf{ln}\frac{b_i}{a_i}| + \sum_{i=1}^{k} |\mathbf{ln}\frac{(1-b_i)}{(1-a_i)}| \\
&\leq \frac{2}{B}\sum_{i=1}^{k} |a_i - b_i|.
\end{aligned}$$

The latter inequality follows from the fact that for $x, y > 0$,

$$|\mathbf{ln}\frac{x}{y}| = \mathbf{ln}\frac{\max(x,y)}{\min(x,y)} \leq \frac{\max(x,y)}{\min(x,y)} - 1 = \frac{|x-y|}{\min(x,y)}.$$

Thus in this case we may take $c_{\mathbf{L}} = 2k/B$.

We now turn to the task of obtaining an upper bound on the capacity $\mathcal{C}(\epsilon, \mathcal{H}, d_{L^1})$ when the decision rules in $\mathcal{H}$ map into a decision space $A \subset \Re^k$, and in particular, when these decision rules are represented by neural networks. When $k = 1$, i.e. the neural net has only one output, the decision rule space $\mathcal{H}$ is a family of real valued functions and $d_{L^1}(a,b) = |a-b|$ for $a, b \in A$. In this case we can sometimes apply methods based on the work of Vapnik and Chervonenkis [128] and Pollard [104, 106] to get bounds on the capacity. Vapnik's method for doing this is described in [129] and his chapter. Another way to do this, based on Pollard's notion of the *pseudo dimension* of $\mathcal{H}$, denoted $\mathbf{dim_C}(\mathcal{H})$, is developed in section 8.2 in the appendix. Both methods are extensions of basic method of Vapnik and Chervonenkis method for $\{0,1\}$-valued functions, using the notion of the VC dimension [130], which has already seen many applications in machine learning [128, 24]. Using the results on the pseudo dimension derived in the appendix, we get the following result. (Here $\mathbf{dim_C}$ denotes the pseudo dimension.)

**Theorem 5** *Let $\mathcal{H}$ be a family of functions from $X$ into $A = [0, M]$. Assume $\mathbf{dim_C}(\mathcal{H}) = d$ for some $1 \leq d < \infty$.*

1. *For all $0 < \epsilon \leq M$,*
$$\mathcal{C}(\epsilon, \mathcal{H}, d_{L^1}) < 2\left(\frac{2eM}{\epsilon}\mathbf{ln}\frac{2eM}{\epsilon}\right)^d.$$

2. $\overline{\mathbf{dim}}(\mathcal{H}) \leq \mathbf{dim_C}(\mathcal{H})$.

**Proof.** Let $P$ be any probability measure on $X$. Then by Theorems 7 and 10 in sections 8.1 and 8.2 of the appendix,

$$\mathcal{N}(\epsilon, \mathcal{H}, d_{L^1(P)}) \leq \mathcal{M}(\epsilon, \mathcal{H}, d_{L^1(P)}) < 2\left(\frac{2eM}{\epsilon}\mathbf{ln}\frac{2eM}{\epsilon}\right)^d.$$

(Theorem 10 is applied with $Z = X$ and $\mathbf{F} = \mathcal{H}$.) This gives (1.), and (2.) follows easily from (1.). $\square$

In the general case, where $A \subset \Re^k$ for $k > 1$, we can apply the methods from the previous section, in addition to the pseudo dimension methods from section 8.2, to obtain bounds on $\mathcal{C}(\epsilon, \mathcal{H}, d_{L^1})$. We illustrate this for the case when $\mathcal{H}$ is the class of decision rules represented by a feedforward neural network.

A feedforward neural network is defined as a directed acyclic graph in which the incoming edges to each node (or *unit*) are ordered and each incoming edge can carry a real number representing the *activation* on that edge. We will assume that all activations are restricted to the interval $[c_0, c_1]$ for some constants $c_0 < c_1$. The units are divided into *input units*, which have no incoming edges from other units and serve as input ports for the network (their activations are determined by these external inputs), and *computation units*, which have incoming edges from other units and compute an activation based on the activations on these incoming edges. After an activation has been determined, this activation is placed on the outgoing edges of the unit. Computation units with no outgoing edges are called *output units* and serve as output ports for the network. Computation units that are not output units are called *hidden units.* The network as a whole computes a function that maps from vectors of activation values in its input units to vectors of activation values in its output units by composing the functions computed by its computation units in the obvious way.

The action of a computation unit with $n$ incoming edges can be specified by a function $f$ from $[c_0, c_1]^n$ into $[c_0, c_1]$, where $f(\bar{x})$ is the resulting activation of the unit when the activations of its incoming edges are given be the vector $\bar{x} \in [c_0, c_1]^n$. In the nets we consider, the function $f$ is defined by

$$f(\bar{x}) = \sigma\left(\mu(\phi_1(\bar{x}), \ldots, \phi_k(\bar{x})) + \theta + \sum_{j=1}^{k} w_j \phi_j(\bar{x})\right),$$

where the $w_j$'s are adjustable real *weights*, $\theta$ is an adjustable real *bias*, $\phi_1, \ldots, \phi_k$ are fixed real-valued functions which we call the *input transformers*, $\mu : \Re^k \to \Re$ is a fixed function which we call the *global modifier*, and $\sigma : \Re \to [c_0, c_1]$ is a fixed non-increasing or non-decreasing function which we call the *squashing function*. Different units can have different modifiers, transformers and squashing functions. We say that the function $f$ computed by a given computation unit with $n$ incoming edges has *Lipschitz bound* $b$ if for any $\bar{x}, \bar{y} \in [c_0, c_1]^n$, $|f(\bar{x}) - f(\bar{y})| \leq b d_{L^1}(\bar{x}, \bar{y})$.

We give a few examples to illustrate the flexibility of this model at the level of the individual computation unit. First assume that $k = n$, i.e. the number of input transformers is the same as the number of inputs, and that each input transformer simply extracts a component of the input, i.e. $\phi_j(\bar{x}) = x_j$, $1 \leq j \leq n$. In this case, which is the standard case for most neural net research, the overall input transformation is just the identity map and can be ignored. In this standard case, if the global modifier $\mu = 0$ we get what is known as a quasi-linear unit [113]:

$$f(\bar{x}) = \sigma\left(\theta + \sum_{j=1}^{k} w_j x_j\right).$$

In the standard case, if $\mu(\bar{x}) = \sum_{j=1}^{n} x_j^2$ we get a unit that computes a function of the form

$$f(\bar{x}) = \sigma\left(\theta' + \sum_{j=1}^{n} (x_j - a_j)^2\right),$$

where $a_j = -w_j/2$ and $\theta' = \theta - \sum_{j=1}^{n} a_j^2$. This is similar to what is called a *radial basis* unit in the neural net literature [103, 86].

Now assume that $k = n$ but the input transformers take logs of the components of the inputs, i.e. $\phi_j(\bar{x}) = \mathbf{log} x_j$. (Here we assume $c_0 > 0$.) Let $\mu = 0$ and change the squashing function $\sigma$ to $\sigma'$, where $\sigma'(x) = \sigma(e^x)$. Then

$$f(\bar{x}) = \sigma'\left(\theta + \sum_{j=1}^{n} w_j \mathbf{log} x_j\right) = \sigma\left(e^{\theta} \prod_{j=1}^{n} x_j^{w_j}\right),$$

giving what is commonly known as a *product unit* [38].

We define a feedforward *architecture* as a feedforward net with unspecified weights and biases, i.e. each computation unit has a fixed global modifier, a fixed squashing function and fixed input transformers, but it has variable weights and a variable bias. We say a unit is *at depth $j$* in an architecture if the longest (directed) path from an input unit to that unit has $j$ edges. Thus all input units are at depth 0, all computation units that have incoming edges only from input units are at depth 1, all computation units that have incoming edges only from input units and computation units at depth 1 are at depth 2, etc. The *depth of the architecture* is the depth of the deepest unit in it.

We can bound the capacity of the decision rule space represented by a feedforward architecture as follows.

**Theorem 6** *Let $\mathcal{A}$ be a feedforward architecture as above with $n \geq 1$ input units, $k \geq 1$ output units, and depth $d \geq 1$. Let $W$ be the total number of adjustable weights and biases in $\mathcal{A}$. Assume $b_j \geq 1$ for $2 \leq j \leq d$, and let $\mathcal{H}$ be all functions from $[c_0, c_1]^n$ into $[c_0, c_1]^k$ representable on $\mathcal{A}$ by setting the adjustable weights and biases such that for all $j$, $2 \leq j \leq d$, the average of the Lipschitz bounds of the functions computed by computation units at depth $j$ is at most $b_j$. Then for all $0 < \epsilon \leq c_2 - c_1$,*

$$\mathcal{C}(\epsilon, \mathcal{H}, d_{L^1}) \leq \left(\frac{2e(c_2 - c_1)d\prod_{l=2}^{d} b_l}{\epsilon}\right)^{2W}.$$

□

**Proof.** For each $j$, $0 \leq j \leq d$, let $n_j$ be the number of units at depth $j$ in the architecture $\mathcal{A}$. For each $j$, $0 \leq j \leq d-1$, let $l_j = \sum_{i=0}^{j} n_j$, and let $l_d = n_d = k$. For each $j$, $1 \leq j \leq d+1$, let $X_j = [c_1, c_2]^{l_{j-1}}$. Then for each $j$, $1 \leq j \leq d$, we can define the family $\mathcal{H}_j$ of functions from $X_j$ into $X_{j+1}$ in the following manner.

First assume $j < d$. Let $u_1, \ldots, u_{n_j}$ be an enumeration of the computation units at depth $j$ and $f_1, \ldots, f_{n_j}$ be functions such that $f_i$ can be represented by $u_i$, $1 \leq i \leq n_j$, and the average Lipschitz bound on the $f_i$s is at most $b_j$. Let $h_j$ be the free product of $f_1, \ldots, f_{n_j}$ and $l_{j-1}$ copies of the identity function on $[c_1, c_2]$. Thus $h_j : X_j \to X_{j+1}$. The function $h_j$ represents a mapping from the sequence of all activations of units at depth at most $j-1$ to the sequence of all activations of units at depth at most $j$, where the activations at depth at most $j-1$ are unaltered, and the new activations, i.e. those at depth $j$, are calculated by $f_1, \ldots, f_{n_j}$. The family $\mathcal{H}_j$ consists of all functions $h_j$ obtained in this manner, by varying the weights and biases in the units $u_1, \ldots, u_{n_j}$ at depth $j$ in such a manner that the Lipschitz constraint is satisfied.

When $j = d$, no subsequent calculations will be performed so we no longer need to preserve the activations of shallower units. Hence, we omit the identity function components in each $h_d \in \mathcal{H}_d$. Otherwise the definition of $\mathcal{H}_d$ is the same as that for $\mathcal{H}_j$, where $j < d$.

It is clear that the class $\mathcal{H}$ in the statement of the theorem can be represented as the class of compositions of functions from classes $\mathcal{H}_1, \ldots, \mathcal{H}_d$. Since the identity function has Lipshitz bound $1 \leq b_j$, the average Lipschitz bound on the components of each function $h_j \in \mathcal{H}_j$ is at most $b_j$. It is easily verified that a free product function is Lipschitz bounded by the average of the Lipschitz bounds on its component functions. Hence by assumption, $b_j$ is a uniform Lipschitz bound on $\mathcal{H}_j$, $2 \leq j \leq d$. For each $j$, $1 \leq j \leq d$, let $a_j = \prod_{l=j+1}^{d} b_l$ and $\epsilon_j = \frac{\epsilon}{d a_j}$. Since $\epsilon < c_2 - c_1$ and $a_j \geq 1$, $\epsilon_j \leq c_2 - c_1$. Let $\rho_j$ be the $d_{L^1}$ metric on $X_j$, $1 \leq j \leq d+1$. Then by Lemma 4, part (1),

$$\mathcal{C}(\epsilon, \mathcal{H}, d_{L^1}) \leq \prod_{j=1}^{d} \mathcal{C}(\epsilon_j, \mathcal{H}_j, \rho_{j+1}).$$

For each $j$, $\mathcal{H}_j$ is contained in the free product of $l_j$ function classes. Each class $\mathcal{F}$ in this product is either the trivial class containing only the identity function, or is a finite dimensional vector space of real-valued functions, summed with a fixed modifier and then composed with a non-increasing or non-decreasing squashing function. In the latter case, the dimension $N$ of this vector space is the number of free parameters associated with the corresponding computation unit, i.e. the number of weights plus one (for the adjustable bias). Hence by Theorems 8 and 9 in section 8.2 of the appendix, the pseudo dimension $\mathbf{dim_C}(\mathcal{F}) \leq N$. Thus by Theorem 5 above,

$$\mathcal{C}(\epsilon_j, \mathcal{F}, d_{L^1}) \leq 2\left(\frac{2e(c_2 - c_1)}{\epsilon_j} \mathbf{ln} \frac{2e(c_2 - c_1)}{\epsilon_j}\right)^N \leq \left(\frac{2e(c_2 - c_1)}{\epsilon_j}\right)^{2N},$$

since $2\mathbf{ln}x < x$ and $N \geq 1$. Since the capacity of a class with only one function is 1, it follows from Lemma 3 part (1) that

$$\mathcal{C}(\epsilon_j, \mathcal{H}_j, \rho_{j+1}) \leq \left(\frac{2e(c_2 - c_1)}{\epsilon_j}\right)^{2W_j},$$

where $W_j$ is the total number of weights and biases of all computation nodes at depth $j$. Multiplying these bounds over all $j$, it follows that

$$\mathcal{C}(\epsilon, \mathcal{H}, d_{L^1}) \leq \prod_{j=1}^{d} \left(\frac{2e(c_2 - c_1)}{\epsilon_j}\right)^{2W_j}$$

$$
\begin{aligned}
&= \prod_{j=1}^{d}\left(\frac{2e(c_2-c_1)d\prod_{l=j+1}^{d} b_l}{\epsilon}\right)^{2W_j} \\
&\le \left(\frac{2e(c_2-c_1)d\prod_{l=2}^{d} b_l}{\epsilon}\right)^{2W}.
\end{aligned}
$$

□

**Corollary 2** *Let $n$, $k$, $\mathcal{H}$, $W$, $d$ and $b_2,\ldots,b_d$ be as in the previous theorem. Let $X$ be the instance space $[c_1,c_2]^n$, $A$ be the decision space $[c_1,c_2]^k$, and $Y$ be any outcome space. Let $\mathbf{L}: Y\times A\to[0,M]$ be a loss function and $c_{\mathbf{L}}$ be a constant such that $\rho_{\mathbf{L}}(\vec{a},\bar{b})\le c_{\mathbf{L}} d_{L^1}(\vec{a},\bar{b})$ for all $\vec{a},\bar{b}\in A$. Let $m\ge 1$, $0<\nu\le 8(c_2-c_1)$, $0<\alpha<1$, and $P$ be any probability distribution on $Z=X\times Y$. Let $\bar{z}$ be generated by $m$ independent random draws from $Z$ according to $P$.*

1. *Then*

$$
\mathbf{Pr}\left(\exists f\in\mathcal{H}: d_\nu(\widehat{\mathbf{r}}_{f,\mathbf{L}}(\vec{z}),\mathbf{r}_{f,\mathbf{L}}(P))>\alpha\right)\le 4\left(\frac{c_{\mathbf{L}}16e(c_2-c_1)d\prod_{l=2}^{d} b_l}{\alpha\nu}\right)^{2W} e^{-\alpha^2\nu m/8M}.
$$

2. *Assume that for each computation unit at depth 2 and above the number of weights is at most $W_{max}$, no weight is allowed to have absolute value greater than $\beta$, the input transformers are identity functions, the global modifier has Lipschitz bound at most $r$, and the squashing function has Lipschitz bound at most $s$, where $s(\beta W_{max}+r)\ge 1$. Then for any $0<\delta<1$, the probability in (1.) is less than $\delta$ for sample size*

$$
m = O\left(\frac{M}{\alpha^2\nu}\left(W\left(\mathbf{log}\frac{c_{\mathbf{L}}(c_2-c_1)}{\alpha\nu}+d\,\mathbf{log}(s(\beta W_{max}+r))\right)+\mathbf{log}\frac{1}{\delta}\right)\right).
$$

**Proof.** Let $\epsilon=\alpha\nu/8\le c_2-c_1$. It can be verified that $l_{\mathcal{H}}$ is permissible for the decision rule space $\mathcal{H}$. Hence, using Theorem 4 and Theorem 6,

$$
\begin{aligned}
\mathbf{Pr}\left(\exists f\in\mathcal{H}: d_\nu(\widehat{\mathbf{r}}_{f,\mathbf{L}}(\vec{z}),\mathbf{r}_{f,\mathbf{L}}(P))>\alpha\right) &\le 4\mathcal{C}(\alpha\nu/8,\mathcal{H},\rho_{\mathbf{L}})e^{-\alpha^2\nu m/8M} \\
&\le 4\mathcal{C}(\alpha\nu/8c_{\mathbf{L}},\mathcal{H},d_{L^1})e^{-\alpha^2\nu m/8M} \\
&\le 4\left(\frac{c_{\mathbf{L}}16e(c_2-c_1)d\prod_{l=2}^{d} b_l}{\alpha\nu}\right)^{2W} e^{-\alpha^2\nu m/8M}.
\end{aligned}
$$

For the second bound, it is readily verified that the $L^1$ Lipschitz bound for a linear function defined by $W_{max}$ weights and a bias is no more than $W_{max}$ times the largest absolute value of any weight. Furthermore, the Lipschitz bound for the sum of two functions is no more than the sum of the Lipschitz bounds on the individual functions, and the Lipschitz bound for the composition of two functions is no more than the product of the Lipschitz bounds on the individual functions. Thus if the input transformers are identity functions, the global modifier and squashing function have Lipschitz bounds $r$ and $s$ respectively and no weight is allowed to have absolute value greater than $\beta$, then the Lipschitz bound for a computation unit is at most $s(\beta W_{max} + r)$. If this holds for all units at depth 2 and above, may take $b_j = s(\beta W_{max} + r)$ for all $j \geq 2$ in the first bound. Solving for $m$, this gives the order-of-magnitude estimate of the second bound. □

We give the constants in the upper bound of part (1.) the above theorem only to show that they are not outlandishly large. We do not mean to suggest that the bound is tight. At present we cannot even verify that the asymptotic bound of part (2.) is tight. In particular, we cannot show that the dependence on the Lipschitz bounds is necessary. Evidence that it may not be necessary comes from the analysis of the case where the squashing function $\sigma$ is a sharp threshold function, i.e. $\sigma(x) = sign(x)$. Corollary 2 does not apply in this case, because the jump in $\sigma$ prevents us from obtaining a Lipschitz bound on the computation units. As we let a smooth $\sigma$ approach the *sign* function, its slope increases without limit, and thus the bound given in Corollary 2 degenerates. Nevertheless, using the techniques in [17] it can be shown that results similar to Corollary 2 hold in this case, except that no Lipschitz bounds are required, and a bound on the sample size is

$$O\left(\frac{1}{\alpha^2\nu}\left(W\mathbf{log}\frac{N}{\alpha\nu} + \mathbf{log}\frac{1}{\delta}\right)\right),$$

where $N$ is the total number of computation units in the net. Details are given in [54].

Despite the uncertainty about the need for the Lipschitz bounds, the result does give some indication of the maximum training sample size that will be needed for many popular network configurations. For example, if the squashing function is chosen as $\sigma(x) = 1/(1+e^{-x/T})$ for some *temperature* $T > 0$ then it can be shown that the Lipschitz bound $s$ for $\sigma$ is $1/4T$. When the modifier $\mu \equiv 0$, then $r = 0$. Thus in this case the term $d\mathbf{log}(s(\beta W_{max} + r))$ in the bound of Corollary 2 becomes $d\mathbf{log}(\beta W_{max}/T)$. If the maximum weight $\beta$, the temperature $T$ and the depth $d$ are constants, along with $c_2 - c_1$, $M$, and $c_{\mathbf{L}}$, then the asymptotic bound of the theorem becomes

$$O\left(\frac{1}{\alpha^2\nu}\left(W\mathbf{log}\frac{W_{max}}{\alpha\nu} + \mathbf{log}\frac{1}{\delta}\right)\right),$$

which is similar to the bound obtained in [17].

It should also be noted that Corollary 2 does have the feature that no Lipschitz bounds are required on the computation units at depth one. Thus if all computation units are at depth one, i.e. there are no hidden units, then no Lipschitz units are required at all. If the architecture has only one layer of hidden units at depth one and a single output unit at depth two, as is quite common, then Lipschitz bounds are required only on the output unit. This means that the weights and biases associated with the hidden units do not need to be bounded in order to get the rates of uniform convergence given by Corollary 2, as they would, for example, if the methods given in White's chapter were used to obtain a result of this type.

For an example of the above, consider networks that implement generalized radial basis[15] functions, as described in [103]. These networks have one layer of hidden units at depth 1 and one output unit at depth 2. The structure of the hidden units is as described in the example above: the input transformers are identity functions, the modifier is $\sum_{i=1}^{n} x_i^2$ and the squashing function is usually a smooth decreasing function. The output unit simply computes a weighted sum, so for this unit the modifier is the 0 function and the squashing function is the identity. Since this is the only unit at depth 2 and above, we require a Lipschitz bound only for this unit. If $\beta$ is a bound on the maximum weight coming into the output unit, and $W_{max}$ is the number of units in the hidden layer, then the term $d\mathbf{log}(s(\beta W_{max} + r))$ in the above bound becomes $\mathbf{log}(\beta W_{max})$. Again, fixing $\beta, c_2 - c_1$, $M$, and $c_{\mathbf{L}}$ gives the same sample size bound,

$$O\left(\frac{1}{\alpha^2 \nu}\left(W \mathbf{log}\frac{W_{max}}{\alpha\nu} + \mathbf{log}\frac{1}{\delta}\right)\right),$$

similar to that obtained in [17].

Since $W$ appears to be the dominant factor in these bounds, apart from the accuracy parameters $\alpha$ and $\nu$, these bounds support the conventional wisdom that the training set size should be primarily related to the number of adjustable parameters in the net. They also support the notion that this relationship between appropriate training size and the number of parameters is nearly linear, at least in the worst case. Further work is needed to sharpen these relationships (see e.g. the lower

[15]The computation units in the network of radial basis functions described here are quite primitive in that they have no adjustable multiplicative parameter included in their basic radial distance calculation. Such parameters would be needed to do any reasonable type of kernel based density estimation (see e.g. [34]). These parameters can be simulated by inserting another layer of computation units between the inputs and the layer described here. Alternately, the analysis can also be done directly for adjustable kernel units. This cleaner approach is detailed in [105].

bounds obtained in [17]).

## 6 Conclusion

We have extended the PAC learning model to a more general decision theoretic framework so that it addresses many of the concerns raised by machine learning practitioners, and also introduced a number of new theoretical tools. Here we concentrate on applications of the extended model to the problem of obtaining upper bounds on sufficient training sample size. Further work will be required to obtain lower bounds on sample size needed, and to determine the computational complexity of finding decision rules with near minimal empirical risk. Some promising results along these lines are given in [66]. However, even granting that such results can be obtained, the extended model still has a number of shortcomings in its present form. Some of these can be easily remedied, others may be more problematic.

First, we define the model only for a fixed decision rule space $\mathcal{H}$. The model should be extended to learning problems on a sequence of decision rule spaces $\{\mathcal{H}_n : n \geq 1\}$, where $\mathcal{H}_n$ is a decision rule space on an $n$-attribute domain $X_n$ (e.g. $[0,1]^n$), and to families of decision rule spaces of different "complexities" on a fixed domain [64] [24] [55], so that tradeoffs between decision rule complexity and empirical risk can be addressed. The former extension is easy, the latter more involved. One approach to the latter problem is via Vapnik's principle of structural risk minimization [129] (see also [33]). Other approaches include the MDL (see e.g. [14]), regularization (see e.g. [103]), and more general Bayesian methods (see e.g. [20]).

Second, the constants in the upper bounds are still too large to give sample size estimates that are useful in practice. It may be difficult to improve them to the point where the results are directly usable in applied work. Thus even with matching asymptotic lower bounds, practitioners may still need to rely at least in part on empirically derived sample size bounds. It is possible that the Bayesian viewpoint may yield better tools for calculating sample complexities. Support for this belief is given in [29, 28, 56, 97]. However, necessary sample size estimates for decision rule spaces as general as those studied from the minimax perspective using uniform convergence have not yet been tackled from the Bayesian perspective. Vapnik's recent work gives an alternative, related approach (see Vapnik's chapter).

Finally, many other issues would need to be considered in a complete treatment of the problem of overfitting, including distribution specific bounds on sample complexity (Theorem 2 is actually distribution specific, since the random covering numbers are distribution specific, yet we only apply it here in a distribution independent setting), decision rule spaces with infinite pseudo and metric dimensions (these include various classes of "smooth" functions and their relatives, see [36], Chapter 7 and [107]) and non i.i.d. sources of examples (see [137, 93] and White's chapter). Despite these shortcomings, we feel that the theory we give here provides useful insights into the nature of the problem of overfitting in learning, and because of its generality will be a useful starting point for further research in this area.

# 7 Acknowledgements

I would like to thank Dana Angluin, David Pollard and Phil Long for their careful criticisms of an earlier draft of this paper, and their numerous suggestions for improvements. I also thank Naoki Abe, Anselm Blumer, Richard Dudley, and Michael Kearns for helpful comments on earlier drafts. I would also like to thank Ron Rivest, David Rumelhart, Andrzej Ehrenfeucht and Nick Littlestone for stimulating discussions on these topics.

# 8 Appendix

## 8.1 Metric spaces, covering numbers and metric dimension

A *pseudo metric* on a set $S$ is a function $\rho$ from $S \times S$ into $\Re^+$ such that for all $x, y, z \in S$, $x = y \Rightarrow \rho(x,y) = 0$, $\rho(x,y) = \rho(y,x)$ (symmetry), and $\rho(x,z) \leq \rho(x,y) + \rho(y,z)$ (triangle inequality). If in addition $\rho(x,y) = 0 \Rightarrow x = y$, then $\rho$ is a *metric.* $(S,\rho)$ is a *(pseudo) metric space.* $(S,\rho)$ is *complete* if every Cauchy sequence of points in $S$ converges to a point in $S$; $(S,\rho)$ is *separable* if it contains a countable dense subset, i.e. a countable subset $A$ such that for every $x \in X$ and $\epsilon > 0$ there exists $a \in A$ with $\rho(x,a) < \epsilon$. If $\rho(x,y) = 1 \Leftrightarrow x \neq y$ then $\rho$ is called the *discrete metric.*

The *diameter* of a set $T \subseteq S$ is $\sup\{\rho(x,y) : x, y \in T\}$. If the diameter of $T$ is finite the we say that $T$ is *bounded.* For any $\epsilon > 0$, an $\epsilon$*-cover* for $T$ is a finite set $N \subseteq S$ (not necessarily contained in $T$) such that for all $x \in T$ there is a $y \in N$ with $\rho(x,y) \leq \epsilon$. If $T$ has a (finite) $\epsilon$-cover for all $\epsilon > 0$ then $T$ is *totally bounded.* (Note that this implies that $(T,\rho)$ is separable and bounded.) In this case the function $\mathcal{N}(\epsilon, T, \rho)$ denotes the size of the smallest $\epsilon$-cover for $T$ (w.r.t. the space $S$ and the (pseudo) metric $\rho$). We refer to $\mathcal{N}(\epsilon, T, \rho)$ as a *covering number.* A set $R \subseteq T$ is $\epsilon$*-separated* if for all distinct $x, y \in R$, $\rho(x,y) > \epsilon$. We denote by $\mathcal{M}(\epsilon, T, \rho)$ the size of the largest $\epsilon$-separated subset of $T$. We refer to $\mathcal{M}(\epsilon, T, \rho)$ as a *packing number*. The third argument to $\mathcal{N}$ and $\mathcal{M}$ will be omitted when the metric $\rho$ is clear from the context.

The following inequalities are easily verified (see e.g. [68]):

**Theorem 7** *If $T$ is a totally bounded subset of the (pseudo) metric space $(S, \rho)$ then for any $\epsilon > 0$,*

$$\mathcal{M}(2\epsilon, T, \rho) \le \mathcal{N}(\epsilon, T, \rho) \le \mathcal{M}(\epsilon, T, \rho).$$

Hence both these measures of boundedness, by covering number and by packing number, are equivalent to within a factor of 2 of $\epsilon$. Following [68] we define the *upper metric dimension* of a (pseudo) metric space $(S, \rho)$ by

$$\overline{\mathbf{dim}}(S) = \mathbf{limsup}_{\epsilon \to 0} \frac{\mathbf{log} \mathcal{N}(\epsilon, S, \rho)}{\mathbf{log}(1/\epsilon)}.$$

The *lower metric dimension*, denoted by $\underline{\mathbf{dim}}$, of a (pseudo) metric space $(S, \rho)$ is defined similarly using **liminf**. When $\overline{\mathbf{dim}}(S) = \underline{\mathbf{dim}}(S)$, then this quantity is denoted $\mathbf{dim}(S)$, and referred to simply as the *metric dimension* of $(S, \rho)$. This quantity has also been called the *fractal dimension* [41] and the *capacity dimension* [42]. A very lucid and intuitive treatment is given in [83].

## 8.2 Pseudo dimension of classes of real-valued functions

In this section we will look at one way that bounds on the covering numbers appearing in Theorem 2 can be obtained. This technique, due to Pollard [104], who extended methods from [35], is based on certain intuitions from combinatorial geometry. It generalizes the techniques based on the Vapnik-Chervonenkis dimension used in [24], which apply only to $\{0, 1\}$-valued functions. We begin by establishing some basic notation.

**Definition 7** *For $x \in \Re$, let $sign(x) = 1$ if $x > 0$ else $sign(x) = 0$. For $\bar{x} = (x_1, \ldots, x_d) \in \Re^d$, let $sign(\bar{x}) = (sign(x_1), \ldots, sign(x_d))$ and for $T \subset \Re^d$ let $sign(T) = \{sign(\bar{x}) : \bar{x} \in T\}$. For any Boolean vector $\bar{b} = (b_1, \ldots, b_d)$, $\{\bar{x} \in \Re^d : sign(\bar{x}) = \bar{b}\}$ is called the $\bar{b}$-*orthant *of $\Re^d$, where we have, somewhat arbitrarily, included points with value zero for a particular coordinate in the associated lower orthant. Thus $sign(T)$ denotes the set of orthants intersected by $T$. For any $T \subset \Re^d$, and $\bar{x} \in \Re^d$, let $T + \bar{x} = \{\bar{y} + \bar{x} : \bar{y} \in T\}$, i.e. the translation of $T$ obtained by adding the vector $\bar{x}$. We* *say that $T$ is* full *if there exists $\bar{x} \in \Re^d$ such that $sign(T + \bar{x}) = \{0,1\}^d$, i.e. if there exists some translation of $T$ that intersects all $2^d$ orthants of $\Re^d$.*

The following result is well known and can be proved in a variety of ways. For example, it follows easily from well known bounds on the number of cells in arrangements of hyperplanes (see e.g. [39]). We give an elementary proof using a technique from [35].

**Lemma 5** *No hyperplane in $\Re^d$ intersects all orthants of $\Re^d$.*

**Proof.** Let $T$ be a hyperplane in $\Re^d$. Choose a vector $\bar{x} \in \Re^d$ as follows. If $T$ includes the origin, then let $\bar{x}$ be any vector that is orthogonal to $T$ and has at least one strictly negative coordinate. (For any nonzero orthogonal vector $\bar{x}$, if $\bar{x}$ doesn't have a negative coordinate then $-\bar{x}$ does.) Otherwise let $\bar{x}$ be the (nonzero) vector in $T$ on the line perpendicular to $T$ that passes through the origin. To complete the proof, we show that for all $\bar{y} \in T$, $sign(\bar{y}) \neq \vec{1} - sign(\bar{x})$, where $\vec{1}$ denotes the all 1's vector.

Suppose to the contrary that $sign(\bar{y}) = \vec{1} - sign(\bar{x})$ for some $\bar{y} \in T$. This implies that the inner product $\sum_{i=1}^{d} x_i y_i$ is non positive, and is in fact strictly negative if either $\bar{x}$ or $\bar{y}$ contain a strictly negative coordinate. However, by our choice of $\bar{x}$, either $\bar{x}$ is orthogonal to $\bar{y}$ and contains a strictly negative coordinate, giving an immediate contradiction, or $\bar{x}$ is non-zero and $\bar{x}$ is orthogonal to $\bar{y} - \bar{x}$. In this last case,

$$\sum_{i=1}^{d} x_i y_i = \sum_{i=1}^{d} x_i^2,$$

which is again a contradiction, since the left side is non-positive while the right side is strictly positive. $\Box$

It follows from this lemma that if $T$ is contained in a hyperplane of $\Re^d$ then $T$ is not full.

**Definition 8** *Let $\mathbf{F}$ be a family of functions from a set $Z$ into $\Re$. For any sequence $\bar{z} = (z_1, \ldots, z_d)$ of points in $Z$, let $\mathbf{F}_{|\bar{z}} = \{(\mathbf{f}(z_1), \ldots, \mathbf{f}(z_d)) : \mathbf{f} \in \mathbf{F}\}$. If $\mathbf{F}_{|\bar{z}}$ is full then we say that $\bar{z}$ is* shattered *by $\mathbf{F}$. The* pseudo dimension *of $\mathbf{F}$, denoted $\mathbf{dim_C(F)}$, is the largest $d$ such that there exists a sequence of $d$ points in $Z$ that is shattered by $\mathbf{F}$. If arbitrarily long finite sequences are shattered, then $\mathbf{dim_C(F)}$ is infinite.*

It is clear that when $\mathbf{F}$ is a set of $\{0,1\}$-valued functions then for any sequence $\bar{z}$ of $d$ points in $Z$, $\mathbf{F}_{|\bar{z}}$ is full if and only if $\mathbf{F}_{|\bar{z}} = \{0,1\}^d$. Thus in this case $\mathbf{dim_C(F)}$ is the length $d$ of the longest sequence of points $\bar{z}$ such that $\mathbf{F}_{|\bar{z}} = \{0,1\}^d$. This is the definition of the Vapnik-Chervonenkis dimension of a class $\mathbf{F}$ of $\{0,1\}$-valued functions [128][59][24]. Thus the pseudo dimension generalizes the Vapnik-Chervonenkis dimension to arbitrary classes of real-valued functions.

The pseudo dimension also generalizes the algebraic notion of the dimension of a vector space of real-valued functions [35].

**Theorem 8** *(Dudley) Let* $\mathbf{F}$ *be a* $d$*-dimensional vector space of functions from a set* $Z$ *into* $\Re$. *Then* $\mathbf{dim_C}(\mathbf{F}) = d$.

**Proof.** Fix any sequence $\bar{z} = (z_1, \ldots, z_{d+1})$ of points in $Z$. For any $\mathbf{f} \in \mathbf{F}$ let $\Psi(\mathbf{f}) = (\mathbf{f}(z_1), \ldots, \mathbf{f}(z_{d+1}))$. Then $\Psi$ is a linear mapping from $\mathbf{F}$ into $\Re^{d+1}$, and the image of $\Psi$ is $\mathbf{F}_{|\bar{z}}$. Since $\mathbf{F}$ is a vector space of dimension $d$, this implies that $\mathbf{F}_{|\bar{z}}$ is a subspace of $\Re^{d+1}$ of dimension at most $d$. Hence by Lemma 5, $\mathbf{F}_{|\bar{z}}$ is not full. This implies $\mathbf{dim_C}(\mathbf{F}) \leq d$. On the other hand, if $\mathbf{F}$ is a $d$-dimensional vector space of real-valued functions on $Z$, then there exists a sequence $\bar{z}$ of $d$ points in $Z$ such that $\mathbf{F}_{|\bar{z}} = \Re^d$. Hence $\bar{z}$ is shattered, implying that $\mathbf{dim_C}(\mathbf{F}) \geq d$. $\square$

There are many other ways that the VC dimension can be generalized to real-valued functions [91] [90] [104] [129] [37] (see also Vapnik's chapter). Dudley [37] compares several such generalizations, albeit in a different context. The generalization we have proposed here, the pseudo dimension, is a minor variant of the notion used by Pollard in [104] to define classes of real-valued functions of polynomial discrimination, called VC-subgraph classes in [37]. The pseudo dimension will be used in the form defined above in Pollard's new book [106].

The pseudo dimension has a few invariance properties that are useful (see [106] for further results of this type).

**Theorem 9** *Let* $\mathbf{F}$ *be a family of functions from* $Z$ *into* $\Re$. *Fix any function* $\mathbf{g}$ *from* $Z$ *into* $\Re$ *and let* $\mathbf{G} = \{\mathbf{g} + \mathbf{f} : \mathbf{f} \in \mathbf{F}\}$. *Let* $I$ *be a real interval (possibly all of* $\Re$*) such that every function in* $\mathbf{F}$ *takes values only in* $I$. *Fix any nondecreasing (resp. nonincreasing) function* $h : I \to \Re$ *and let* $\mathbf{H} = \{h \circ \mathbf{f} : \mathbf{f} \in \mathbf{F}\}$, *where* $\circ$ *indicates function composition. Then*

1. *([136])* $\mathbf{dim_C}(\mathbf{G}) = \mathbf{dim_C}(\mathbf{F})$ *and*
2. *([94, 37])* $\mathbf{dim_C}(\mathbf{H}) \leq \mathbf{dim_C}(\mathbf{F})$, *with equality if* $h$ *is continuous and strictly increasing (resp. continuous and strictly decreasing).*

**Proof.** Part (1) follows directly from the fact that the notion of a set of points being full is invariant under translation. For part (2) it suffices to prove the results for $h$ nondecreasing and

$h$ continuous and strictly increasing. Let $\bar{z} = (z_1, \ldots, z_d)$ be such that $\mathbf{H}_{|\bar{z}}$ is full, i.e. such that $\mathbf{H}_{|\bar{z}} - \bar{x}$ intersects all $2^d$ orthants of $\Re^d$ for some vector $\bar{x} = (x_1, \ldots, x_d)$ in $\Re^d$. Then for every Boolean vector $\bar{b} \in \{0,1\}^d$ there exists a function $\mathbf{f}_{\bar{b}} \in \mathbf{F}$ such that for every $i$, $1 \leq i \leq d$, we have $h \circ \mathbf{f}_{\bar{b}}(z_i) > x_i$ if and only if the $i^{th}$ bit of $\bar{b}$ is 1. For each $i$, $1 \leq i \leq d$, let

$$u_i = min\{\mathbf{f}_{\bar{b}}(z_i) : \; the\ i^{th}\ bit\ of\ \bar{b}\ is\ 1\}$$

and

$$l_i = max\{\mathbf{f}_{\bar{b}}(z_i) : \; the\ i^{th}\ bit\ of\ \bar{b}\ is\ 0\}.$$

Since $h$ is nondecreasing, we have $u_i > l_i$ for each $i$. Let $r_i = (u_i + l_i)/2$ for each $i$ and $\vec{r} = (r_1, \ldots, r_d)$. Let $T = \{\mathbf{f}_{\bar{b}} : \bar{b} \in \{0,1\}^d\}$. Then clearly $T - \vec{r}$ intersects every orthant of $\Re^d$, so $T$ is full. Since $T \subset \mathbf{F}$, this implies that $\mathbf{F}_{|\bar{z}}$ is full, and hence $\mathbf{dim_C}(\mathbf{H}) \leq \mathbf{dim_C}(\mathbf{F})$. Equality follows when $h$ is continuous and strictly increasing since we obtain the class $\mathbf{F}$ from $\mathbf{H}$ by composing with $h^{-1}$. $\square$

By putting a probability measure on $Z$, we can view a class $\mathbf{F}$ of real-valued functions on $Z$ as a pseudo metric space. The distance between two functions is the integral of the absolute value of their difference, i.e. the $L^1$ distance, relative to the given measure. To make this work, we need to make some assumptions about the integrability of the functions in $\mathbf{F}$ under the given measure. Since we will be concerned only with families of functions taking values in a bounded range in this paper, this will cause no problems for us. For convenience, we choose this range to be $[0, M]$. For a more general treatment, see [104][36].

**Definition 9** *Let $\mathbf{F}$ be a class of functions from $Z$ into $[0, M]$, where $M > 0$, and $P$ be a probability measure on $Z$. Then $d_{L^1(D)}$ is the pseudo metric on $\mathbf{F}$ defined by*

$$d_{L^1(D)}(\mathbf{f}, \mathbf{g}) = \mathbf{E}(|\mathbf{f} - \mathbf{g}|) = \int_Z |\mathbf{f}(z) - \mathbf{g}(z)| dP(z) \;\; for\ all\ \mathbf{f}, \mathbf{g} \in \mathbf{F}.$$

Using techniques that go back to Dudley [35], Pollard has obtained a beautiful theorem bounding the metric dimension of $(\mathbf{F}, d_{L^1(D)})$ by $\mathbf{dim_C}(\mathbf{F})$ for any probability measure $P$ on $Z$. Actually this result is much stronger in that it gives explicit bounds on the packing numbers for $\mathbf{F}$ using $d_{L^1(D)}$ balls of radius $\epsilon$. Since packing numbers are closely related to covering numbers (Theorem 7 in section 8.1), these bounds can then be used with Theorem 2 to obtain uniform convergence results for empirical estimates of functions in $\mathbf{F}$. We now state a version of Pollard's result ([104], Lemma 25, p. 27) for the special case when $\mathbf{F}$ is a class of functions taking values in the interval $[0, M]$

with somewhat better bounds on the packing numbers.[16]

**Theorem 10** *(Pollard) Let* $\mathbf{F}$ *be a family of functions from a set* $Z$ *into* $[0, M]$*, where* $\mathbf{dim_C}(\mathbf{F}) = d$ *for some* $1 \leq d < \infty$*. Let* $P$ *be a probability measure on* $Z$*. Then for all* $0 < \epsilon \leq M$*,*

$$\mathcal{M}(\epsilon, \mathbf{F}, d_{L^1(D)}) < 2\left(\frac{2eM}{\epsilon}\mathbf{ln}\frac{2eM}{\epsilon}\right)^d.$$

The proof uses essentially the same techniques as Pollard's, and is given in [54]. Using our results on uniform convergence from section 3.2, we can now show the following.

**Theorem 11** *Let* $\mathbf{F}$ *be a permissible family of functions from a set* $Z$ *into* $[0, M]$ *with* $\mathbf{dim_C}(\mathbf{F}) = d$ *for some* $1 \leq d < \infty$*. Assume* $m \geq 1$*,* $0 < \nu \leq 8M$ *and* $0 < \alpha < 1$*. Let* $\bar{z}$ *be generated by* $m$ *independent draws according to any distribution on* $Z$*. Then*

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : d_\nu(\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}), \mathbf{E}(\mathbf{f})) > \alpha\right) \leq 8\left(\frac{16eM}{\alpha\nu}\mathbf{ln}\frac{16eM}{\alpha\nu}\right)^d e^{-\alpha^2\nu m/8M}.$$

*Moreover, for* $m \geq \frac{8M}{\alpha^2\nu}\left(2d\mathbf{ln}\frac{8eM}{\alpha\nu} + \mathbf{ln}\frac{8}{\delta}\right)$ *this probability is at most* $\delta$*.*

**Proof.** Let $\epsilon = \alpha\nu/8$. Since $\alpha < 1$ and $\nu \leq 8M$, $\epsilon \leq M$. For any sequence $\bar{z}$ of points in $Z$ there is a trivial isometry between $(\mathbf{F}_{|\bar{z}}, d_{L^1})$ and $(\mathbf{F}, d_{L^1(P_{\bar{z}})})$, where $P_{\bar{z}}$ is the empirical measure induced by $\bar{z}$, in which each set has measure equal to the fraction of the points in $\bar{z}$ it contains. Thus by Theorem 7 of section 8.1 and Theorem 10, we have

$$\mathcal{N}(\epsilon, \mathbf{F}_{|\bar{z}}, d_{L^1}) \leq \mathcal{M}(\epsilon, \mathbf{F}_{|\bar{z}}, d_{L^1}) \leq 2\left(\frac{2eM}{\epsilon}\mathbf{ln}\frac{2eM}{\epsilon}\right)^d,$$

for all $\bar{z} \in Z^{2m}$. Hence the given probability is at most

$$8\left(\frac{2eM}{\epsilon}\mathbf{ln}\frac{2eM}{\epsilon}\right)^d e^{-\alpha^2\nu m/8M} = 8\left(\frac{16eM}{\alpha\nu}\mathbf{ln}\frac{16eM}{\alpha\nu}\right)^d e^{-\alpha^2\nu m/8M}$$

by Theorem 3.

For the second result, setting the bound above equal to $\delta$ and solving for $m$ gives

$$m \geq \frac{8M}{\alpha^2\nu}\left(d\mathbf{ln}\left(\frac{16eM}{\alpha\nu}\mathbf{ln}\frac{16eM}{\alpha\nu}\right) + \mathbf{ln}\frac{8}{\delta}\right).$$

[16] Still better bounds, without the logarithmic factor that is present in the theorem quoted here, are given in [53].

It is easily verified that $\mathbf{ln}(a\mathbf{ln}a) < 2\mathbf{ln}(a/2)$ when $a \geq 5$, and from this the bound given in the second result follows. $\square$

**Corollary 3** *Under the same assumptions as above, for all* $0 < \epsilon \leq M$,

$$\mathbf{Pr}\left(\exists \mathbf{f} \in \mathbf{F} : |\widehat{\mathbf{E}}_{\bar{z}}(\mathbf{f}) - \mathbf{E}(\mathbf{f})| > \epsilon\right) \leq 8\left(\frac{32eM}{\epsilon}\mathbf{ln}\frac{32eM}{\epsilon}\right)^d e^{-\epsilon^2 m/64M^2}.$$

*Moreover, for* $m \geq \frac{64M^2}{\epsilon^2}\left(2d\mathbf{ln}\frac{16eM}{\epsilon} + \mathbf{ln}\frac{8}{\delta}\right)$ *this probability is at most* $\delta$.

**Proof.** This follows directly from the above result by setting $\nu = 2M$, $\alpha = \epsilon/4M$, and using property (3) of the $d_\nu$ metric, as in the proof of Corollary 1 in section 3.2. $\square$

## References


[1] N. Abe and M. Warmuth. On the computational complexity of approximating distributions by probabilistic automata. In *Proceedings of the 3nd Workshop on Computational Learning Theory*, pages 52–66. published by Morgan Kaufmann, 1990.

[2] K. Alexander. Rates of growth for weighted empirical processes. In *Proc. of Berkeley Conference in Honor of Jerzy Neyman and Jack Kiefer*, volume 2, pages 475–493, 1985.

[3] K. Alexander. Rates of growth and sample moduli for weighted empirical processes indexed by sets. *Probability Theory and Related Fields*, 75:379–423, 1987.

[4] J. Amsterdam. The valiant learning model: Extensions and assessment. Master's thesis, MIT Department of Electrical Engineering and Computer Science, Jan. 1988.

[5] D. Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75:87–106, Nov. 1987.

[6] D. Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.

[7] D. Angluin, M. Frazier, and L. Pitt. Learning conjunctions of Horn clauses. In *31th Annual IEEE Symposium on Foundations of Computer Science*, pages 186–192, 1990.

[8] D. Angluin, L. Hellerstein, and M. Karpinski. Learning read-once formulas with queries. *J. ACM*, 40:185–210, 1993.

[9] D. Angluin and M. Kharitonov. Why won't membership queries help? In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, pages 444–454, New Orleans, May 1991. ACM.

[10] D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

[11] D. Angluin and L. G. Valiant. Fast probabilistic algorithms for hamiltonian circuits and matchings. *Journal of Computer and System Sciences*, 18(2):155–193, Apr. 1979.

[12] M. Anthony and N. Biggs. *Computational Learning Theory.* Cambridge Tracts in Theoretical Computer Science (30). Cambridge University Press, 1992.

[13] A. Barron. Statistical properties of artificial neural networks. In *28th Conference on Decision and Control*, pages 280–285, 1989.

[14] A. Barron and T. Cover. Minimum complexity density estimation. *IEEE Transactions on Information Theory*, 37:1034–1054, 1991.

[15] A. G. Barto and P. Anandan. Pattern recognizing stochastic learning automata. *IEEE Trans. on Systems, Man and Cybernetics*, 15:360–374, 1985.

[16] E. Baum. When are k-nearest neighbor and back propogation accurate for feasible sized sets of examples. In *Snowbird conference on Neural Networks for Computing*, 1990. unpublished manuscript.

[17] E. Baum and D. Haussler. What size net gives valid generalization? *Neural Computation*, 1(1):151–160, 1989.

[18] G. M. Benedek and A. Itai. Learnability by fixed distributions. In *Proc. 1988 Workshop on Comp. Learning Theory*, pages 80–90, San Mateo, CA, 1988. Morgan Kaufmann.

[19] F. Bergadano and L. Saitta. On the error probabilty of boolean concept descriptions. In *Proceedings of the 1989 European Working Session on Learning*, pages 25–35, 1989.

[20] J. Berger. *Statistical Decision Theory and Bayesian Analysis.* Springer-Verlag, New York, 1985.

[21] P. Billingsley. *Probability and Measure.* Wiley, New York, 1986.

[22] A. Blum and R. L. Rivest. Training a three-neuron neural net is NP-Complete. In *Proceedings of the 1988 Workshop on Computational Learning Theory*, pages 9–18, San Mateo, CA, 1988. published by Morgan Kaufmann.

[23] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's razor. *Information Processing Letters*, 24:377–380, 1987.

[24] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association for Computing Machinery*, 36(4):929–965, 1989.

[25] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth International Group, 1984.

[26] W. Buntine. *A Theory of Learning Classification Rules*. PhD thesis, University of Technology, Sydney, 1990.

[27] W. Buntine and A. Weigend. Bayesian back propagation. *Complex Systems*, 5:603–643, 1991.

[28] B. Clarke and A. Barron. Entropy, risk and the Bayesian central limit theorem. manuscript.

[29] B. Clarke and A. Barron. Information-theoretic asymptotics of Bayes methods. *IEEE Transactions on Information Theory*, 36(3):453–471, 1990.

[30] T. M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Trans. on Electronic Computers*, EC-14:326–334, 1965.

[31] T. M. Cover. Capacity problems for linear machines. In L. Kanal, editor, *Pattern Recognition*, pages 283–289. Thompson Books, 1968.

[32] J. Denker, D. Schwartz, B. Wittner, S. Solla, R. Howard, L. Jackel, and J. Hopfield. Automatic learning, rule extraction and generalization. *Complex Syst.*, 1:877–922, 1987.

[33] L. Devroye. Automatic pattern recognition: A study of the probability of error. *IEEE Trans. on Pattern Anal. and Mach. Intelligence*, 10(4):530–543, 1988.

[34] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973.

[35] R. M. Dudley. Central limit theorems for empirical measures. *Ann. Prob.*, 6(6):899–929, 1978.

[36] R. M. Dudley. A course on empirical processes. *Lecture Notes in Mathematics*, 1097:2–142, 1984.

[37] R. M. Dudley. Universal Donsker classes and metric entropy. *Ann. Prob.*, 15(4):1306–1326, 1987.

[38] R. Durbin and D. E. Rumelhart. Product units: A computationaly powerful and biologically plausible extension to backporpogation networks. *Neural Computation*, 1(1):133–142, 1989.

[39] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer-Verlag, 1987.

[40] A. Ehrenfeucht, D. Haussler, M. Kearns, and L. Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82:247–261, 1989.

[41] J. D. Farmer. Information dimension and the probabilistic structure of chaos. *Z. Naturforsch. A*, 37:1304–1325, 1982.

[42] J. D. Farmer, E. Ott, and J. A. Yorke. The dimension of chaotic attractors. *Physica D*, 7:153–180, 1983.

[43] T. Ferguson. *Mathematical Statistics: A Decision Theoretic Approach*. Academic Press, 1967.

[44] M. Fulk and J. Case, editors. *Proceedings of the 1990 Workshop on Computational Learning Theory*. Morgan Kaufmann, San Mateo, CA, 1990.

[45] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[46] V. Gullapalli. A stochastic reinforcement algortihm for learning real-valued functions. *Neural Networks*, 3(6):671–692, 1990.

[47] G. Gyorgyi and N. Tishby. Statistical theory of learning a rule. In K. Thuemann and R. Koeberle, editors, *Neural Networks and Spin Glasses*. World Scientific, 1990.

[48] S. E. Hampson and D. J. Volper. Linear function neurons: Structure and training. *Biol. Cybern.*, 53:203–217, 1986.

[49] D. Haussler. Quantifying inductive bias: AI learning algorithms and Valiant's learning framework. *Artificial Intelligence*, 36:177–221, 1988.

[50] D. Haussler. Learning conjunctive concepts in structural domains. *Machine Learning*, 4:7–40, 1989.

[51] D. Haussler. Decision theoretic generalizations of the pac learning model. In *Proc. First Workshop on Algorithmic Learning Theory*, pages 21–41, Yokyo, Japan, 1990.

[52] D. Haussler. Probably approximately correct learning. In *Proc. of the 8th National Conference on Artificial Intelligence*, pages 1101–1108. Morgan Kaufmann, 1990.

[53] D. Haussler. Sphere packing numbers for subsets of the Boolean $n$-cube with bounded Vapnik-Chervonenkis dimension. Technical Report UCSC-CRL-91-41, University of Calif. Computer Research Laboratory, Santa Cruz, CA, 1991. to appear in Journal of Combinatorial Theory A.

[54] D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100(1):78–150, September 1992.

[55] D. Haussler, M. Kearns, N. Littlestone, and M. K. Warmuth. Equivalence of models for polynomial learnability. *Information and Computation*, 95:129–161, 1991.

[56] D. Haussler, M. Kearns, and R. Schapire. Bounds on the sample complexity of Bayesian learning using information theory and the VC dimension. *Machine Learning*, 14:84–114, 1994.

[57] D. Haussler, N. Littlestone, and M. Warmuth. Predicting $\{0, 1\}$-functions on randomly drawn points. Technical Report UCSC-CRL-90-54, University of California Santa Cruz, Computer Research Laboratory, Dec. 1990. To appear, *Information and Computation*.

[58] D. Haussler and L. Pitt, editors. *Proceedings of the 1988 Workshop on Computational Learning Theory*. Morgan Kaufmann, San Mateo, CA, 1988.

[59] D. Haussler and E. Welzl. Epsilon nets and simplex range queries. *Disc. Comp. Geometry*, 2:127–151, 1987.

[60] D. Helmbold and P. Long. Tracking drifting concepts using random examples. In *Proceedings of the 1991 Workshop on Computational Learning Theory*, pages 13–23, San Mateo, CA, August 1991. Morgan Kaufmann.

[61] D. Helmbold, R. Sloan, and M. K. Warmuth. Learning nested differences of intersection closed concept classes. *Machine Learning*, 5:165–196, 1990.

[62] M. Kearns and M. Li. Learning in the presence of malicious errors. In *20th ACM Symposium on Theory of Computing*, pages 267–279, Chicago, 1988.

[63] M. Kearns, M. Li, L. Pitt, and L. Valiant. On the learnability of boolean formulae. In *19th ACM Symposium on Theory of Computing*, pages 285–295, New York, 1987.

[64] M. Kearns, M. Li, L. Pitt, and L. Valiant. On the learnability of boolean formulae. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, pages 285–295, New York, New York, May 1987.

[65] M. Kearns and L. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. In *21st ACM Symposium on Theory of Computing*, pages 433–444, Seattle, WA, 1989.

[66] M. J. Kearns and R. E. Schapire. Efficient distribution-free learning of probabilistic concepts. In *31st Annual Symposium on Foundations of Computer Science*, pages 382–391, 1990.

[67] J. Kiefer. *Introduction to Statistical Inference*. Springer-Verlag, 1987.

[68] A. N. Kolmogorov and V. M. Tihomirov. $\epsilon$-entropy and $\epsilon$-capacity of sets in functional spaces. *Amer. Math. Soc. Translations (Ser. 2)*, 17:277–364, 1961.

[69] T. Kuh, T. Petsche, and R. Rivest. Mistake bounds of incremental learners when concepts drift with applications to feedforward networks. In *NIPS 4*. Morgan Kaufmann, 1991.

[70] S. Kulkarni. On metric entripy, Vapnik-Chervonenkis dimension, and learnability for a class of distributions. Technical Report LIDS-P-1910, Center for Intelligent Control Systems, MIT, 1989.

[71] S. Kullback. *Information Theory and Statistics*. Wiley, New York, 1959.

[72] Y. LeCun, J. Denker, and S. Solla. Optimal brain damage. In D. Touretsky, editor, *Advances in Neural Information Processing Systems*, volume 2, page 589. Morgan Kaufmann, 1990.

[73] L. A. Levin. One-way functions and pseudorandom generators. *Combinatorica*, 7(4):357–363, 1987.

[74] D. V. Lindley. The present position in Bayesian statistics. *Statistical Science*, 5(1):44–89, 1990.

[75] N. Lineal, Y. Mansour, and R. Rivest. Results on learnability and the Vapnik-Chervonenkis dimension. In *Proceedings of the 1988 Workshop on Computational Learning Theory*, pages 56–68, San Mateo, CA, 1988. published by Morgan Kaufmann.

[76] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[77] N. Littlestone. From on-line to batch learning. In *Proceedings of the Second Annual Workshop on Computational Learning Theory*, pages 269–284. Morgan Kaufmann, 1989.

[78] N. Littlestone. *Mistake Bounds and Logarithmic Linear-threshold Learning Algorithms*. PhD thesis, University of California Santa Cruz, 1989.

[79] N. Littlestone, P. Long, and M. Warmuth. On-line learning of linear functions. Technical Report UCSC-CRL-91-29, UC Santa Cruz, October 1991. For an extended abstract see *Proceedings of Twenty Third Annual ACM Symposium on Theory of Computing* New Orlearns, Louisiana, May 1991, pages 465-475.

[80] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

[81] P. Long and M. K. Warmuth. Composite geometric concepts and polynomial learnability. *Information and Computation*, 1991. To appear.

[82] D. MacKay. *Bayesian Methods for Adaptive Models*. PhD thesis, California Institute of Technology, 1992.

[83] B. B. Mandelbrot. *The Fractal Geometry of Nature*. W. H. Freeman, 1982.

[84] P. McCullagh and J. A. Nelder. *Generalized Linear Models.* Chapman and Hall, London, 1989.

[85] T. Mitchell. The need for biases in learning generalizations. Technical Report CBM-TR-117, Rutgers University, New Brunswick, NJ, 1980.

[86] J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294, 1989.

[87] K. S. Narendra and M. A. L. Thathachar. *Learning Automata – An Introduction.* Prentice Hall, 1989.

[88] B. K. Natarajan. Learning over classes of distributions. In *Proceedings of the 1988 Workshop on Computational Learning Theory*, pages 408–409, San Mateo, CA, 1988. published by Morgan Kaufmann.

[89] B. K. Natarajan. On learning sets and functions. *Machine Learning*, 4(1), 1989.

[90] B. K. Natarajan. Probably-approximate learning over classes of distributions. Technical Report HPL-SAL-89-29, Hewlett Packard Labs, Palo Alto, CA, 1989.

[91] B. K. Natarajan. Some results on learning. Technical Report CMU-RI-TR-89-6, Carnegie Mellon, 1989.

[92] B. K. Natarajan and P. Tadepalli. Two new frameworks for learning. In *Proceedings of the 5th International Conference on Machine Learning*, pages 402–415, San Mateo, CA, 1988. published by Morgan Kaufmann.

[93] A. Nobel and A. Dembo. On uniform convergence for dependent processes. Technical Report 74, Stanford University, dept. of Statistics, Stanford, CA, 1990.

[94] D. Nolan and D. Pollard. U-processes: Rates of convergence. *Annals of Statistics*, 15(2):780–799, 1987.

[95] S. Nowlan. Maximum likelihood competitive learning. In D. Touretsky, editor, *Advances in Neural Information Processing Systems*, volume 2, pages 574–582. Morgan Kaufmann, 1990.

[96] S. Nowlan and G. Hinton. Soft weight-sharing. Technical report, Dept. of Comp. Sci., U. Toronto, 1991.

[97] M. Opper and D. Haussler. Calculation of the learning curve of Bayes optimal classification algorithm for learning a perceptron with noise. In *Computational Learning Theory: Proceedings of the Fourth Annual Workshop*, pages 75–87. Morgan Kaufmann, 1991.

[98] M. Opper and D. Haussler. Generalization performance of Bayes optimal classification algorithm for learning a perceptron. *Physical Review Letters*, 66(20):2677–2680, May 1991.

[99] J. Pearl. On the connection between the complexity and credibility of inferred models. *Journal of General Systems*, 4:255–264, 1978.

[100] L. Pitt. Inductive Inference, DFAs, and Computational Comple xity. In *Proceedings of AII-89 Workshop on Analogical andInductive Infer ence; Lecture Notes in Artificial Intelligence 397*, pages 18–44, Heidelberg, October 1989. Springer-Verlag.

[101] L. Pitt and L. Valiant. Computational limitations on learning from examples. *J. ACM*, 35(4):965–984, 1988.

[102] L. Pitt and M. K. Warmuth. Prediction preserving reducibility. *J. Comp. Sys. Sci.*, 41(3):430–467, December 1990. Special issue of the for the *Third Annual Conference of Structure in Complexity Theory* (Washington, DC., June 88).

[103] T. Poggio and F. Girosi. A theory of networks for approximation and learning. Technical Report A.I. Memo No. 1140, Massachusetts Institute of Technology, Cambridge, MA, 1989.

[104] D. Pollard. *Convergence of Stochastic Processes.* Springer-Verlag, 1984.

[105] D. Pollard. Rates of uniform almost-sure convergence for empirical processes indexed by unbounded classes of functions. manuscript, 1986.

[106] D. Pollard. *Empirical Processes: Theory and Applications*, volume 2 of *NSF-CBMS Regional Conference Series in Probability and Statistics.* Institute of Math. Stat. and Am. Stat. Assoc., 1990.

[107] A. J. Quiroz. Metric entropy and learnability. Unpublished manuscript, Universidad Simón Bolívar, Caracas, Venezuela, 1989.

[108] A. Renyi. *Probability Theory.* North Holland, Amsterdam, 1970.

[109] J. Rissanen. Stochastic complexity and modeling. *The Annals of Statistics*, 14(3):1080–1100, 1986.

[110] R. Rivest, D. Haussler, and M. Warmuth, editors. *Proceedings of the 1989 Workshop on Computational Learning Theory.* Morgan Kaufmann, San Mateo, CA, 1989.

[111] R. L. Rivest. Learning decision lists. *Machine Learning*, 2:229–246, 1987.

[112] D. Rumelhart. personal communication, 1990.

[113] D. E. Rumelhart and J. L. McClelland. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations.* MIT Press, Cambridge, Mass., 1986.

[114] W. Sarrett and M. Pazzani. Average case analysis of empirical and explanation-based learning algorithms. *Machine Learning*, 9(4):349–372, 1992.

[115] G. Shackelford and D. Volper. Learning $k$-DNF with noise in the attributes. In *Proceedings of the 1988 Workshop on Computational Learning Theory*, pages 97–103, San Mateo, CA, 1988. published by Morgan Kaufmann.

[116] R. Sloan. Types of noise in data for concept learning. In *Proc. 1988 Workshop on Comp. Learning Theory*, pages 91–96, San Mateo, CA, 1988. Morgan Kaufmann.

[117] H. Sompolinsky, N. Tishby, and H. Seung. Learning from examples in large neural networks. *Physical Review Letters*, 65:1683–1686, 1990.

[118] C. L. T. Cormen and R. Rivest. *Introduction to algorithms.* MIT Press, 1990.

[119] M. Talagrand. Sharper bounds for Gaussian and empirical processes. *Annals of Probability*, 22(1):28–76, 1994.

[120] G. Tesauro and D. Cohn. Can neural networks do better than the Vapnik-Chervonenkis bounds? In R. Lippmann, J. Moody, and D. Touretzky, editors, *Advances in Neural Information Processing, Vol. 3*, pages 911–917. Morgan Kaufmann, 1991.

[121] N. Tishby, E. Levin, and S. Solla. Consistent inference of probabilities in layered networks: predictions and generalizations. In *IJCNN International Joint Conference on Neural Networks*, volume II, pages 403–409. IEEE, 1989.

[122] D. Touretsky. *Advances in Neural Information Processing Systems*, volume 1. Morgan Kaufmann, 1989.

[123] D. Touretsky. *Advances in Neural Information Processing Systems*, volume 2. Morgan Kaufmann, 1990.

[124] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, Nov. 1984.

[125] L. G. Valiant. Learning disjunctions of conjunctions. In *Proc. 9th IJCAI*, volume 1, pages 560–6, Los Angeles, August 1985.

[126] L. G. Valiant and M. Warmuth, editors. *Proceedings of the 1991 Workshop on Computational Learning Theory*. Morgan Kaufmann, San Mateo, CA, 1991.

[127] V. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its applications*, XVI(2):264–280, 1971.

[128] V. N. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, 1982.

[129] V. N. Vapnik. Inductive principles of the search for empirical dependences (methods based on weak convergence of probability measures). In *Proceedings of the 2nd Workshop on Computational Learning Theory*, San Mateo, CA, 1989. published by Morgan Kaufmann.

[130] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–80, 1971.

[131] S. Venkatesh. On learning binary weights for majority functions. In *Computational Learning Theory: Proceedings of the Fourth Annual Workshop*, pages 257–266. Morgan Kaufmann, 1991.

[132] V. Vovk. Aggregating strategies. In *Proceedings of the 3nd Workshop on Computational Learning Theory*, pages 371–383. published by Morgan Kaufmann, 1990.

[133] A. Weigend, B. Huberman, and D. Rumelhart. Predicting the future: A connectionist approach. *International Journal of Neural Systems*, 1:193–209, 1990.

[134] S. Weiss and C. Kulikowski. *Computer Systems that Learn*. Morgan Kaufmann, San Mateo, CA, 1991.

[135] E. Welzl. Partition trees for triangle counting and other range search problems. In *4th ACM Symp. on Comp. Geometry*, pages 23–33, Urbana, IL, 1988.

[136] R. S. Wenocur and R. M. Dudley. Some special Vapnik-Chervonenkis classes. *Discrete Mathematics*, 33:313–318, 1981.

[137] H. White. Connectionist nonparametric regression: Multilayer feedforward networks can learn arbitrary mappings. *Neural Networks*, 3:535–549, 1990.

[138] H. White. Learning in artificial neural networks: a statistical perspective. *Neural Computation*, 1(4):425–464, 1990.

[139] K. Yamanishi. A learning criterion for stochastic rules. In *Proceedings of the 3nd Workshop on Computational Learning Theory*, pages 67–81. published by Morgan Kaufmann, 1990.

[140] K. Yamanishi. A learning criterion for stochastic rules. *Machine Learning*, 1992. Special Issue on the Proceedings of the 3nd Workshop on Computational Learning Theory, to appear.