A General Lower Bound on the Number of Examples Needed for Learning

Andrzej Ehrenfeucht
University of Colorado

David Haussler
U.C. Santa Cruz

Michael Kearns
Harvard University

Leslie Valiant
Harvard University

Abstract

We prove a lower bound of $\Omega(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{\mathrm{VCdim}(C)}{\epsilon})$ on the number of random examples required for distribution-free learning of a concept class C, where $\mathrm{VCdim}(C)$ is the Vapnik-Chervonenkis dimension and ϵ and δ are the accuracy and confidence parameters. This improves the previous best lower bound of $\Omega(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \mathrm{VCdim}(C))$, and comes close to the known general upper bound of $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{\mathrm{VCdim}(C)}{\epsilon}\ln\frac{1}{\epsilon})$ for consistent algorithms. We show that for many interesting concept classes, including kCNF and kDNF, our bound is actually tight to within a constant factor.

[0]A. Ehrenfeucht was supported by NSF grant MCS-8305245, D. Haussler by ONR grant N00014-86-K-0454, M. Kearns by ONR grant N00014-86-K-0454 and an A.T. & T. Bell Laboratories Scholarship, and L. Valiant by grants ONR-N00014-85-K-0445, NSF-DCR-8600379, DAAL03-86-K-0171 and by the SERC of the U.K. Part of this research was done while M. Kearns was visiting U.C. Santa Cruz.

1 Introduction

In [V84], a stochastic model of machine learning from examples based on computational complexity was introduced. Informally, this model can be described as follows: positive and negative examples of some unknown target concept, chosen from a concept class C, are presented to a learning algorithm. These examples are drawn randomly according to a fixed but arbitrary probability distribution. From the examples drawn, the learning algorithm must, with high probability, produce a hypothesis concept that is a good approximation to the target.

Most of the recent research in this model (see e.g. [AL86], [BEHW86,87a,87b], [BI88], [KLPV87], [KL87], [KV88], [LMR88], [N87], [PV86], [R87], [S88], [V85], [VL88]) has emphasized the broad distinction between those classes that are learnable in polynomial time and those that are not. Little attention has been paid to determining precise complexity bounds (both upper and lower) for classes already known to be learnable in polynomial time.

In this paper, we resolve several problems regarding the *sample complexity* (i.e., the number of examples required) for learning various concept classes by giving a general lower bound theorem. We apply this result to show that the existing algorithms for learning monomials, kDNF formulae, kCNF formulae and symmetric functions all use the optimal number of examples (within a constant factor). By similar methods, we prove that the algorithm given in [R87] for learning decision lists on n variables uses a sample size that is at most a logarithmic factor off optimal, and give an alternative analysis of this algorithm that yields a small improvement in its sample size. We also show that some existing algorithms for concept classes over continuous domains use a sample size that is within a multiplicative logarithmic factor of optimal.

The lower bound we prove is information-theoretic in the sense that *no* algorithm in the learning model of [V84], even one with infinite computational resources, can learn from fewer examples. It comes within a multiplicative log factor of the information-theoretic upper bound on the number of examples needed by any algorithm that always produces consistent hypotheses in the target concept class [V82] [BEHW86,87b].

The outline of the paper is as follows: in Section 2, we define the model of [V84], and give relevant notation. Section 3 presents our main result, the lower bound. In Section 4 we apply the lower bound to obtain tight and nearly tight bounds on the sample complexity for learning several well-studied concept classes. Section 5 mentions open problems.

2 Definitions and Notation

Let $P_1, \ldots, P_k$ be probability distributions over spaces $X_1, \ldots, X_k$ respectively. Let $E(v_1, \ldots, v_k)$ and $\psi(v_1, \ldots, v_k)$ be an event and a random variable, respectively, where $v_i \in X_i, 1 \leq i \leq k$. Then we denote by $\mathbf{Pr}_{v_1 \in P_1, \ldots, v_k \in P_k}(E)$ the probability of E and by $\mathbf{E}_{v_1 \in P_1, \ldots, v_k \in P_k}(\psi)$ the expectation of ψ, when each v_i is independently chosen according to the distribution P_i. If P is a distribution over X, then P^m shall denote the m-fold product distribution defined by P over X^m.

Let X be a set which we will call the *domain*, and let $C \subseteq 2^X$ be a *concept class* over X. In this paper we assume that X is either finite, countably infinite, or n-dimensional Euclidean space for some $n \geq 1$ (see [BEHW86,87b] for additional measurability restrictions on C for Euclidean domains). An *example* of a *concept* $c \in C$ is a pair (x, b) where $x \in X$, $b \in \{0, 1\}$, and $b = 1$ if and only if $x \in c$. For $\vec{x} = (x_1, \ldots, x_m) \in X^m$ and $c \in C$, we denote by $< \vec{x}, c >$ the *sample of c generated by $\vec{x}$* — namely, $< \vec{x}, c >= ((x_1, b_1), \ldots, (x_m, b_m))$ where $b_i = 1$ if and only if $x_i \in c$. The *size* of $< \vec{x}, c >$ is m. A *random sample* of $c \in C$ according to a distribution P over X is a sample $< \vec{x}, c >= ((x_1, b_1), \ldots, (x_m, b_m))$ where each x_i is drawn independently according to P.

Let $c \in C$ be a fixed *target concept* and P a distribution on the domain X. Given input ϵ and δ, $0 < \epsilon, \delta < 1$, a *(randomized) learning algorithm* A draws a random sample $< \vec{x}, c >$ (according to P) of size $m_A(\epsilon, \delta)$ and a random bit string r of length $r_A(\epsilon, \delta)$ (representing unbiased coin tosses available to the algorithm), and produces an *hypothesis* $h = h_A(< \vec{x}, c >, r) \in 2^X$. Here $m_A(\epsilon, \delta)$ and $r_A(\epsilon, \delta)$ are positive integer-valued functions of ϵ and δ. We call $m_A(\epsilon, \delta)$ the *sample size used by A*.

Let $e_A(< \vec{x}, c >, r) = P(h \Delta c)$, where $h = h_A(< \vec{x}, c >, r)$ and Δ denotes the symmetric difference. Thus $e_A(< \vec{x}, c >, r)$ is the probability that the hypothesis h produced by A on inputs $< \vec{x}, c >$ and r disagrees with the target concept c on a point randomly drawn from X according to the distribution P. We refer to this as the *error* of the hypothesis h.

Let U be the uniform distribution on $\{0, 1\}$. For a given ϵ and δ, let $m = m_A(\epsilon, \delta)$ and $k = r_A(\epsilon, \delta)$. Then A is an *(ϵ, δ)-learning algorithm for C* if for every distribution P on X and every target concept $c \in C$,

$$\mathbf{Pr}_{\vec{x} \in P^m, r \in U^k}(e_A(< \vec{x}, c >, r) > \epsilon) \leq \delta.$$

Thus we require that for any distribution and target concept, with probability at least $1 - \delta$, A produces a hypothesis with error at most ϵ. We will call ϵ the *accuracy parameter* and δ the *confidence parameter*. We note that our results also hold in the model where distributions over the positive and negative examples are distinguished (e.g. [V84], [KLPV87]), with only minor modifications to the proof.

If C is a concept class over domain X, and $W \subseteq X$, we say that W is *shattered* by C if for every $W' \subseteq W$, there exists a $c \in C$ such that $W' = c \cap W$. We define the *Vapnik-Chervonenkis dimension* of C (denoted $\mathrm{VCdim}(C)$) to be the cardinality of the largest $W \subseteq X$ such that W is shattered by C. Note that for $|C|$ finite, we have $\mathrm{VCdim}(C) \leq \log|C|$.

3 The Lower Bound

Theorem 1 *Assume* $0 < \epsilon \leq \frac{1}{8}$, $0 < \delta \leq \frac{1}{100}$, *and* $\mathrm{VCdim}(C) \geq 2$. *Then any* (ϵ, δ)*-learning algorithm* A *for* C *must use sample size*

$$m_A(\epsilon, \delta) \geq \frac{\mathrm{VCdim}(C) - 1}{32\epsilon} = \Omega(\frac{\mathrm{VCdim}(C)}{\epsilon}).$$

Proof: Let the set $X_0 = \{x_0, x_1, \ldots, x_d\} \subseteq X$ be shattered by C, where $d = \mathrm{VCdim}(C) - 1$. We define the following distribution P on X:

$$P(x_0) = 1 - 8\epsilon$$
$$P(x_i) = \tfrac{8\epsilon}{d}, 1 \leq i \leq d.$$

Since P is 0 except on X_0, we may assume without loss of generality that $X = X_0$ and $C \subseteq 2^{X_0}$. Since X_0 is shattered by C, we may further assume that $C = 2^{X_0}$. Let $C_0 \subseteq C$ be defined by

$$C_0 = \{\{x_0\} \cup T : T \subseteq \{x_1, \ldots, x_d\}\}.$$

Fix $\epsilon \leq \frac{1}{8}$ and $\delta \leq \frac{1}{100}$, and let $m = m_A(\epsilon, \delta)$. We define the set $S \subset X^m$ to be the set of all $\vec{x} \in X^m$ such that there are at most $\frac{d}{2}$ distinct elements from $\{x_1, \ldots, x_d\}$ appearing in $\vec{x}$.

Lemma 2 *Let* $m = m_A(\epsilon, \delta)$, $k = r_A(\epsilon, \delta)$, *and let* U *be the uniform distribution on* $\{0, 1\}$. *Then there exists a* $c_0 \in C_0$ *satisfying*

$$\mathbf{Pr}_{\vec{x} \in P^m, r \in U^k}(e_A(< \vec{x}, c_0 >, r) > \epsilon) > \frac{1}{7} P^m(S).$$

Proof: Fix $\vec{x}_0 \in S$ and the bit string $r_0 \in \{0, 1\}^k$, and let l be the number of distinct elements of $\{x_1, \ldots, x_d\}$ appearing in $\vec{x}_0$. Note that $l \leq \frac{d}{2}$ by definition of S. Now for each $c \in C_0$, there are exactly 2^{d-l} concepts in C_0 that are consistent with the sample $< \vec{x}_0, c >$, since $\{x_1, \ldots, x_d\}$ is shattered by C_0. For any of the $d - l$ points in $\{x_1, \ldots, x_d\}$ that do not appear in $\vec{x}_0$, half of these concepts will contain the point, and half will not, hence $h_A(< \vec{x}_0, c >, r)$ is correct on this point for exactly half of these 2^{d-l} consistent concepts. Thus, if we denote by $w_A(<\vec{x}, c>, r)$ the number of points in $\{x_1, \ldots, x_d\}$ on which $h_A(<\vec{x}, c>, r)$ and c disagree, and by Q the uniform distribution over C_0, then we have shown (for $\vec{x}_0$ and r_0 fixed)

$$\mathbf{E}_{c\in Q}(w_A(<\vec{x}_0, c>, r_0)) \geq \frac{d-l}{2} \geq \frac{d}{4}.$$

The first inequality comes from the fact that the hypothesis of A may also be incorrect on points that did appear in $\vec{x}_0$, and the second inequality from the fact that $l \leq \frac{d}{2}$. Without loss of generality we may assume that any learning algorithm is always correct on the point x_0 whenever concepts are selected from C_0, so we may write

$$e_A(<\vec{x}_0, c>, r_0) = \frac{8\epsilon}{d} w_A(<\vec{x}_0, c>, r_0).$$

Then

$$\begin{aligned} (1)\quad \mathbf{E}_{c\in Q}(e_A(<\vec{x}_0, c>, r_0)) &= \mathbf{E}_{c\in Q}(\frac{8\epsilon}{d} w_A(<\vec{x}_0, c>, r_0)) \\ &= \frac{8\epsilon}{d}\mathbf{E}_{c\in Q}(w_A(<\vec{x}_0, c>, r_0)) \geq \frac{8\epsilon}{d}\frac{d}{4} = 2\epsilon. \end{aligned}$$

Let $P^m|S$ be the distribution P^m restricted to the set S. Since (1) holds for any fixed $\vec{x}_0 \in S$ and any fixed bit string r_0, if we instead choose $\vec{x}$ randomly according to $P^m|S$ and r randomly according to U^k, we have

$$\mathbf{E}_{c\in Q, \vec{x}\in P^m|S, r\in U^k}(e_A(<\vec{x}, c>, r)) \geq 2\epsilon.$$

Thus, there must be some fixed $c_0 \in C_0$ satisfying

$$(2)\quad \mathbf{E}_{\vec{x}\in P^m|S, r\in U^k}(e_A(<\vec{x}, c_0>, r)) \geq 2\epsilon.$$

On the other hand, we have

$$(3)\quad e_A(<\vec{x}, c_0>, r) \leq 8\epsilon$$

for any $\vec{x} \in X^m$ and any bit string r, since we assume that the hypothesis of A is correct on the point x_0. From (2) and (3) we can show

$$(4)\quad \mathbf{Pr}_{\vec{x}\in P^m|S, r\in U^k}(e_A(<\vec{x}, c_0>, r) \geq \epsilon) > \frac{1}{7}.$$

To see this, let ψ be a random variable whose expectation $\mathbf{E}(\psi)$ according to some distribution is at least 2ϵ, but whose absolute value is bounded above by 8ϵ (as is the case with the random variable $e_A(<\vec{x}, c_0>, r)$). Then if p is the probability that ψ is larger than ϵ, we have

$$2\epsilon \leq \mathbf{E}(\psi) < p8\epsilon + (1-p)\epsilon.$$

Solving for p, we obtain $p > \frac{1}{7}$ as claimed.

Now from (4) we have

$$\mathbf{Pr}_{\vec{x}\in P^m, r\in U^k}(e_A(<\vec{x}, c_0>, r) \geq \epsilon) \geq P^m(S)\mathbf{Pr}_{\vec{x}\in P^m|S, r\in U^k}(e_A(<\vec{x}, c_0>, r) \geq \epsilon) > \frac{1}{7}P^m(S).$$

This completes the proof of the lemma. $\square$

Lemma 3 *Let $\delta \leq \frac{1}{100}$. For $m = \frac{d}{32\epsilon}$, $P^m(S) > 7\delta$.*

Proof: We will use the following fact from probability theory (Proposition 2.4, [AV79]):

For $0 \leq p \leq 1$ and m, r positive integers, let $GE(p, m, r)$ denote the probability of at least r successes in m independent trials of a Bernoulli variable with probability of success p.

Fact. For any $0 \leq \alpha \leq 1$, $GE(p, m, (1+\alpha)mp) \leq e^{-\alpha^2 mp/3}$.

Let E be the event that a point drawn at random according to P is contained in $\{x_1, \ldots, x_d\}$. Thus, $P(E) = 8\epsilon$. Now if $\vec{x}$ is drawn from P^m and $\vec{x} \notin S$, then clearly event E must occur at least $\frac{d}{2}$ times in $\vec{x}$. The probability of event E occurring at least $\frac{d}{2}$ times in m draws from P is bounded above by $GE(8\epsilon, m, \frac{d}{2})$. Setting $m = \frac{d}{32\epsilon}$, we have

$$GE(8\epsilon, \frac{d}{32\epsilon}, \frac{d}{2}) = GE(8\epsilon, \frac{d}{32\epsilon}, 2\frac{d}{32\epsilon}8\epsilon) \leq e^{-\frac{d}{12}} \leq e^{-\frac{1}{12}}.$$

But $e^{-\frac{1}{12}} < 1 - 7\delta$ for $\delta < \frac{1}{100}$. Thus, for such δ we have $P^m(S) > 7\delta$. $\square$

By Lemmas 2 and 3, if $m_A(\epsilon, \delta) \leq \frac{d}{32\epsilon}$, then there is probability at least δ that A outputs a hypothesis with error greater than ϵ, thus proving the theorem. $\square$

A slightly modified version of the proof of Theorem 1 can be used to show that when $\mathrm{VCdim}(C) \geq 2$ and the sample size is less than $\frac{\mathrm{VCdim}(C)-1}{2e\epsilon}$ (where e denotes the base of the natural logarithm), for any learning algorithm there is a distribution and a target concept such that the expected error of the hypothesis produced by the learning algorithm is at least ϵ [HLW88]. No serious attempt has been made to optimize the constants in either this result or in Theorem 1 above.

It should be noted that the lower bound of Theorem 1 holds independent of the *computational* complexity of a learning algorithm — that is, even algorithms allowed infinite computational resources must use $\Omega(\frac{\mathrm{VCdim}(C)}{\epsilon})$ examples. Theorem 1 also makes no assumptions on the class from which the learning algorithm's hypothesis is chosen (in particular, the hypothesis h need not be chosen from C for the bound to hold).

For purposes of comparison, we now state precisely the previous best lower bound on the sample size. We say that the concept class C is *trivial* if C consists of one concept, or two disjoint concepts whose union is the domain X.

Theorem 4 *[BEHW86,87b] Let C be a non-trivial concept class. Then any (ϵ, δ)-learning algorithm A for C must use sample size*

$$m_A(\epsilon, \delta) = \Omega(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \mathrm{VCdim}(C)).$$

$\square$

Theorem 1 improves separate lower bounds proportional to $\mathrm{VCdim}(C)$ and $\frac{1}{\epsilon}$ to a single lower bound that is proportional to their product. Using Theorems 1 and 4, we obtain:

Corollary 5 *Let C be a non-trivial concept class. Then any (ϵ, δ)-learning algorithm A for C must use sample size*

$$m_A(\epsilon, \delta) = \Omega(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{\mathrm{VCdim}(C)}{\epsilon}).$$

$\square$

4 Applications

For fixed ϵ and δ, define the *sample complexity* of learning a concept class C to be the minimum sample size $m_A(\epsilon, \delta)$ over all (ϵ, δ)-learning algorithms A for C. In this section we apply Theorem 1 to obtain lower bounds on the sample complexity for a variety of concept classes. These bounds obtained are tight within a constant factor in many important cases.

We begin by recalling results of [BEHW86,87a,87b], derived from [V82], that bound the sample complexity of algorithms that produce hypotheses in the target class C that are consistent with the examples they have seen. We will call A a *consistent algorithm* for C if whenever A receives examples of a concept in C, it always produces a hypothesis that is consistent with those examples. If A always outputs a hypothesis $h \in H \subset 2^X$, then we say that A *uses hypothesis space* H.

Theorem 6 *[V82] [BEHW86,87a,87b] Let A be a consistent algorithm for C using hypothesis space C, and let $0 < \epsilon, \delta < 1$. Then*

(i) A is an (ϵ, δ)-learning algorithm for C with sample size

$$m_A(\epsilon, \delta) = O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{\mathrm{VCdim}(C)}{\epsilon}\ln\frac{1}{\epsilon}).$$

(ii) If C is finite then A is an (ϵ, δ)-learning algorithm for C with sample size

$$m_A(\epsilon, \delta) = O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{\ln|C|}{\epsilon}).$$

$\square$

We now apply Corollary 5 and Theorem 6 to obtain lower and upper bounds on the sample complexity of learning several well-studied classes of concepts. For many classes the lower bound is met or almost met (e.g., within a log factor) by an algorithm that is both efficient (i.e., runs in time polynomial in the length of the sample) and works for all values of ϵ and δ. We begin with Boolean functions. We will use the following notation: if $f : \{0,1\}^n \rightarrow \{0,1\}$ is a Boolean function, then $s(f) \subseteq \{0,1\}^n$ will be the set $\{\vec{v} \in \{0,1\}^n : f(\vec{v}) = 1\}$. For all of our Boolean examples, the domain X is just $\{0,1\}^n$ and the concept class $C \subseteq 2^X$ is the class of all $s(f)$, where f is a function of the type under consideration.

Monomials: Monomials are simply conjunctions of literals over the variables $x_1, \ldots, x_n$. For each $1 \leq i \leq n$, let $\vec{u}_i \in \{0,1\}^n$ be the assignment with the ith bit set to 0, and all other bits set to 1. To see that $S = \{\vec{u}_i : 1 \leq i \leq n\}$ is shattered by the class C of monomials, let $S' = \{\vec{u}_{l_1}, \ldots, \vec{u}_{l_m}\} \subseteq S$ and $S - S' = \{\vec{u}_{l_{m+1}}, \ldots, \vec{u}_{l_n}\} \subseteq S$. Then $S \cap s(x_{l_{m+1}} \cdots x_{l_n}) = S'$, so S is shattered. Thus, we have that $\mathrm{VCdim}(C) \geq |S| = n$, so by Corollary 5 the sample complexity of learning C is $\Omega(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n}{\epsilon})$. On the other hand, in [V84] an efficient consistent algorithm for C using hypothesis space C is given. Since $|C| = 3^n$, we have by Theorem 6(ii) that C is learnable with sample complexity $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n}{\epsilon})$, which is within a constant factor of the lower bound. Note that by duality, we also have a tight lower bound for the class of disjunctions of literals.

This example demonstrates a more general principle: if $\ln|C| = O(\mathrm{VCdim}(C))$ and there is an efficient consistent algorithm for C using hypothesis space C, then C is efficiently learnable (by Theorem 6(ii)) with provably optimal sample complexity to within a constant factor (by Corollary 5).

kDNF Formulae: The kDNF formulae are disjunctions of bounded monomials, i.e., formulae of the form $T_1 + \cdots + T_l$ where each T_i is a monomial length at most k. There is no bound on the number of disjuncts l. Let $S \subseteq \{0,1\}^n$ be the set of all vectors with exactly k bits assigned 1 and all remaining bits assigned 0, so $|S| = \Theta(n^k)$. To see that S is shattered by the class C of kDNF formulae, let $S' = \{\vec{u}_{l_1}, \ldots, \vec{u}_{l_m}\} \subseteq S$. Let T_{l_i} be the monomial containing the conjunction of all variables that are assigned 1 in the vector $\vec{u}_{l_i}$ (thus, the length of T_{l_i} is exactly k). Then $S \cap s(T_{l_1} + \cdots + T_{l_m}) = S'$, and S is shattered. By Corollary 5, we have a lower bound of $\Omega(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n^k}{\epsilon})$ on the number of examples needed to learn kDNF. Since $\ln|C| = O(n^k)$ and [V84] gives a consistent algorithm for kDNF using kDNF hypotheses that runs in time polynomial in the length of the sample, this lower bound proves that the algorithm of [V84] is optimal in terms of the number of examples used by Theorem 6(ii). By duality, we have an analogous result for the class kCNF of conjunctions of clauses, where each clause is a disjunction of at most k literals.

Symmetric Functions: A symmetric function $f : \{0,1\}^n \to \{0,1\}$ is a function that is invariant under permutations of the input bits — thus, the value of f is uniquely determined by the number of 1's in the input. Let the vector $\vec{u}_i$ be 1 on the first i bits and 0 on the last $n-i$ bits, and let $S = \{\vec{u}_i : 0 \leq i \leq n\}$. If $S' = \{\vec{u}_{l_1}, \ldots, \vec{u}_{l_m}\} \subseteq S$, and f is the symmetric function that is 1 if and only if the number of bits assigned 1 in the input is contained in the set $\{l_1, \ldots, l_m\}$, then $S \cap s(f) = S'$, so S is shattered by symmetric functions. Hence, $\text{VCdim}(C) \geq |S| = n+1$. Corollary 5 then gives a lower bound of $\Omega(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n}{\epsilon})$ on the number of examples needed to learn symmetric functions, proving that the algorithm given in [KL87] has optimal sample complexity.

k-term DNF: For constant k, a k-term DNF formulae is one of the form $T_1 + \cdots + T_k$, where each T_i is a monomial whose length is not restricted. If C is the class of k-term DNF concepts, then by arguments similar to those given above it can be shown that $\text{VCdim}(C) = \Theta(n)$ for fixed k. Thus Corollary 5 gives a lower bound of $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n}{\epsilon})$ on the sample size required for learning C. However, the best known efficient algorithm for learning C (given in [PV86]) uses the algorithm of [V84] for k-CNF, and thus needs sample size $\Omega(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n^k}{\epsilon})$. Note that the hypothesis produced by this learning algorithm is not in the class C, but in kCNF. It is shown in [PV86] (see also [KLPV87]) that learning k-term DNF using hypothesis space k-term DNF is NP-hard. There is an algorithm using hypothesis space k-term DNF, but it is an exhaustive-search algorithm requiring superpolynomial time. Thus, for this example there is a significant gap ($O(n^{k-1})$) between the information-theoretic lower bound and the smallest sample size used by an efficient learning algorithm.

l-term kDNF: If C is the class of all kDNF functions (for k fixed) with at most l terms then $\text{VCdim}(C) = \Theta(l \ln\frac{n}{l})$ ([L87b]). Results in [L88] and [HLW87] can be combined to show that there is an efficient (ϵ, δ)-learning algorithm for C using sample size $O(\frac{l\ln\frac{n}{l}}{\epsilon}\ln\frac{1}{\delta})$. Corollary 5 shows that this sample size exceeds the optimal by at most a factor of $O(\ln\frac{1}{\delta})$.

k-Decision Lists: A *k-Decision List* ([R87]) is a list $L = \langle (M_1, b_1), (M_2, b_2), \ldots, (M_m, b_m) \rangle$ where each M_i is a monomial containing at most k literals, and $b_i \in \{0,1\}$. The value of $L(\vec{v})$ for $\vec{v} \in \{0,1\}^n$ is defined as follows: let $1 \leq i \leq m$ be the least value such that $M_i(\vec{v}) = 1$. Then $L(\vec{v}) = b_i$ (or 0 if no such i exists). In [R87] it is shown that the concept class represented by

k-Decision Lists properly includes the kCNF and kDNF functions, and an efficient consistent algorithm for learning k-Decision Lists using k-Decision List hypotheses is given that uses sample size $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n^k}{\epsilon}\ln n)$. The analysis of this algorithm uses Theorem 6(ii). It is shown in the appendix of Section 7 that if C is the class of k-Decision Lists, then $\mathrm{VCdim}(C) = \Theta(n^k)$, thus giving a lower bound on the sample size of $\Omega(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n^k}{\epsilon})$ by Corollary 5. Thus, the sample size of the algorithm of [R87] is at most $O(\ln n)$ above the optimal. Furthermore, the upper bound on $\mathrm{VCdim}(C)$ yields an alternative analysis of this algorithm: by applying Theorem 6(i), we see that in fact a sample of size $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n^k}{\epsilon}\ln\frac{1}{\epsilon})$ also suffices. If it is decided at run time which log factor is smaller, then we have shown that the sample complexity of the algorithm of [R87] is in fact $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n^k}{\epsilon}\min(\ln\frac{1}{\epsilon}, \ln n))$, a factor of $\min(\ln\frac{1}{\epsilon}, \ln n)$ above optimal.

We now turn our attention to examples where the concept class is defined over a continuous domain.

Linear Separators (Half-spaces): Let C be the class of all half-spaces (open or closed) in Euclidean n-dimensional space E^n. Then $\mathrm{VCdim}(C) = n + 1$ (see e.g. [WD81] or [HW87]), and an efficient consistent algorithm for C using hypotheses in C can be implemented using linear programming (see [K84], [K79], see [BEHW86,87b]). By Theorem 6(i) this algorithm requires sample size $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n}{\epsilon}\ln\frac{1}{\epsilon})$, which is within a factor of $O(\ln\frac{1}{\epsilon})$ of optimal by Corollary 5.

Axis-Parallel Rectanges: An axis-parallel rectangle in E^n is the cross product of n open or closed intervals, one on each axis. If C is the concept class of all such rectangles, then $\mathrm{VCdim}(C) = 2n$ ([WD81], see [BEHW86,87b]) and an efficient (ϵ, δ)-learning algorithm for C is given in an example of [BEHW86,87b], using sample size $O(\frac{n}{\epsilon}\ln\frac{n}{\delta})$. By Corollary 5, this bound is off from optimal by a factor of at most $O(\ln\frac{n}{\delta})$. Since the algorithm of [BEHW86,87b] is also a consistent algorithm using hypotheses in C, from Theorem 6(i) we obtain a different upper bound on its sample size, namely $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{n}{\epsilon}\ln\frac{1}{\epsilon})$. This bound is off from optimal by a factor of at most $O(\ln\frac{1}{\epsilon})$.

In [H88] other applications of Corollary 5 to learning algorithms in Artificial Intelligence domains are given. Further consequences of Corollary 5 for learning algorithms on feedforward neural networks of linear threshold functions are discussed in [BH88].

5 Open Problems

Disregarding computational resources, does there always exist an (ϵ, δ)-learning algorithm for C using sample size $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{\mathrm{VCdim}(C)}{\epsilon})$? It is shown in [HLW87] that the upper bound of $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{\mathrm{VCdim}(C)}{\epsilon}\ln\frac{1}{\epsilon})$ given in Theorem 6 for arbitrary consistent algorithms using hypotheses in C cannot be improved, i.e. for all $d \geq 1$ there are concept classes C with $VCdim(C) = d$ with consistent algorithms using hypotheses in C requiring at least $\Omega(\frac{\mathrm{VCdim}(C)}{\epsilon}\ln\frac{1}{\epsilon})$ examples. They also show that there always exists a (not necessarily consistent) (ϵ, δ)-learning algorithm for C using sample size $O(\frac{\mathrm{VCdim}(C)}{\epsilon}\ln\frac{1}{\delta})$. However, this also fails to meet the lower bound.

Restricting attention to polynomial time computation, do there exist efficient learning algorithms for C the class of k-Decision Lists, k-term DNF, l-term kDNF, half spaces or axis parallel rectangles, using sample size $O(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{\mathrm{VCdim}(C)}{\epsilon})$?

6 Acknowledgements

We gratefully acknowledge helpful discussions with Sally Floyd, Nick Littlestone and Manfred Warmuth. We also thank an anonymous referee for several valuable comments.

7 Appendix

Theorem 7 *Let* kDL *be the class of* k*-Decision Lists. Then* $\mathrm{VCdim}(k\mathrm{DL}) = \Theta(n^k)$.

Proof: The lower bound on the dimension follows easily from the fact that kDL contains the class of kDNF functions [R87], thus $\mathrm{VCdim}(k\mathrm{DL}) \geq \mathrm{VCdim}(k\mathrm{DNF}) = \Theta(n^k)$. For the upper bound, we begin by proving that $\mathrm{VCdim}(1\mathrm{DL}) = O(n)$. We then give a simple transformation that proves the theorem for arbitrary k. We adopt the following notation: let

$$L \ = < (l_1, b_1), (l_2, b_2), \ldots, (l_m, b_m) >$$

denote a 1-Decision List, where each l_i is a literal over the Boolean variables $\{x_1, \ldots, x_n\}$, and each $b_i \in \{0, 1\}$. We will call each pair (l_i, b_i) an *item* of the list L.

We show that the class of 1DL functions is linearly separable — that is, for each $L \in 1\mathrm{DL}$ there is a linear function

$$P_L(x_1, \ldots, x_n) = c_1x_1 + c_2x_2 + \cdots + c_nx_n$$

with coefficients $c_i \in \mathrm{R}, 1 \leq i \leq n$ and a *threshold value* $\tau \in \mathrm{R}$ such that for $x_i \in \{0,1\}, 1 \leq i \leq n$

$$(1) \quad P_L(x_1, \ldots, x_n) \geq \tau \iff L(x_1, \ldots, x_n) = 1.$$

Let C be the class of all half-spaces in Euclidean n-dimensional space. Then it follows that $\mathrm{VCdim(1DL)} \leq \mathrm{VCdim}(C) = n+1$ (see e.g. [WD81],[HW87]). Hence $\mathrm{VCdim(1DL)} = O(n)$.

P_L is constructed as follows: with each item (l_i, b_i), associate a linear term $T_i = T(l_i, b_i)$ involving a variable in $\{x_1, \ldots, x_n\}$, defined by

$$T(x_i, 1) = x_i \quad T(x_i, 0) = 1 - x_i \quad T(\overline{x}_i, 1) = 1 - x_i \quad T(\overline{x}_i, 0) = x_i.$$

We now describe a method for constructing a linear inequality of the form

$$(2) \quad P_L(x_1, \ldots, x_n) = a_1 T_1 + \cdots + a_m T_m \geq \tau$$

with $a_i \in \mathrm{R}, 1 \leq i \leq m$, and satisfying (1). This is done by constructing the coefficients a_i and τ in the order $a_m, \ldots, a_1, \tau$ according to the following inductive rules. We assume without loss of generality that $b_m = 1$; if $b_m = 0$ then L is equivalent to the list of length $m-1$ obtained by deleting the mth item.

I. (First step) $a_m = 1$.

II. (The 0-Rule) If $b_i = 0$ then $a_i = 1 + \sum_{j=i+1}^{m} a_j$.

III. (The 1-Rule) If $b_i = 1$ then $a_i = \sum_{j=i+1}^{k} a_j$, where k is the least l, $i+1 \leq l \leq m$ such that $b_l = 1$.

IV. (The τ-Rule) $\tau = \sum_{j=1}^{k} a_j$, where k is the least l, $1 \leq l \leq m$ such that $b_l = 1$.

Note that this procedure always constructs non-negative coefficients for the linear terms T_i, so $P_L(x_1, \ldots, x_n) \geq 0$ always. We also note that $\tau \geq 0$ and $P_L(x_1, \ldots, x_n) \leq \sum_{j=1}^{m} a_j$ for $0 \leq x_i \leq 1$.

We now show by induction on the length m of L that the inequality (2) constructed by the above method satisfies (1).

Base Cases: Let $m = 1$. Then either $L =< (x_i, 1) >$ or $L =< (\overline{x}_i, 1) >$. The linear inequalities constructed for these two cases by the above rules are $x_i \geq 1$ and $1 - x_i \geq 1$, respectively, and these inequalities clearly satisfy (1).

Inductive Step: Let $L' = < (l_2, b_2), \ldots, (l_m, b_m) >$ — i.e., L' is simply L with the first item deleted. Then by the inductive hypothesis, there is a linear function $P_{L'}(x_1, \ldots, x_n)$ and a threshold τ' constructed by the above rules and satisfying

$$(3) \quad P_{L'}(x_1, \ldots, x_n) \geq \tau' \iff L'(x_1, \ldots, x_n) = 1.$$

Consider the following cases:

Case 1. Suppose $(l_1, b_1) = (x_i, 1)$. Then by application of the 1-Rule and the τ-Rule, we obtain the linear inequality

$$P_L(x_1, \dots, x_n) = \tau' x_i + P_{L'}(x_1, \dots, x_n) \geq \tau'.$$

Now suppose that $L(x_1, \dots, x_n) = 1$. Then either $x_i = 1$, or $x_i = 0$ and $L'(x_1, \dots, x_n) = 1$. If $x_i = 1$ then $P_L(x_1, \dots, x_n) \geq \tau'$ since $P_{L'}(x_1, \dots, x_n) \geq 0$. If $L'(x_1, \dots, x_n) = 1$, then $P_L(x_1, \dots, x_n) \geq \tau'$ since $P_{L'}(x_1, \dots, x_n) \geq \tau'$ by (3). On the other hand, if $L(x_1, \dots, x_n) = 0$ then $x_i = 0$ and $P_{L'}(x_1, \dots, x_n) < \tau'$ by (3), hence $P_L(x_1, \dots, x_n) < \tau'$.

Case 2. Suppose $(l_1, b_1) = (\overline{x}_i, 1)$ Then by application of the 1-Rule and the τ-Rule, we obtain

$$P_L(x_1, \dots, x_n) = \tau'(1 - x_i) + P_{L'}(x_1, \dots, x_n) \geq \tau'$$

and the analysis is similar to that of Case 1.

Case 3. Suppose $(l_1, b_1) = (x_i, 0)$. Then by application of the 0-Rule and the τ-Rule, we obtain the inequality

$$(4) \quad P_L(x_1, \dots, x_n) = (1 + \sum_{j=2}^{m} a_j)(1 - x_i) + P_{L'}(x_1, \dots, x_n) \geq (1 + \sum_{j=2}^{m} a_j) + \tau'$$

Now if $L(x_1, \dots, x_n) = 1$ then we must have $x_i = 0$ and $L'(x_1, \dots, x_n) = 1$. Hence using (3) we see that (4) is satisfied. Conversely, suppose that (4) is satisfied. Then $x_i = 0$ since $P_{L'}(x_1, \dots, x_n) \leq \sum_{j=2}^{m} a_j$ and $\tau' \geq 0$ always. Hence $P_{L'}(x_1, \dots, x_n) \geq \tau'$, so $L'(x_1, \dots, x_n) = 1$ by (3), and thus $L(x_1, \dots, x_n) = 1$.

Case 4. Suppose $(l_1, b_1) = (\overline{x}_i, 0)$. Then by the 0-Rule and the τ-Rule, we obtain

$$P_L(x_1, \dots, x_n) = (1 + \sum_{j=2}^{m} a_j)x_i + P_{L'}(x_1, \dots, x_n) \geq (1 + \sum_{j=2}^{m} a_j) + \tau'$$

and the analysis is similar to Case 3.

Thus, 1-Decision Lists are linearly separable, and VCdim(1DL) $= O(n)$.

To see that VCdim(kDL) $= O(n^k)$, we use the following transformation: For each of the $p(n) = \Theta(n^k)$ monomials M_i of length at most k, create a variable $y_i(x_1, \dots, x_n)$ defined by

$$y_i(x_1, \dots, x_n) = 1 \iff M_i(x_1, \dots, x_n) = 1.$$

Then under this transformation, each monomial M_i is mapped to a variable y_i in $\Theta(n^k)$ dimensions, so a k-Decision List L is mapped to a 1-Decision List $\tilde{L}$ in the variables $\{y_i\}$. This mapping has the property that $L(x_1, \ldots, x_n) = 1 \iff \tilde{L}(y_1, \ldots, y_{p(n)}) = 1$. Furthermore, by the above proof the image decision lists are linearly separable in $p(n)$-dimensional space, so VCdim(kDL) $= O(p(n)) = O(n^k)$. Similar transformation techniques can be found in [KLPV87] and [L88]. $\square$

8 References

[AL86] Angluin, D., P. Laird, "Identifying kCNF formulas from noisy examples", *Machine Learning* 2(4), 1988, pp. 319-342.

[AV79] Angluin, D., L.G. Valiant, "Fast probabilistic algorithms for Hamiltonian circuits and matchings", *Journal of Computer and Systems Sciences*, 18, 1979, pp. 155-193.

[BH88] Baum, E., D. Haussler, "What size net gives valid generalization?", to appear in *I.E.E.E. Conference on Neural Information Processing Systems*, Denver, CO, 1988.

[BI88] Benedek, G.M., A. Itai, "Learnability by fixed distributions", *First Workshop on Computational Learning Theory*, M.I.T., 1988.

[BEHW86] Blumer, A., A. Ehrenfeucht, D. Haussler, M. Warmuth, "Classifying learnable geometric concepts with the Vapnik-Chervonenkis dimension", *18th ACM Symposium on the Theory of Computing*, Berkeley, CA, 1986, pp. 273-282.

[BEHW87a] Blumer, A., A. Ehrenfeucht, D. Haussler, M. Warmuth, "Occam's Razor", *Information Processing Letters*, 24, 1987, pp. 377-380.

[BEHW87b] Blumer, A., A. Ehrenfeucht, D. Haussler, M. Warmuth, "Learnability and the Vapnik-Chervonenkis dimension", technical report UCSC-CRL-87-20, U.C. Santa Cruz, 1987, and to appear in *JACM.*

[H88] Haussler, D., "Quantifying Inductive Bias: AI Learning Algorithms and Valiant's Model", *Artificial Intelligence*, 36, 1988, pp. 177-221.

[HLW87] Haussler, D., N. Littlestone, M. Warmuth, "Predicting $\{0,1\}$-functions on randomly drawn points", to appear in *29th I.E.E.E. Symposium on Foundations of Computer Science*, White Plains, NY, 1988.

[HW87] Haussler, D., E. Welzl, "Epsilon-nets and simplex range queries", *Discrete Computational Geometry*, 2, 1987, pp. 127-151.

[K84] Karmarkar, N., "A new polynomial-time algorithm for linear programming", *Combinatorica*, 4, 1984, pp. 373-395.

[KL87] Kearns, M., M. Li, "Learning in the presence of malicious errors", *20th ACM Symposium on the Theory of Computing*, Chicago, IL, 1988, pp. 267-280.

[KLPV87] Kearns, M., M. Li, L. Pitt, L. Valiant, "On the learnability of Boolean formulae", *19th ACM Symposium on the Theory of Computing*, New York, NY, 1987, pp. 285-295.

[KV88] Kearns, M., L.G. Valiant, "Learning Boolean formulae or finite automata is as hard as factoring", technical report TR-14-88, Aiken Computation Laboratory, Harvard University, 1988.

[K79] Khachiyan, L.G., "A polynomial algorithm for linear programming", *Doklady Akademiia Nauk SSSR*, 244:S, 1979, pp. 191-194.

[L88] Littlestone, N., "Learning quickly when irrelevant attributes abound: a new linear threshold algorithm", *Machine Learning* 2(4), 1988, pp. 245-318.

[LMR88] Linial, N., Y. Mansour, R.L. Rivest, "Results on learnability and the Vapnik-Chervonenkis dimension", *First Workshop on Computational Learning Theory*, M.I.T., 1988, and to appear in *29th I.E.E.E. Symposium on Foundations of Computer Science*, White Plains, N.Y., 1988.

[N87] Natarajan, B.K., "On learning Boolean functions", *19th A.C.M. Symposium on the Theory of Computing*, New York, NY, 1987, pp. 296-304.

[PV86] Pitt, L., L.G. Valiant, "Computational limitations on learning from examples", technical report TR-05-86, Aiken Computation Laboratory, Harvard University, 1986.

[R87] Rivest, R., "Learning decision lists", *Machine Learning* 2(3), 1987, pp. 229-246.

[S88] Shvayster, H., "Non-learnable classes of Boolean formulae that are closed under variable permutation", *First Workshop on Computational Learning Theory*, M.I.T., 1988.

[V84] Valiant, L.G., "A theory of the learnable", *Comm. ACM*, 27(11) 1984, pp. 1134-1142.

[V85] Valiant, L.G., "Learning disjunctions of conjunctions" *Proc. 9th IJCAI*, Los Angeles, CA, 1985, pp. 560-566.

[V82] Vapnik, V.N., "Estimation of dependences based on empirical data", Springer Verlag, New York, 1982.

[VC71] Vapnik, V.N., A.Ya. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities", *Th. Prob. and its Appl.*, 16(2), 1971, pp. 264-280.

[VL88] Vitter, J.S., J. Lin, "Learning in parallel", *First Workshop on Computational Learning Theory*, M.I.T., 1988.

[WD81] Wencour, R.S., R.M. Dudley, "Some special Vapnik-Chervonenkis classes", *Discrete Math*, 33, 1981, pp. 313-318.