Classifying Learnable Geometric Concepts with the Vapnik-Chervonenkis Dimension

(extended abstract)

Anselm Blumer[1)]
Andrzej Ehrenfeucht[2)]
David Haussler[3)]
Manfred Warmuth[4)]

[1)] and [3)] Department of Mathematics and Computer Science, University of Denver, Denver, Colorado 80208.

[2)] Department of Computer Science, University of Colorado, Boulder, Colorado 80302.

[4)] Department of Computer and Information Sciences, University of California, Santa Cruz, California 95064.

Abstract We extend Valiant's learnability model to learning classes of concepts defined by regions in Euclidean space E^r. Our methods lead to a unified treatment of some of Valiant's results, along with previous results of Pearl and Devroye and Wagner on distribution-free convergence of certain pattern recognition algorithms. We show that the essential condition for distribution-free learnability is finiteness of the Vapnik-Chervonenkis dimension, a simple combinatorial parameter of the class of concepts to be learned. Using this parameter, we analyze the complexity and closure properties of learnable classes.

Authors A. Blumer and D. Haussler gratefully acknowledge the support of NSF grant IST-8317918, author A. Ehrenfeucht the support of NSF grant MCS-8305245, and author M. Warmuth the support of the Faculty Research Committee of the University of California at Santa Cruz. Part of this work was done while A. Blumer was visiting the University of California at Santa Cruz and M. Warmuth the University of Denver.

1. Introduction

Valiant has recently introduced a new complexity-based model of learning from examples and illustrated this model by exhibiting and analyzing several learning algorithms for classes of Boolean functions [V84,V85]. In this paper we extend Valiant's model to learning concepts defined by regions in Euclidean space E^r. Our methods lead to a unified treatment of some of Valiant's results, along with previous results of Pearl [P78,P79] and Devroye and Wagner [DW76].

In learning a class C of concepts from examples, a single concept is selected from C and we are given a finite set of points that are in the concept (positive examples) and a finite set of points that are outside the concept (negative examples). This entire set is called a *labeled sample*. A learning algorithm for C is a function that, given a labeled sample of a concept in C, returns a region in E^r that is its hypothesis as to which concept the sample represents.

As in [V84], we let P be a fixed probability distribution on E^r and assume that samples are created by drawing points at random based on this distribution. After drawing m points, the algorithm forms a hypothesis. The *error* of this hypothesis is taken to be the probability that it disagrees with the real concept on the next point drawn, i.e. the error is just the probability of the region that represents the symmetric difference between the hypothesis and the real concept. What we want in a learning algorithm is the following:

(1) For large enough m we should get arbitrarily small error with arbitrarily high probability, no matter which concept from C we are trying to learn. The bounds on m should be independent of the underlying

distribution P.

(2) In addition, the bounds on m should be polynomial in the inverse of the error probabilities, and the computation of the hypothesis should be polynomial in the length of the sample.

A class of concepts that fulfills (1) is called *learnable*. If (2) holds as well then the class is *polynomially learnable*. Condition (1) is formalized by demanding error greater than ε with probability at most δ for small ε and δ, uniformly for all concepts in C. Condition (2) implies that the number of required samples is a function $m(\varepsilon,\delta)$ that grows polynomially in $1/\varepsilon$ and $1/\delta$.

Our main result gives necessary and sufficient conditions on a class of concepts C for the existence of a learning algorithm satisfying (1). Our criterion is a simple combinatorial one, based on the work of Vapnik and Chervonenkis on uniform approximation by empirical distributions [VC71]. This approach was pioneered in [DW76] [P78]. We define a parameter called the *Vapnik–Chervonenkis dimension* of the class C of concepts[1] and show that there is a learning algorithm satisfying (1) if and only if this dimension is finite. Moreover, if C does have finite dimension d then there is a learning algorithm for C, uniformly achieving error no more than ε with probability at least $1-\delta$, using sample size $m(\varepsilon,\delta) = \max\left[\frac{4}{\varepsilon}\log\frac{2}{\delta}, \frac{8d}{\varepsilon}\log\frac{8d}{\varepsilon}\right]$. Hence the polynomial restriction on the growth of $m(\varepsilon,\delta)$ is automatic in this case. In fact *any* function from labeled samples into the class C that gives hypotheses consistent with the sample is a learning algorithm for C using this many samples. As a corollary, we obtain significantly better distribution-free bounds on the convergence of many well-known pattern recognition algorithms including the classic perceptron learning algorithm and its successors.[2]

In relating the Vapnik-Chervonenkis dimension to learnability, we use the concept of an ε-net for C with respect to a fixed distribution P. This is a finite set of points N such that every concept in C of probability greater than ε contains a point in N. This generalizes the ε-net idea from [HW85], which lead to improved time bounds for half-space range queries in all Euclidean spaces with dimension greater than one. The key to both the query and the learning results is to prove bounds on the frequency of occurrence of small ε-nets for classes of concepts (ranges) with finite Vapnik-Chervonenkis dimension. We are confident that this will have other applications in computational geometry besides those given in *[HW85]*.

Finally we deal with the case of polynomial learnability. Following Pearl [P78] and Valiant [V84], we introduce the idea of the complexity, n, of a concept and consider the learning problem for a sequence of concept classes $\{C_n\}_{n\geq 1}$, where C_n represents the set of concepts of complexity at most n. We show that if the Vapnik-Chervonenkis dimension of C_n grows faster than polynomial in n then this sequence of concepts classes is not learnable in the sense of Valiant. On the other hand, we identify certain general strategies related to the principle of preferring the simpler hypothesis (Occam's Razor) that in many cases give learning algorithms that satisfy conditions (1) and (2). (Condition (2) is modified to include polynomial dependence on the complexity n.) These results are illustrated with several learning algorithms for various concept classes. Our analysis also leads to a host of open problems, some of which are outlined in the final section.

Notation: $S \Delta T$ denotes the symmetric difference of sets S and T. For $S \subseteq X$, I_S denotes the indicator function for S on X, $I_S(x) = 1$ if $x \in S$, $I_S(x) = 0$ otherwise. X^m denotes the m-fold Cartesian product of X. Elements of X^m will be denoted by barred variables, e.g. $\bar{x}$ or $\bar{y}$. We assume $\bar{x}$ denotes $(x_1,\ldots,x_m)$ where $x_i \in X$, $1 \leq i \leq m$, when m is clear from the context, and similarly for other barred variables. If P is a probability measure on X then P^m denotes the m-fold product probability measure on X^m. Natural logarithms are written "ln", all other logarithms are base 2.

2. Learnability

We will use the following notions of learning algorithms and learnability.

Definition. A *concept class* is a set $C \subseteq 2^X$ of *concepts*. In this paper it is assumed that X is a fixed set, usually finite, countable, or E^r (Euclidean r-dimensional space) for some $r \geq 1$. In the latter case, we assume that each $c \in C$ is a Borel set. Elements of X^m will be called *m-samples* (or just *samples*). A *labeled m-sample* is an m-tuple $(<x_1,a_1>, \ldots, <x_m,a_m>)$, where $a_i \in \{0,1\}$, $1 \leq i \leq m$, and $a_i = a_j$ if $x_i = x_j$. For $\bar{x} = (x_1,\ldots,x_m) \in X^m$, the *labeled m-sample of $c \in C$ generated by $\bar{x}$* is given by $sam_c(\bar{x}) = (<x_1,I_c(x_1)>, \ldots, <x_m,I_c(x_m)>)$. The *sample space* of C, denoted S_C, is the set of all labeled m-samples over all $c \in C$ and all $\bar{x} \in X^m$ for all $m \geq 1$.

A denotes the set of all functions $A : S_C \to H$,

[1] This number is one less than the number V(C) defined in [D78].

[2] Previous bounds were $O\left(\max\left(\frac{1}{\varepsilon^2}\log\frac{1}{\varepsilon}, \frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)\right)$ [DW76], obtained using the results of [VC71] directly.

where H is a set of Borel sets on X. (In fact, we would like the range of A to be C, but we may gain some power by allowing A to approximate concepts in C.) Elements in H are called *hypotheses*. $A \in \mathbf{A}$ is *consistent* if its hypothesis always agrees with the sample, i.e. whenever $h = A(< x_1,a_1 > \cdots < x_m,a_m >)$ then for all i, $1 \le i \le m$, $a_i = I_h(x_i)$. For any $A \in \mathbf{A}$, probability measure P on X, $c \in C$, and $\overline{x} \in X^m$, the *error* of A for concept c on sample $\overline{x}$ (w.r.t. P) is given by $error_{A,c,P}(\overline{x}) = P(c \Delta A(sam_c(\overline{x})))$. Thus A's error is measured as the probability of the region that forms the symmetric difference between the real concept and A's hypothesized concept, which is just the probability that A's hypothesis will be inconsistent with the real concept on a randomly drawn point (w.r.t. P).

Let $m(\varepsilon,\delta)$ be an integer-valued function of ε and δ for $0 < \varepsilon, \delta < 1$ and P be a probability measure on X. $A \in \mathbf{A}$ is a *learning algorithm for C (w.r.t. P) with sample size $m(\varepsilon,\delta)$* if for all $0 < \varepsilon,\delta < 1$ and for all $c \in C$, $P^{m(\varepsilon,\delta)}(error_{A,c,P}(\overline{x}) > \varepsilon) \le \delta$, where $\overline{x} \in X^{m(\varepsilon,\delta)}$. Thus we insist that using a randomly drawn sample of size $m(\varepsilon,\delta)$, any concept in C can be learned with error no more than ε with probability at least $1-\delta$. If A is defined in such a way that $Z = \{\overline{x} \in X^{m(\varepsilon,\delta)} : error_{A,c,P}(\overline{x}) > \varepsilon\}$ is not measurable for some $c \in C$, then we require that there exist a measurable Z' such that $Z \subseteq Z'$ and $P^{m(\varepsilon,\delta)}(Z') \le \delta$. We say $A \in \mathbf{A}$ is a *learning algorithm for C with sample size $m(\varepsilon,\delta)$* only when A is a learning algorithm for C w.r.t. P with sample size $m(\varepsilon,\delta)$ for *all* probability measures P on X. If such an A exists, we say C is *learnable*.

Example. Consider the problem of learning concepts such as the concept of "medium build", defined (for men) as having weight between 150 and 185 pounds and height between 5' 4'' and 6' 0''. By looking at a finite database of randomly chosen men that gives their weight, height and classification (medium build or not), we want to form a rule that approximates the true concept of medium build, and we want our approximation to be accurate independently of the underlying distribution on height-weight pairs (height and weight values are *not* assumed to be independent). This type of learning problem is formalized and solved as follows.

Let X be E^2, C be the set of all orthogonal closed rectangles, (i.e. products of closed intervals on the x-axis with closed intervals on the y-axis) and P be any arbitrary probability distribution on E^2. A simple algorithm to learn a concept $c \in C$ is the following. Keep track of the minimum and maximum x and y coordinates of any positive sample point. Let l', r', b', t' denote these four coordinates. After drawing a number of sample points, predict that the concept is $h = [l',r'] \times [b',t']$. If no positive samples are drawn, let $h = \emptyset$. Call this algorithm A.

Theorem 1. *A is a learning algorithm for C with sample size $\frac{4}{\varepsilon} ln(\frac{4}{\delta})$.*

Proof. Assume the concept c to be learned is the product of the intervals $[l,r]$ on the x axis and $[b,t]$ on the y axis. Since A's hypothesis h is always contained in c, if $P(c) < \varepsilon$ then $error_{A,c,P}$ of any labeled sample of c is always less than ε. Otherwise we define four minimal side rectangles within c that each cover an area of probability at least $\frac{\varepsilon}{4}$:

$$left := min(\{[l,x] \times [b,t] : P([l,x] \times [b,t]) \ge \tfrac{\varepsilon}{4}\})$$

and *right*, *bottom*, and *top* are defined similarly. If the sample size is m, the probability that a particular side rectangle contains no sample point is at most $(1-\varepsilon/4)^m$. The probability that some side rectangle contains no sample point is bounded by 4 times this quantity. This latter quantity is smaller than $4e^{-m\varepsilon/4}$, so if the sample size is at least $\frac{4}{\varepsilon} ln(\frac{4}{\delta})$, then with probability at least $1-\delta$ we will draw a sample point in each of these four rectangles. If this occurs, then the probability of the region given by the symmetric difference of A's hypothesis and c will be less than ε, thus bounding the error of the hypothesis. Since m is independent of the particular distribution P, the result follows. □

This proof readily generalizes to r-dimensional rectangles for $r > 2$, with a bound of $\frac{2r}{\varepsilon} \ln(\frac{2r}{\delta})$, on the sample size. However, it is not clear how to generalize it even in 2 dimensions to other types of concepts, e.g. circles, half-planes or rectangles of arbitrary orientation. To show that these classes are also learnable, we use a concept first introduced in [VC71].

Definition. Given a concept class C and finite $S \subseteq X$, $\Pi_C(S)$ denotes the set of all subsets of S that can be obtained by intersecting S with a concept in C, i.e. $\Pi_C(S) = \{S \cap c : c \in C\}$. If $\Pi_C(S) = 2^S$, then we say that S is *shattered* by C. The *Vapnik-Chervonenkis dimension* of C (or simply the *dimension* of C) is the smallest integer d such that no $S \subseteq X$ of cardinality $d+1$ is shattered by C. If no such d exists, the dimension of C is infinite.

[VC71], [D78], [WD81], [A83] and [HW85] give numerous examples of concept classes of finite dimension.[3] For example, the dimension of the set of all

[3]When C is of finite dimension, Dudley calls C a Vapnik-Chervonenkis Class (VCC) [D78], [WD81]. The Vapnik-Chervonenkis number of this class, denoted V(C), corresponds to the dimension of C plus one.

orthogonal rectangles in E^r as defined above is $2r$, and the dimension of half-spaces and balls of E^r is $r+1$. Restricting ourselves to the real line, the set of all half-lines open to the left has dimension 1, the set of all intervals has dimension 2 and in general, the set of all sets of at most k intervals has dimension $2k$. In general, whenever C is of finite dimension, then C_k, the set of all Boolean combinations formed from at most k concepts in C, is also of finite dimension [D78]. Thus for example, since the set C_k of k-gons in E^r for fixed $k > r$ is formed by k-fold intersections of half-spaces, C_k is of finite dimension for any finite k. Wenocur and Dudley [WD81] also prove more general results that imply e.g. that the concept class formed by set of all half-spaces bounded by polynomial curves of fixed degree also has finite dimension. On the other hand, the set of all finite sets of intervals, like the set of all open sets or all Borel sets, obviously has infinite dimension.

Definition. A concept class C is *trivial* if C is one concept, or two disjoint concepts.

It is clear that a sample size of one is the most that is required to learn C when C is trivial.

We can now state our main result.

Theorem 2. The following are equivalent for all non-trivial, well-behaved[4] concept classes C:

(i) The dimension of C is finite.

(ii) C is learnable.

(iii) If d is the dimension of C,

(a) for sample size greater than

$$\max\left[\frac{4}{\varepsilon}\log\frac{2}{\delta}, \frac{8d}{\varepsilon}\log\frac{8d}{\varepsilon}\right],$$

any consistent function $A : S_C \to H$ is a learning algorithm for C and

(b) for $\varepsilon < \frac{1}{2}$ and sample size less than

$$\max\left[\frac{1}{2\varepsilon}\log\frac{1}{\delta}, d(1-2(\varepsilon(1-\delta)+\delta))\right]$$

no function $A : S_C \to H$, where $C \subseteq H$ is a learning algorithm for C.

The proof will be given in a series of lemmas and theorems. We begin by showing that (i) implies (iii(a)). The critical property of finite dimensional concept classes we use is the following:

Definition. For all $d \geq 0$ and $m \geq 1$, $\Phi_d(m) = \sum_{i=0}^{d}\binom{m}{i}$ if $m \geq d$, $\Phi_d(m) = 2^m$ otherwise.

Proposition 3. (a) If the dimension of C is d then $|\Pi_C(S)| \leq \Phi_d(m)$ for all $S \subseteq X$ of cardinality m.

(b) $\Phi_d(m) \leq m^d + 1$ for all $d \geq 0$ and $m \geq 1$. $\Phi_d(m) \leq m^d$ for all $d \geq 2$ and $m \geq 2$.

A short proof of part (a) above can be found in [HW85] or (along with its history of independent discoveries) in [A83]. Part (b) is easily verified.

To characterize learnable concept classes, we use the following, which generalizes the corresponding notion in [HW85].

Definition. For any concept class C, probability measure P on X, and $\varepsilon > 0$, $C_{P,\varepsilon} = \{c \in C : P(c) > \varepsilon\}$. $S \subseteq X$ is an ε-*net* for C (w.r.t. P) if S contains a point in every $c \in C_{P,\varepsilon}$.

Examples. If X is the interval [0,1], P is the uniform distribution and C is the set of closed intervals in X, then the set of all points εk, $0 \leq k \leq \frac{1}{\varepsilon}$, is an ε-net for C for any $\varepsilon > 0$. In fact C has a net of this size for any distribution P on X. On the other hand, if C is all open sets, then clearly there are no finite ε-nets for C w.r.t. the uniform distribution.

For the next two lemmas and the following theorem, let C be a fixed well-behaved concept class and P be a fixed probability measure on X. The proofs of these lemmas are analogous to those of Lemma and Theorem 2 of [VC71]. These results generalize Lemmas 3.5 and 3.6 and Theorem 3.7 of [HW85].

Definition. For any $m \geq 1$ and $\varepsilon > 0$, Q_ε^m denotes the set of all $\bar{x} \in X^m$ such that the set of distinct elements of $\bar{x}$ do not form an ε-net for C w.r.t. P, i.e. such that there exists $c \in C_{P,\varepsilon}$ with $\bar{x} \cap c = \emptyset$. J_ε^{2m} denotes the set of all $\bar{x}\bar{y} \in X^{2m}$, where $\bar{x},\bar{y} \in X^m$, such that there exists $c \in C_{P,\varepsilon}$, where $\bar{x} \cap c = \emptyset$ and $|\{i : y_i \in c, 1 \leq i \leq m\}| \geq \frac{\varepsilon m}{2}$, i.e. no element of c occurs in the first half of the sequence, but elements of c occur with frequency at least $\frac{\varepsilon m}{2}$ in the second half.

Lemma 4. For any $\varepsilon > 0$ and $m \geq \frac{8}{\varepsilon}$, $P^m(Q_\varepsilon^m) \leq 2P^{2m}(J_\varepsilon^{2m})$.

Lemma 5. If C is of finite dimension d, $P^{2m}(J_\varepsilon^{2m}) \leq \Phi_d(2m)2^{\frac{-\varepsilon m}{2}}$ for all $m \geq 1$.

Proof. For each j, $1 \leq j \leq (2m)!$ let π_j be a distinct permutation of the indices 1, ..., $2m$. It is clear that

$$P^{2m}(J_\varepsilon^{2m}) = \int_{X^{2m}} I_{J_\varepsilon^{2m}}(\bar{x})dP^{2m} = \int_{X^{2m}} I_{J_\varepsilon^{2m}}(\pi_j(\bar{x}))dP^{2m}$$

for all permutations π_j. Hence

$$P^{2m}(J_\varepsilon^{2m}) = \int_{X^{2m}} \frac{1}{(2m)!}\sum_{j=1}^{(2m)!} I_{J_\varepsilon^{2m}}(\pi_j(\bar{x}))dP^{2m}$$

Thus it suffices to show that

$$\frac{1}{(2m)!}\sum_{j=1}^{(2m)!} I_{J_\varepsilon^{2m}}(\pi_j(\bar{x})) \leq \Phi_d(2m)2^{\frac{-\varepsilon m}{2}}$$

for all $\bar{x} \in X^{2m}$.

Consider a fixed $\bar{x} \in X^{2m}$. Let S be the set of distinct elements of X that appear in $\bar{x}$. Since $|S| \leq 2m$

[4] See Appendix.

and C is of dimension d, there are at most $\Phi_d(2m)$ distinct subsets of S induced by intersections with $c \in C_{P,\varepsilon}$ (by Proposition 3 part (a)). Each such subset T of S is a witness to the fact that certain permutations of $\bar{x}$ are in J_ε^{2m}. Specifically, whenever all occurrences of members of T appear in the second half of $\pi_j(\bar{x})$ and there are at least $\frac{\varepsilon m}{2}$ such occurrences, then $\pi_j(\bar{x}) \in J_\varepsilon^{2m}$, otherwise $\pi_j(\bar{x}) \notin J_\varepsilon^{2m}$. However, for a given T this can occur in only a small fraction of all permutations of $\bar{x}$. In particular, if there are l occurrences of members of T in $\bar{x}$, then T is a witness for at most

$$\frac{\binom{m}{l}}{\binom{2m}{l}} = \frac{m(m-1)\ldots(m-l+1)}{2m(2m-1)\ldots(2m-l+1)} \leq 2^{-l} \leq 2^{\frac{-\varepsilon m}{2}}$$

of all permutations of $\bar{x}$. It follows that

$$\frac{1}{(2m)!}\sum_{j=1}^{(2m)!} I_{J_\varepsilon^{2m}}\pi_j(\bar{x}) \leq \Phi_d(2m)2^{\frac{-\varepsilon m}{2}}.$$

□

Directly from the above two lemmas we get

Theorem 6. If C has finite dimension d and $m \geq \frac{8}{\varepsilon}$ then $P^m(Q_\varepsilon^m) \leq 2\Phi_d(2m)2^{\frac{-\varepsilon m}{2}}$.

Since the negative exponential term dominates the polynomial $\Phi_d(2m)$, this shows that the probability of not getting an ε-net for C in an m-sample goes very rapidly to zero for large m. The m required for this probability to be less than δ is estimated in the following.

Lemma 7. If

$$m \geq \max\left\{\frac{4}{\varepsilon}\log\frac{2}{\delta}, \frac{8d}{\varepsilon}\log\frac{8d}{\varepsilon}\right\} \text{ then } 2\Phi_d(2m)2^{\frac{-\varepsilon m}{2}} \leq \delta.$$

Theorem 8. If C is well-behaved and has finite dimension d then for any distribution P on X and any $0 < \varepsilon,\delta \leq 1$, if $m \geq \max\left\{\frac{4}{\varepsilon}\log\frac{2}{\delta}, \frac{8d}{\varepsilon}\log\frac{8d}{\varepsilon}\right\}$ points in X are drawn at random according to P then the probability that these points do not form an ε-net for C w.r.t. P is at most δ.

Proof. Follows directly from Lemma 7 and Theorem 6. □

To complete the proof that (i) implies (iii(a)), we need the following.

Lemma 9. For any $z \subseteq X$, the dimension of C equals the dimension of $C_{\Delta z}$, where $C_{\Delta z} = \{c \Delta z : c \in C\}$.

Proof. This follows from the result in [*W86*] that the dimension of C is unchanged if for any $x \in X$, each $c \in C$ is replaced by $c \cup \{x\}$ if $x \notin c$ and $c - \{x\}$ if $x \in c$. □

Proof that (i) implies (iii(a)). Let $A : S_C \to C$ be a consistent function in A, P be a distribution on X and c be a fixed concept in C to be learned. If the dimension of C is d, then the dimension of $C_{\Delta c}$ is d by the above lemma, where $C_{\Delta c} = \{c \Delta c' : c' \in C\}$. Hence by Theorem 8, if we draw a sample of size $m \geq \max\left\{\frac{4}{\varepsilon}\log\frac{2}{\delta}, \frac{8d}{\varepsilon}\log\frac{8d}{\varepsilon}\right\}$, we get an ε-net for $C_{\Delta c}$ w.r.t P with probability at least $1 - \delta$ for any distribution P. If we get an ε-net for $C_{\Delta c}$ then for any $c' \in C$ produced by A on the corresponding labeled sample we must have $P(c \Delta c') \leq \varepsilon$, otherwise the hypothesis would be contradicted by a sample point. Hence the error of A is at most ε with probability at least $1 - \delta$. □

Proof that (iii(a)) implies (ii). We can simply well-order the concepts in C and for each labeled sample choose the first concept that is consistent with this sample. (Note that we are not concerned with the computability of A here.) □

It remains to show that (i) implies (iii(b)) and (ii) implies (i).

Lemma 10. (i) implies (iii(b)): Let C be a non-trivial concept class of finite dimension d and $\varepsilon < \frac{1}{2}$. Then any learning algorithm for C must use sample of size at least $m \geq \max\left\{\frac{1}{2\varepsilon}\log\frac{1}{\delta}, d(1 - 2(\varepsilon(1-\delta)+\delta))\right\}$.

Proof. We begin with the first term of the lower bound, considering two cases for the non-trivial concept C. The first is that C contains two distinct concepts c_1 and c_2 that have a non-empty intersection. The second case is that all concepts in C are pairwise disjoint, and there are at least three concepts in C. We give the proof only for the first case, the other argument being similar.

Let A be a learning algorithm for C and let $a \in X$ be a point in $c_1 \cap c_2$ and $b \in X$ be a point in $(c_1 \cup c_2) - (c_1 \cap c_2)$. Without loss of generality, assume that $b \in c_2$. Let P be the probability distribution such that $P(b) = \varepsilon$, $P(a) = 1 - \varepsilon$ and $P(x) = 0$ for any other point $x \in X$. With respect to this distribution, we can effectively replace X with the set $\{a,b\}$ and H with the four subsets of $\{a,b\}$, modifying A accordingly.

It is easily verified that if the sample size m is less than $\frac{1}{-\log(1-\varepsilon)}\log\frac{1}{\delta}$, then the probability of drawing the point a each time is greater than δ. Since $\varepsilon < \frac{1}{2}$, this holds for $m < \frac{1}{2\varepsilon}\log\frac{1}{\delta}$. Assume the concept to be learned is either $\{a\}$ or $\{a,b\}$. We can divide the possible learning algorithms for C into four types, depending on how they respond to a labeled m-sample in which each point is a (the label will always be 1). If the algorithm responds by producing the hypothesis $\{a\}$, then for this sample it has error ε if the concept to be learned is $\{a,b\}$. If it responds $\{a,b\}$, $\{b\}$ or $\emptyset$

then it has error at least ε if the concept to be learned is $\{a\}$. In any case, there is a concept in C such that the probability that A produces a hypothesis of error at least ε with sample size m is greater than δ. By increasing the probability of a slightly and decreasing the probability of b by the same amount it follows that that the previous statement also holds for error strictly larger than ε instead of error at least ε. We conclude A cannot be a learning algorithm for C with sample size $m < \frac{1}{2\varepsilon}\log\frac{1}{\delta}$.

For the second term in the lower bound, note that since C is non-trivial, the dimension d of C is at least 1. There must exist a set X_d of d points in X that are shattered by C. Let the probability distribution P on X be uniform on these points and 0 everywhere else. With respect to this distribution, we may replace X with X_d and C with 2^{X_d}.

Suppose we draw an m-sample $\bar{x}$ of X and l different points are observed in this sample. For each of the 2^l possible labelings of $\bar{x}$, there are 2^{d-l} concepts consistent with this labeling. Whatever the hypothesis of the learning algorithm, for every point of X not observed in $\bar{x}$, it will be correct for exactly half these concepts. Thus, the average error of the learning algorithm on $\bar{x}$ over all concepts in C is at least $\frac{d-l}{2d} \geq \frac{d-m}{2d}$. This implies that the average error of the learning algorithm over all $\bar{x} \in X^m$ and all concepts in C is at least $\frac{d-m}{2d}$. Hence there must be a concept with average error at least $\frac{d-m}{2d}$. To make the frequency of m-samples in which the error on this concept is greater than ε at most δ, the error can be greater than ε (i.e. 1) on at most $\delta 2^m$ of the m-samples, and must be at most ε on the remainder. Hence the average error can be at most $\varepsilon(1-\delta)+\delta$, which gives the second part of the lower bound. □

Proof that (ii) implies (i) This follows from the second lower bound in the above lemma. This completes the proof of Theorem 2. □

The above theorem shows a significant gap in the complexity of geometric learning problems as measured by the sample size needed: either the set of concepts is learnable with sample size $O(\frac{1}{\varepsilon}\log\frac{1}{\varepsilon\delta})$ or it is not learnable at all. Furthermore, this result is easily extended to probabilistic learning algorithms (i.e. algorithms that for a given sample choose a hypothesis probabilistically) and to algorithms that use an oracle to query the values of particular points (as defined by Valiant [V84]). We need only observe that if the Vapnik-Chervonenkis dimension is not finite, then no matter how the m points are drawn in the proof of Lemma 10 and no matter how the hypothesis is formed from the values on these points, the algorithm can do no better than random guessing on the points that it has not seen.

Theorem 2 has immediate consequences for several well-known pattern recognition algorithms. For example, since the Vapnik-Chervonenkis dimension of positive half-spaces in E^r is r, the sample size required for the perceptron learning algorithm (see e.g. [YC74], pg. 120) to achieve error at most ε with probability at least $1-\delta$ is $\max\left[\frac{4}{\varepsilon}\log\frac{2}{\delta}, \frac{8r}{\varepsilon}\log\frac{8r}{\varepsilon}\right]$, independent of the underlying distribution governing the selection of samples. Since δ appears only in a log term, this implies that good linear separators can be found with very high probability from relatively small random samples. Furthermore, the same bound applies to *any* learning algorithm for half-spaces that eventually produces consistent hypotheses, e.g. the Ho-Kashyap algorithm [YC74] or Meggido's prune-and-search linear programming technique [M84]. The best previous distribution-free bound for these algorithms was $\frac{16}{\varepsilon^2}(r\ln\frac{16r}{\varepsilon^2}+\ln\frac{4}{\delta})$ [DW76], derived directly from the results given in [VC71].

In particular cases, such as the orthogonal rectangles example in Theorem 1, we can tighten and simplify the bounds on the sample size given in Theorem 2. The same is true for certain linear separator algorithms in E^2 [L85].

3. Occam's Razor

In this section we examine the computational feasibility of learning various concept classes of finite dimension and extend the learnability definitions of the previous sections by adding a parameter that defines the size or complexity of the concept to be learned [P78] [V84].

Definition. A concept class C is *polynomially recognizable* if there exists a polynomial time algorithm that, given any labeled m-sample, produces a concept $c \in C$ consistent with this sample, or returns *false* if no such concept exists. An algorithm of this type will be called a *recognition* algorithm. A concept class C is *polynomially learnable* if there exists an algorithm A and a function $m(\varepsilon,\delta)$, polynomial in $\frac{1}{\varepsilon}$ and $\frac{1}{\delta}$, such that A is a learning algorithm for C using $m(\varepsilon,\delta)$ samples and A runs in time polynomial in the length of the sample.

If C is polynomially recognizable and has finite dimension, Theorem 2 shows that C is polynomially learnable. Since finding a consistent hypothesis can often be reduced to linear programming, Meggido's results [M84] show that many interesting concept

classes have linear recognition algorithms. For example, if $X = E^2$ then the classes of circles, and in general concepts defined by curves of degree at most k (i.e. regions of the form $p(x,y) \leq 0$ for some k-degree polynomial p, fixed k) have linear recognition algorithms and hence are polynomially learnable by Theorem 2. Similar results hold for $X = E^r$ for fixed r, however the time complexity of Meggido's technique grows doubly exponentially in the dimension of the linear programming problem [LP84].

Definition. $\{C_n\}_{n \geq 1}$ will always denote a sequence of concept classes C_n with $C_n \subseteq C_{n+1}$, for all $n \geq 1$. $\{C_n\}_{n \geq 1}$ is *polynomially learnable* if there exists an algorithm A and a function $m(\varepsilon,\delta,n)$, polynomial in $\frac{1}{\varepsilon}, \frac{1}{\delta}$, and n, such that for all $n \geq 1$, A is a learning algorithm for C_n using $m(\varepsilon,\delta,n)$ samples and A runs in time polynomial in the length of the sample. If $C = \bigcup_{n \geq 1} C_n$, then C_n represents the set of all concepts of C of *complexity* at most n. We do not assume that a sample of a concept from $C_n \subseteq C$ contains any specific information about the index n.

If the dimension of C_n grows faster than polynomially in n then $\{C_n\}_{n \geq 1}$ is not polynomially learnable, by Theorem 2. This shows for example that when X is a countably infinite dimensional real space and for every $n \geq 1$, C_n is the set of all regions in the Boolean closure of at most n half-spaces in E^n, where for any point in $c \in C_n$, only the first n coordinates are nonzero, then C_n is not polynomially learnable. To see this, consider 2^n points at the corners of the n-cube. Any subset of these points can be defined as a Boolean combination of n–dimensional half-spaces whose defining hyperplanes are each normal to one of the first n coordinate axes. Thus these points are shattered by C_n and hence the dimension of C_n is at least 2^n. This also shows that if C_n is the set of all regions defined in a countably infinite dimensional discrete space by Boolean formulas of n variables, then $\{C_n\}_{n \geq 1}$ is not learnable.

On the other hand, if the dimension of C_n grows polynomially in n and there is an algorithm that, given an m-sample of a concept in C_n, always produces a consistent hypothesis in C_n and runs in time polynomial in the length of the sample, then Theorem 2 shows that $\{C_n\}_{n \geq 1}$ is polynomially learnable. For example, if C_n is the set of orthogonal rectangles in E^n (which has dimension $2n$), where, as above, for any point in $c \in C_n$, only the first n coordinates are nonzero, then $\{C_n\}_{n \geq 1}$ is polynomially learnable using the algorithm of Theorem 1. To show polynomial learnability when C_n is the set of n-dimensional half-spaces or balls, one must abandon Meggido's technique for finding consistent hypotheses in favor of a linear programming technique that is polynomial in both the number of constraints and the (linear programming) dimension, e.g. Karmarker's technique [K84]. However, this polynomial bound depends on restricting the real numbers involved to a fixed finite precision, which runs contrary to the usual model adopted in computational geometry (which we have also implicitly adopted here).

In a discrete space, Theorem 2 shows that whenever the cardinality of the set of functions that define the regions in C_n is $O(2^{p(n)})$ for some polynomial $p(n)$, then any polynomial algorithm that produces consistent hypotheses in C_n for any sample from a concept in C_n is a polynomial learning algorithm for $\{C_n\}_{n \geq 1}$. This is because when $|C_n| \leq 2^{p(n)}$, at most $p(n)$ points can be shattered by C_n, and hence the dimension of C_n is at most $p(n)$. This result applies, e.g., when C_n is the set of all functions of a particular type that can be described using at most $p(n)$ bits [P78] [V84]. As another example, let C_n be the set of all regions defined in a countably infinite dimensional discrete space by k-CNF (DNF) Boolean formulas of at most n variables ([V84]). Since each clause in a k-CNF (DNF) involves at most k variables, $|C_n| \leq 2^{((2n)^k)}$. This shows that for fixed k, any algorithm that produces a k-CNF (DNF) formula consistent with the sample in polynomial time (and in particular, the algorithm given by Valiant) is a polynomial learning algorithm for $\{C_n\}_{n \geq 1}$.

However there are simple examples where Theorem 2 does not apply directly. Let C_n be all unions of at most n (possibly overlapping) closed orthogonal rectangles in E^2. Lemma 13 (below) shows that the dimension of C_n is polynomial in n. However, given a set of points in E^2 labeled with 0's and 1's, it is NP-hard to determine the smallest n such that the set of 1-labeled points can be covered by n closed orthogonal rectangles, where none of these rectangles contains a 0-labeled point [M78]. Hence given positive and negative points from a concept in C_n for unknown n, it is NP-hard to determine the smallest n such that there exists a concept in C_n consistent with the sample. Any learning algorithm that always gives a consistent hypothesis in C_n for any sample of a concept in C_n implicitly solves this optimization problem.

The upshot of this is that to apply Theorem 2 directly, we need to find an efficient learning algorithm that works according to the principle of always preferring the simplest hypothesis that explains the data, usually called *Occam's Razor*. Yet in many cases this is not possible. To address the cases when it is not feasible to find the simplest hypotheses, we will show that it

suffices to settle for simpler rather than simplest hypotheses, i.e. it suffices to produce hypotheses that are significantly simpler than the sample data itself.

Definition. An *Occam algorithm* for $\{C_n\}_{n\geq 1}$ is an algorithm that, given any m-sample of a concept $c \in C_n$, produces in polynomial time a consistent hypothesis $h \in H_{n,m}$, where $H_{n,m}$ is a concept class that includes C_n and has dimension at most $f(n,m)$. $f(n,m)$ is called the *range dimension* of the Occam algorithm.

In this version of Occam, the dimension of the hypothesis class C_n is the measure of simplicity and the range dimension $f(n,m)$ tells how well the Razor is applied. If the dimension of $H_{n,m}$ is equal to that of C_n for all n and m then the Occam-algorithm implements the strictest form of Occam's Razor. An example of this occurs when $H_{n,m} = C_n$, as in Valiant's algorithm for learning k-CNF, or the algorithm given above for learning orthogonal rectangles in E^n. In this case the range dimension does not depend on m. For faster growing range dimensions, especially those that depend on m, the Occam-algorithm can take advantage of the larger range of hypotheses to simplify its search for a consistent hypothesis. The following shows how fast the range dimension can grow, still maintaining polynomial learnability.

Theorem 11. If A is an Occam algorithm for $\{C_n\}_{n\geq 1}$ with range dimension $f(m,n) \leq rn^k m^\alpha$, for some $k \geq 1$, $0 \leq \alpha < 1$ and $r \geq 2n^{-k}$, then A is a polynomial learning algorithm for $\{C_n\}_{n\geq 1}$ with sample size

$$m \geq \max\left(\frac{4}{\varepsilon}\log\frac{2}{\delta}, \left(\frac{8rn^k}{\varepsilon(1-\alpha)}\log\frac{8rn^k}{\varepsilon(1-\alpha)}\right)^{\frac{1}{1-\alpha}}\right)$$

If $f(m,n)$ is bounded by $rn^k(\log m)^l$, then the second bound on m becomes

$$\frac{2^{l+4}rn^k}{\varepsilon}\left(\log\frac{8(2l+2)^{l+1}rn^k}{\varepsilon}\right)^{l+1}$$

In some ways Theorem 11 can be viewed as showing a relationship between learning and data compression. In the discrete case, i.e. for learning Boolean functions, let $H_{n,m}$ be the set of all hypotheses that can be described with $n^k m^\alpha$ bits, for some $k \geq 1$ and $0 \leq \alpha < 1$. Then Theorem 11 says that if, for any m-sample of a function that can be described in n bits, we can effectively find a hypothesis that "explains" the sample and uses $O(n^k m^\alpha)$ bits, then we can learn. For fixed n, this amounts to a kind of data compression on the sample. Numerical bounds on the number of samples needed for learning algorithms in this discrete case that are slightly better than those given in Theorem 11 are derived in [BEHW85] using a simpler argument.

Theorem 11 can be used to show that many kinds of concept classes are learnable. In the remainder we will demonstrate how this is done when C_n is built by n-fold unions of concepts for a fixed class C_1 of finite dimension. Similar results for n-fold intersections follow by considering the class of n-fold unions of complements of concepts in C_1. From Lemma 9, the dimension of the complements of concepts in a class is the same as the dimension of the class itself.

Lemma 12. If C is polynomially recognizable and the dimension of C is finite then for any finite set $S \subseteq X$ the sets of $\Pi_C(S)$ can be listed in time polynomial in the cardinality of S.

Proof. Assume $S = \{x_1, x_2, ..., x_m\}$. The size of $\Pi_C(S)$ is polynomial in m by Proposition 3. To produce a list L of $\Pi_C(S)$ we proceed as follows. Initialize L to the one element list consisting of just the empty set. This corresponds to the case $m = 0$. Now by induction, assume that the list $L = \Pi_C(\{x_1, x_2, ..., x_i\})$ has been produced for some i, $0 \leq i < m$. L is updated to the list $\Pi_C(\{x_1, x_2, ..., x_{i+1}\})$ as follows. For each element T of L, test the sets T and $T \cup \{x_{i+1}\}$ for membership in $\Pi_C(\{x_1, x_2, ..., x_{i+1}\})$. Since C is polynomially recognizable, this can be done in polynomial time by creating the appropriate labeled samples and running the recognition algorithm on them. Now replace the element T in L with either one or both of these sets, according to the outcome of this test. (Note that it is possible that $T \in \Pi_C(\{x_1, x_2, ..., x_i\})$ but $T \notin \Pi_C(\{x_1, x_2, ..., x_{i+1}\})$.) The time for each complete update is polynomial since by Proposition 3 the size of L remains polynomial in m. Hence the entire procedure is polynomial. □

Lemma 13. If the dimension of C_1 is k and for all $n \geq 1$, C_n is the set of all n-fold unions of concepts in C_1 then the dimension of C_n is at most $2kn\log(kn)$.

Proof. Follows from Lemma 4.5 of [HW85]. □

The following theorem shows for example that n-fold unions of orthogonal rectangles in E^2 are polynomially learnable even though finding the smallest set of orthogonal rectangles that explains the data is NP-hard [M78].

Theorem 14. Assume C_1 is polynomially recognizable and the dimension of C_1 is finite. Let C_n be the set of all n-fold unions of concepts in C_1. Then $\{C_n\}_{n\geq 1}$ is polynomially learnable.

Proof. Let S be the set of points in a labeled m-sample of a concept in C_n. Our strategy will be to find a hypothesis consistent with S that is formed from the union of relatively few concepts in C_1, i.e. not many more than n. This problem can be formulated as a set

cover problem. The set to be covered is the set of positive points of S and the sets allowed in the cover are the elements of $\Pi_{C_1}(S)$ that contain only positive points. To find the smallest set cover is NP-hard [GJ78] and remains NP-hard for simple geometric versions such as covering with rectangles [M78]. Fortunately, there is a simple greedy algorithm [N69], [J74] that produces a cover using at most $n \ln p$ sets, where n is the minimum number of sets needed for any cover and p is the size of the set to be covered: pick the set that covers the largest number of points; after this pick the set that covers the largest number of points which haven't been covered previously, and so forth.

Since the sets of $\Pi_{C_1}(S)$ can be listed in polynomial time (Lemma 12), the largest set that contains only positive points can be found in polynomial time. Hence the above algorithm is polynomial in m and given any m-sample of a concept in C_n, produces a consistent hypothesis for this sample in $C_{n\ln(m)}$. Since the dimension of C_n is at most $2kn\log(kn)$ by Lemma 13, this gives an Occam-algorithm with range dimension $O(n\log(m)(\log n + \log\log m))$. Thus by Theorem 11, $\{C_n\}_{n\geq 1}$ is polynomially learnable. □

The above result can also be used to show that concept classes not directly formed by n-fold unions of concepts from a fixed class are learnable. For example, let $X = E^2$ and let C_n be the set of all concepts created in the Boolean closure of n half-planes. These correspond to subsets of the set of cells formed by an arrangement of at most n lines. Since there are $O(n^2)$ vertices in the arrangement, any subset of cells can be triangulated using $O(n^2)$ triangles (some possibly unbounded), hence each concept in C_n can be represented as a union of $O(n^2)$ triangles. Let C'_n be n-fold unions of triangles in E^2. It follows that $C_n \subseteq C'_{O(n^2)}$. Since the class of triangles is polynomially recognizable and has finite dimension (it is 7), $\{C'_n\}_{n\geq 1}$ is polynomially learnable by Theorem 14. It follows that $\{C_n\}_{n\geq 1}$ is polynomially learnable as well.

4. Open Problems

One major question of interest that remains is the polynomial learnability of $\{C_n\}_{n\geq 1}$ when C_n is built by n-fold unions of $\leq n$-dimensional orthogonal rectangles. This is of particular interest because a special case of this is the learning problem for DNF formulas of at most n variables and n clauses [V84]. A more restricted problem is the case that C_n is built by n-fold unions of n-dimensional positive orthogonal half-spaces. The discrete version of this problem is the learning problem for monotone DNF formulas. Valiant proved that this class is learnable provided that we are given certain oracles [V84], but it is not clear whether these formulas can be learned without oracles. It is shown in [*VP 86*] that if the sample is from a monotone boolean function that is the sum of two monomials, then it is already NP-hard to produce two monomials whose sum is consistent with the sample. Thus we do not expect to find a polynomial time algorithm that finds a consistent monotone DNF with the minimal number of monomials. The techniques in the proof of Theorem 14 show that it would suffice to have a polynomial algorithm that finds the monomial that covers the largest number of positive samples. However, this is also NP-hard by a simple reduction from maximal independent set.

The aim of this paper has not been to provide tight bounds on the number of samples and the computation time needed for various learning algorithms. Certainly there are interesting tradeoffs here, hence many open problems remain in this area. The question of incremental learning, in which individual sample points are processed one at a time and only the current hypothesis is maintained and updated, has also not been addressed here. This can be incorporated into the learning model by demanding that the space used by the hypothesis be bounded by some function of the complexity of the concept to be learned as in [H85]. How does this restrict the classes of learnable concepts?

Finally, using results from [GGM84] on polyrandom collections, it can be shown that there are sequences of concept classes $\{C_n\}_{n\geq 1}$ with dimension increasing polynomially in n that are not polynomially learnable, given the existence of 1-1 one-way functions. Are there more natural examples of such sequences? n-clause, n-variable DNF could be one such example. Note that this latter example is polynomially learnable if P=NP (by Theorem 2).

Appendix

For Theorem 2 to apply, we require that the concept class C have some additional properties related to measurability, beyond the assumption that all sets in C are Borel.

Definition. C is *well-behaved* if the sets Q_ε^m and J_ε^{2m} defined before Lemma 4 above are measurable for all $\varepsilon > 0$, $m \geq 1$ and distributions P on X.

An example of a concept class C that is not well-behaved is the following. Let X be the closed interval [0,1] and let X be well-ordered such that all prefixes of the well-ordering are countable.[5] Let C consist of all suffixes of the well-ordering. It is readily verified that the dimension of C is 1, yet Theorem 8 fails even for

[5] This requires the Continuum Hypothesis.

the uniform distribution on X. In fact, in this case no finite set of points in X form an ε-net for C for any $\varepsilon < 1$, since there is always a suffix of the well-ordering that avoids the set, yet has measure 1. Theorem 1 of [VC71] also fails for this case. The problem is that J_ε^{2m} is not measurable, even for $m = 1$.

On the other hand, virtually any concept class that one might consider in the context of machine learning applications will be well-behaved. Proofs of good behavior for most common concept classes can be derived from the following lemma.

Definition. A subset C_0 of C is a *dense approximation* of C if for every finite $S \subseteq X$ and $c \in C$, there exists $c_0 \in C_0$ such that $c_0 \cap S = c \cap S$, i.e. if $\Pi_{C_0}(S) = \Pi_C(S)$ for all finite S.

Lemma 15. If C has a countable dense approximation then C is well-behaved.

Proof. It suffices to show that the sets Q_ε^m and J_ε^{2m} are Borel sets. We show this for Q_ε^m, the argument for J_ε^{2m} being similar. Q_ε^m can be rewritten as $\bigcup_{c \in C_{P,\varepsilon}} \{\bar{x} : \bar{x} \cap c = \emptyset\}$. Since each $c \in C$ is Borel, each term in the union is Borel. Since C has a countable dense approximation, the union can be replaced by a countable union of sets defined using concepts in the approximation without affecting the result. Hence Q_ε^m is Borel. □

Classes of rectangles, half-spaces, etc. clearly have countable dense approximations.

Acknowledgements

We would like to thank Emo Welzl for many helpful discussions and ideas on ε-nets and classes of finite dimension, and for pointing out [A83] to us. Les Valiant, Jan Mycielski, Janet Blumer, Herbert Edelsbrunner and Nick Littlestone also provided valuable suggestions at various stages of this investigation.

References:

[A83] Assouad, P., "Densite et Dimension," *Ann. Inst. Fourier, Grenoble* 33 (3) (1983) 233-282.

[BEHW85] Blumer, A., A. Ehrenfeucht, D. Haussler and M. Warmuth, "Occam's Razor," Technical Report UCSC-CRL-86-2, Computer Research Laboratory, University of California at Santa Cruz, February 1986.

[DW76] Devroye, L.P. and T.J.Wagner, "A distribution-free performance bound in error estimation," IEEE Trans. Info. Th., 22, 1976, pp. 586-7.

[D78] Dudley, R.M., "Central limit theorems for empirical measures," Ann. Prob., 6(6), 1978, pp. 899-929.

[GJ79] Garey, M. and D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H.Freeman, San Francisco, 1979.

[GGM84] Goldreich, O., S. Goldwasser, and S. Micali, "How to construct random functions," 25th FOCS, 1984, pp. 464-79.

[H85] Haussler, D., "Space Efficient Learning Algorithms," unpublished manuscript.

[HW85] Haussler, D. and E. Welzl, "Epsilon-nets and range queries," in preparation.

[J74] Johnson, D.S., "Approximation Algorithms for combinatorial problems," Journal of Computer and Systems Sciences, Vol. 9, 1974.

[K84] Karmarkar, N., "A New Polynomial-Time Algorithm for Linear Programming," Proc. 16th Symp. on Th. of Comp., 1984, pp. 302-311.

[LP85] Lee, D.T. and F.P.Preparata, "Computational geometry - a survey," IEEE Trans. Comp., 33(12), 1984, pp. 1072-101.

[L85] Littlestone, N., unpublished manuscript.

[M78] Masek, W.J., "Some NP-Complete Set Cover Problems," MIT Laboratory for Computer Science, unpublished manuscript.

[M84] Meggido, N., "Linear Programming in Linear Time When the Dimension is Fixed," JACM, 31(1), 1984, pp. 114-127.

[N69] Nigmatullin, R.G., "The Fastest Descent Method for Covering Problems (in Russian)," Proceedings of a Symposium on Questions of Precision and Efficiency of Computer Algorithms," Book 5, Kiev, 1969, pp. 116-126.

[P78] Pearl, J., "On the connection between the complexity and credibility of inferred models," Int. J. Gen. Sys., 4, 1978, pp. 255-64.

[P79] Pearl, J., "Capacity and error estimates for Boolean classifiers with limited capacity," IEEE Trans. Patt. An. Mach. Intell., 1(4), 1979, pp. 350-5.

[V84] Valiant, L.G., "A theory of the learnable," Comm. ACM, 27(11), 1984, pp. 1134-42.

[VP86] Valiant, L.G., L. Pitt, Technical Report, Aiken Computing Lab., Harvard University, to appear.

[V85] Valiant, L.G., "Learning disjunctions of conjunctions," Proc. 9th IJCAI, Los Angeles, CA, August 1985, vol. 1, pp. 560-6.

[VC71] Vapnik, V.N. and A.Ya.Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," Th. Prob. and its Appl., 16(2), 1971, pp. 264-80.

[VC74] Vapnik, V.N. and A.Ya.Chervonenkis, Theory of Pattern Recognition (in Russian), Nauka, Moscow, 1974.

[W86] Welzl, E., "Some properties of the Vapnik-Chervonenkis dimension," in preparation.

[WD81] Wenocur, R.S. and R.M.Dudley, "Some special Vapnik-Chervonenkis classes," Discrete Math., 33, 1981, pp. 313-8.

[YC74] Young, T. and T. Calvert, Classification, Estimation, and Pattern Recognition, Elsevier, New York, 1974.