Machine Learning, 5, 101-113 (1990)

A Necessary Condition for Learning from Positive Examples

HAIM SHVAYTSER* (HAIM%SARNOFF@PRINCETON.EDU)
David Sarnoff Research Center, CN-5300, Princeton, NJ 08543-5300

Editor: David Haussler

Abstract. We present a simple combinatorial criterion for determining concept classes that cannot be learned in the sense of Valiant from a polynomial number of positive-only examples. The criterion is applied to several types of Boolean formulae in conjunctive and disjunctive normal form, to the majority function, to graphs with large connected components, and to a neural network with a single threshold unit. All are shown to be nonlearnable from positive-only examples.

Keywords. Concept learning, learning from positive examples, nonlearnability, sample complexity, predictable concepts

1. Introduction

In many cases a system that learns from examples can be viewed as a black box that gets as input examples of a *target concept* and produces as output an approximation of the target concept which we call the *learned concept*. Valiant suggested in [Valiant, 1984; Valiant, 1985] a complexity based model for learning. Valiant's model can be informally described as follows: training examples are drawn randomly according to a fixed but arbitrary probability distribution. With respect to this distribution, a learning algorithm produces with high probability a learned concept which is a good approximation of the target concept. A class of concepts C is learnable in Valiant's model if there is a learning algorithm that for any probability distribution, and for any target concept in C, produces a learned concept with an arbitrarily small probability of mistake, by using a polynomial number of examples and polynomially bounded computational resources. (The polynomial growth is with respect to some *natural* parameters of the concept class C and the required accuracy.)

To get an intuitive feeling of the difficulty in learning concept classes with exponential resources consider training a system to classify objects based on 100 Boolean features. If the training requires 1% of all the examples, and a new example becomes available every 1 microsecond, the training takes more than 10^{10} million years.

Several papers applied Valiant's model and related models to classify certain classes of concepts as learnable and other classes as nonlearnable. See, for example, [Blumer et al., 1989; Natarajan, 1987; Kearns et al., 1987]. Many results were obtained using variations of Valiant's model, such as considering only the sample complexity (that is, the number

*Part of this work was done while the author was in the computer science department of Cornell University.

of required examples) (see [Blumer et al., 1989; Natarajan, 1987]), requiring that both the learned and the target concept are from the same concept class (see [Natarajan, 1987; Kearns et al., 1987]), and allowing the algorithm to access only positive examples during the training [Valiant, 1984; Natarajan, 1987; Kearns et al., 1987]. (The term *predictable* is used by several researchers for concept classes that are learnable in models that do not require learned concepts and target concepts to be from the same class.)

With no restrictions on the form of the learned concepts the learnability of a concept class C implies the learnability of all concept classes $C' \subset C$. Equivalently, nonlearnability of a concept class C' implies nonlearnability of all concept classes $C \supset C'$ [Kearns et al., 1987]. This suggests that nonlearnability can be determined by identifying simple nonlearnable sub-classes of a given concept class.

In this paper we develop a criterion that will enable us to classify many concept classes as nonlearnable from positive-only examples. The goal is to be able to determine nonlearnability of concept classes that can be easily recognized as contained in other concept classes. Our result is based on the observation that in many *interesting* concept classes that are defined in terms of variables, all the variables play the same role. For example, a concept class with the concept $x \equiv y \wedge z$ will also include the concepts $y \equiv x \wedge z$ and $z \equiv x \wedge y$. In the simplest of such cases, all concepts can be obtained by a permutation of variables of a single Boolean formula. We use the notation PERM(f) for the set of concepts obtained by permutations of the variables in a Boolean formula f.

When f is a formula of n variables, PERM(f) can have at most $n!$ elements. Therefore, by using results from [Blumer et al., 1987] it is always learnable (by possibly exponential algorithms) if both positive and negative examples are available for the training. On the other hand, we will show that when only positive examples are available, PERM(f) is nonlearnable for many simple formulae.

The key to learning in Valiant's distribution free model is the ability to sample examples (during the training) from their *natural* distribution. Unfortunately, this may not be possible when the learning is in a batch style (all training examples are given at once). As an example, consider the problem of training a system to recognize a digitized binary pattern of the printed letter **A**. Figure 1 shows several examples of digitized letters. As positive examples

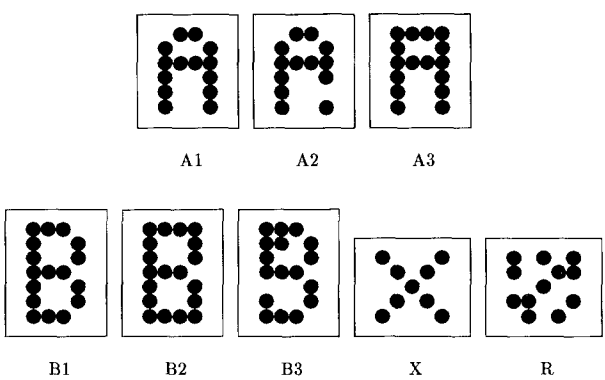

Figure 1. Digitized letters. A1, A2, A3: the letter A. B1, B2, B3: the letter B. X: the letter X. R: *random* noise.

one can take patterns such as A1, A2, and A3. Getting natural counter-examples for the training is more difficult. Using examples such as B1, B2, and B3 as counter-examples for the training may give absurd results such as identifying the pattern X or R in Figure 1 as **A**. On the other hand, trying to cover all possibilities such as the patterns X and R as counter-examples may cause many mistakes in classifying **B**s. The difficulty here is that we would like to train the system to recognize the letter **A**, without constraining the allowed counter examples. This is more difficult than merely separating **A**s from, say **B**s.

We will show that in many cases nonlearnability from positive examples of concept classes such as PERM(f) can be determined by a simple combinatorial criterion. Natarajan gave a complete characterization of learnability from positive examples in [Natarajan, 1987]. Although his results are limited to the case where both the learned and the target concepts belong to the same concept class, they can be easily generalized to arbitrary representations as we point out in Section 2. However, these results cannot be easily applied to concept classes such as PERM(f).

2. Preliminary definitions

The input to the learning algorithm is given in terms of n Boolean variables $x_1, \ldots, x_n$. A *Boolean vector* is an assignment of values from $\{0, 1\}$ to each of the n variables. A *concept* c_n on n variables is a rule that assigns the value TRUE to a subset of the 2^n vectors, and the value FALSE to the rest. Each vector in the TRUE subset is a *positive example* of c_n, and the rest are *negative examples* (counter examples) of c_n. A *concept class* C_n is a set of concepts on the variables $x_1, \ldots, x_n$.

To investigate the asymptotic complexity of concept learning we consider *families* of concept classes. A family of concept classes $\{C_n : n \geq 1\}$ is a set of concept classes such that for each $n \geq 1$, C_n is a concept class on the n Boolean variables $x_1, \ldots, x_n$.

The definition of a concept class was given in terms of subsets of examples. We are interested in cases where the concepts are represented as strings in a language, such as Boolean formulae in predicate calculus. With respect to a given language one can define the length of a concept and its complexity. The *length* of a concept in a language L is the minimal length string in the language that represents the concept. The *complexity* of a concept is the longest time that it takes to determine whether an example is a positive (or negative) example of the concept, when it is given as a minimal length string in the language. The *length* of a concept class C_n is the maximum of the lengths of the concepts $c \in C_n$. The *complexity* of a concept class C_n is the maximum of the complexity of the concepts $c \in C_n$. For a family of concept classes $\{C_n : n \geq 1\}$, the length and the complexity are functions of n. We consider only cases where the length is a polynomial function of n.

The ability to learn a family of concept classes (target concepts) may depend on the choice of *representations* that the learning algorithm can use as learned concepts. Let $\{C_n : n \geq 1\}$ be a family of target concepts, and let $\{H_n : n \geq 1\}$ be a family of representations. We use a definition of learnability that is essentially the same as the ϵ, δ definition that was given in [Kearns et al., 1987; Haussler et al., 1988] (where ϵ is an accuracy parameter and δ a confidence parameter), but consider learnability from positive-only examples. Here, even though the algorithm does not encounter negative examples during the training, we

expect it to classify with small error both positive and negative examples. It was observed in [Valiant, 1984; Natarajan, 1987; and Kearns et al., 1987] that in this case no mistakes can be allowed in classifying negative examples as being positive.

DEFINITION. A family of concept classes $\{C_n : n \geq 1\}$ is learnable by a family of concept classes $\{H_n : n \geq 1\}$ from positive-only examples if there is a learning algorithm such that for all $\epsilon, \delta > 0$ and $n \geq 1$, for all target concepts $c \in C_n$, and for all probability distributions D^+ over the positive examples of c:

(a) The algorithm gets as input N positive examples that are obtained by sampling according to the probability distribution D^+. N is bounded by a polynomial in n, $1/\epsilon$, $1/\delta$.
(b) The algorithm run time is polynomial in n, $1/\epsilon$, $1/\delta$.
(c) The output of the algorithm is a learned concept $h \in H_n$ such that:
  (i) The complexity of h is bounded by a polynomial in n, $1/\epsilon$, $1/\delta$.
  (ii) $h(v) = \text{TRUE} \Rightarrow v$ is a positive example of c.
  (iii) With probability of at least $(1 - \delta)$,

$$\sum_{\substack{c(v) = \text{TRUE} \\ h(v) = \text{FALSE}}} D^+(v) < \epsilon.$$

DEFINITION. The family $\{C_n : n \geq 1\}$ is learnable from positive-only examples if there exists a family of representations $\{H_n : n \geq 1\}$ such that $\{C_n : n \geq 1\}$ is learnable by $\{H_n : n \geq 1\}$ from positive-only examples.

We are interested in cases where the family of concepts $\{\text{PERM}(c_n) : n \geq 1\}$ is nonlearnable from positive only examples (that is, nonlearnable by arbitrary representations). The following is a formal definition for the concept class $\text{PERM}(c_n)$.

DEFINITION. Let $c_n(x_1, \ldots, x_n)$ be a concept on n variables, expressed as a string in a language L. (The language must have symbols for the variables $x_1, \ldots, x_n$.) $\text{PERM}(c_n)$ is the set of concepts $c_n(x_{i_1}, x_{i_2}, \ldots, x_{i_n})$, where $(i_1, i_2, \ldots, i_n)$ is a permutation of $(1, 2, \ldots, n)$. (Two strings in the language may represent the same concept if their truth table is the same.)

For a Boolean vector v_n of n coordinates we define $\mathbf{perm}(v_n)$ to be the set of vectors that can be obtained from v by a permutation of coordinates.

DEFINITION. Let v_n be a Boolean vector of n coordinates. $\mathbf{perm}(v_n)$ is the set of n coordinate vectors with the same number of 0 (and 1) entries as in v_n.

Throughout the paper we use the notation $|\cdot|$ for the cardinality (number of elements) of a set. The cardinality of $\mathbf{perm}(v_n)$ can be easily computed. When v_n is a Boolean vector with m 1 coordinates we have:

$$|\mathbf{perm}(v_n)| = \binom{n}{m}.$$

2.1. Natarajan's characterization

Natarajan gave in [Natarajan, 1987] a complete characterization of concept classes that are learnable from a polynomial number of positive-only examples if the complexity of the learning procedure is ignored. Although his results are limited to the case where both the target and the learned concept are from the same concept class they can be easily generalized to arbitrary representations. This generalization is given by the following theorem. Its proof, which is an immediate corollary of Natarajan's results in [Natarajan, 1987], is omitted. In the theorem the concepts are identified with the set of their positive examples.

THEOREM. Let $\{C_n : n \geq 1\}$ be a family of concept classes. $\forall n \geq 1$ let $\bar{C}_n$ be the smallest set such that:

i. $C_n \subset \bar{C}_n$.
ii. $f_n, g_n \in \bar{C}_n \Rightarrow f_n \cap g_n \in \bar{C}_n$.

The following three conditions are equivalent:

1. $\{C_n : n \geq 1\}$ is learnable from positive examples.
2. $\{\bar{C}_n : n \geq 1\}$ is learnable from positive examples.
3. There exists a polynomial $p(n)$ such that $\forall n \geq 1 \ |\bar{C}_n| \leq 2^{p(n)}$.

Natarajan's characterization is very general, but it cannot be easily applied in cases where the concepts are expressed as strings in a language. Specifically, when $C_n = \text{PERM}(f_n)$ the difficulty is that except for special cases, counting the elements of $\bar{C}_n$ is a difficult combinatorial problem.

3. The nonlearnability lemma

In this section we state and prove a useful nonlearnability lemma.

$\forall n \geq 1$ let $c_n(x_1, \ldots, x_n)$ be concepts on n variables expressed as strings in a language L. Let $\text{NEG}(c_n, n, m)$ be the number of n coordinate Boolean vectors with m "1" coordinates (and $n - m$ "0" coordinates) that are negative examples of c_n.

THE NONLEARNABILITY LEMMA: If $\forall n \geq 1$ there are numbers $0 \leq m_n \leq n$ such that $\text{NEG}(c_n, n, m_n) > 0$ and for all α,

$$\lim_{n \to \infty} \frac{n^{\alpha} \cdot \text{NEG}(c_n, n, m_n)}{\binom{n}{m_n}} = 0$$

then the family of concept classes $\{\text{PERM}(c_n) : n \geq 1\}$ in a language L is nonlearnable from positive-only examples.

Proof. We use the following terminology in the proof:

c_n is a concept on n variables given as a string in the language L.
v_n is a Boolean vector of n coordinates.
$s = s(n) = |\text{PERM}(c_n)|$.
$w = w(n) = |\mathbf{perm}(v_n)|$.
$t = t(n) = |\{v' \in \mathbf{perm}(v_n) : v' \text{ is a negative example of } c_n\}|$.
$r = r(n) = |\{c' \in \text{PERM}(c_n) : v_n \text{ is a negative example of } c'\}|$.

Example. $n = 4$; $c_n(x_1, \ldots, x_4) = (x_1x_2 \vee x_3x_4)$; $v_n = (0011)$
$\text{PERM}(c_n) = \{(x_1x_2 \vee x_3x_4), (x_1x_3 \vee x_2x_4), (x_1x_4 \vee x_2x_3)\}$
$\mathbf{perm}(v_n) = \{(0011), (0101), (1001), (0110), (1010), (1100)\}$
$s = 3$; $w = 6$; $t = 4$; $r = 2$;

Consider a graph whose vertices are $\text{PERM}(c_n)$ and $\mathbf{perm}(v_n)$, with an edge between $v' \in \mathbf{perm}(v_n)$ and $c' \in \text{PERM}(c_n)$ if and only if v' is a negative example of c'. When v' is a negative example of c', a permutation of the coordinates of v' produces a negative example for the concept which is obtained by applying the same permutation to the variables of c'. Therefore, all the $\text{PERM}(c_n)$ vertices and all the $\mathbf{perm}(v_n)$ vertices in the graph have the same degree. In our terminology, the degree of a $\text{PERM}(c_n)$ vertex is t, and that of a $\mathbf{perm}(v_n)$ vertex is r. See Figure 2 for the graph associated with the above example.

Consider a learning algorithm that attempts to learn c_n as a member of the concept class $\text{PERM}(c_n)$ with the confidence and accuracy parameters ϵ, δ by drawing $(n/\epsilon\delta)^\beta$ examples. Let the probability distribution be such that the only vectors with positive probability are the $w - t$ vectors in $\mathbf{perm}(v_n)$ that are positive examples of c_n, and each has a probability of $1/(w - t)$.

Each example from $\mathbf{perm}(v_n)$ drawn during the training enables the algorithm to eliminate all concepts connected to it in the bipartite graph from being candidates to the learned concept. A learning algorithm can classify a vector in $\mathbf{perm}(v_n)$ as being a *positive example* only if all its connected concepts in the bipartite graph were previously eliminated from being candidates. Otherwise, since the vector may be a negative example of the target concept, the error in misclassifying it may be arbitrarily high.

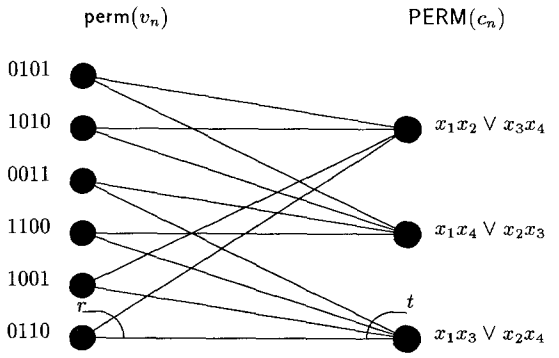

Figure 2. The $\mathbf{perm}(v_n) \leftrightarrow \text{PERM}(c_n)$ bipartite graph of the example.

By drawing $(n/\epsilon\delta)^\beta$ examples, at most r $\cdot$ $(n/\epsilon\delta)^\beta$ concepts from PERM(c_n) can be eliminated. The set of vertices corresponding to the eliminated concepts has at most $r \cdot t \cdot (n/\epsilon\delta)^\beta$ edges connected to examples. These edges can fully cover at most $r \cdot t \cdot (n/\epsilon\delta)^\beta / r = t \cdot (n/\epsilon\delta)^\beta$ vertices from **perm**(v_n).

We conclude that any learning algorithm applied to $(n/\epsilon\delta)^\beta$ positive examples from **perm**(v_n) can classify at most $t \cdot (n/\epsilon\delta)^\beta$ examples from **perm**(v_n) as being positive, and therefore, at least $w - t \cdot (n/\epsilon\delta)^\beta$ examples from **perm**(v_n) as being negative.

Since the number of negative examples in **perm**(v_n) is t, the algorithm makes mistakes in classifying at least $w - t \cdot (n/\epsilon\delta)^\beta - t$ examples, and the probability of error is at least

$$\frac{w - t \cdot \left(\frac{n}{\epsilon\delta}\right)^\beta - t}{w - t} > 1 - \left(\frac{1}{\epsilon\delta}\right)^\alpha \cdot n^\alpha \cdot \frac{t}{w}$$

(the above inequality holds for $1/\epsilon\delta \geq 1$ and $\alpha = \beta + 1$). For fixed ϵ, δ and α, when $n^\alpha \cdot t/w \to 0$, the probability of error goes to 1.

Choosing v_n to be a vector with m_n 1 coordinates we have $t = \text{NEG}(c_n, n, m_n)$, and

$$w = \binom{n}{m_n}. \blacksquare$$

We observe that the proof of the lemma is based on sample complexity arguments. The lemma can be used to find families of concept classes that cannot be predicted from a polynomial number of positive examples even with a training procedure that takes exponential time.

4. Nonlearnable classes

In this section we apply the nonlearnability lemma to prove nonlearnability from positive-only examples of several families of concept classes. The first theorem in this section deals with families of concept classes that are expressed as Boolean formulae in predicate calculus. The second theorem deals with concept classes that cannot be easily expressed as explicit Boolean formulae. The third theorem proves nonlearnability by identifying a nonlearnable sub-class.

For the proofs we use several asymptotic properties of the binomial coefficients (see, for example, [Bollobas, 1978]). The notation $q_1(n) \approx q_2(n)$ is used in this section to indicate that:

$$\lim_{n \to \infty} \frac{\log q_1(n)}{\log q_2(n)} = 1.$$

When $d(n)$ is a function of n such that $0 \leq d(n) \leq n$ and both $d(n) \to \infty$, and $n - d(n) \to \infty$, we say that $d(n)$ and $n - d(n)$ are unbounded.

a. If $d(n)$ and $n - d(n)$ are unbounded then

$$\begin{pmatrix} n \\ d(n) \end{pmatrix}$$

grows faster than any polynomial function of n.

b. $\begin{pmatrix} n \\ n/2 \end{pmatrix} \approx 2^n$.

c. $\begin{pmatrix} n \\ \log n \end{pmatrix} \approx n^{\log n}$.

d. For a constant $0 < k < 1$,

$$\begin{pmatrix} n \\ k \cdot n \end{pmatrix} \approx \left(\frac{\left(\frac{1}{k} - 1\right)^k}{1 - k} \right)^n.$$

THEOREM. The family of concept classes $\{\text{PERM}(f_n) : n \geq 1\}$ is nonlearnable from positive only examples when f_n are Boolean formulae defined in any of the following ways:

1. $$f_n(x_1, \ldots, x_n) = x_1 \vee \ldots \vee x_{d(n)} \tag{1}$$

when $d(n)$ and $n - d(n)$ are unbounded. (This result with $d(n) = n/2$ implies Theorem 15 in [Kearns et al., 1987].)

2. $$f_n(x_1, \ldots, x_n) = (x_1 \wedge x_2) \vee (x_3 \wedge x_4) \vee \ldots \vee (x_{n-1} \wedge x_n) \tag{2}$$

3. $$f_n(x_1, \ldots, x_n) = x_1 \vee \ldots \vee x_{d(n)} \vee (x_{d(n)+1} \wedge \ldots \wedge x_n) \tag{3}$$

when $d(n)$ and $n - d(n)$ are unbounded.

4. $$f_n(x_1, \ldots, x_n) = (x_1 \vee \ldots \vee x_{d(n)}) \wedge (x_{d(n)+1} \vee \ldots \vee x_n) \tag{4}$$

when $d(n)$ and $n - d(n)$ are unbounded.

5. $$f_n(x_1, \ldots, x_n) = (x_1 \wedge \ldots \wedge x_{d(n)}) \rightarrow (x_{d(n)+1} \wedge \ldots \wedge x_n) \tag{5}$$

when $d(n)$ and $n - d(n)$ are unbounded. ($\rightarrow$ is the logical implication.)

6. The majority function:

$$f_n(x_1, \ldots, x_n) = \begin{cases} \text{TRUE} & \Sigma_{i=1}^{d(n)} x_i > \dfrac{d(n)}{2} \\ \text{FALSE otherwise} & \end{cases} \tag{6}$$

for $d(n) = k \cdot n$ with $0 < k < 1$, and for $d(n) = \log n$.

Proof. We apply the nonlearnability lemma by choosing an appropriate value for m_n, computing NEG(f_n, n, m_n), and checking the condition of the lemma. The denominator in the condition of the lemma is denoted by w.

1. Take $m_n = n - d(n)$. There is only one vector with $n - d(n)$ 1 coordinates that gives FALSE in Formula (1): $x_i = 0$ for $i = 1, \ldots, d(n)$, and $x_i = 1$ for $i = d(n) + 1, \ldots, n$. Therefore, NEG(f_n, n, m_n) = 1 with

$$w = \begin{pmatrix} n \\ d(n) \end{pmatrix}.$$

Substituting in the condition of the lemma:

$$\lim_{n \to \infty} \frac{n^{\alpha} \cdot 1}{\begin{pmatrix} n \\ d(n) \end{pmatrix}} = 0.$$

And this holds when $d(n)$ and $n - d(n)$ are unbounded.

2. Take $m_n = n/2$. When Formula (2) is FALSE, each of its $n/2$ terms must have one 0; therefore, NEG(f_n, n, m_n) = $2^{n/2}$ and

$$w = \begin{pmatrix} n \\ n/2 \end{pmatrix} \approx 2^n.$$

The condition of the lemma is easily verified.

3. Take $m_n = n - (d(n) + 1)$. When Formula (3) is FALSE, $x_1 = \ldots = x_{d(n)} = 0$. Therefore, exactly one of $x_{d(n)+1}, \ldots, x_n$ is assigned the value 0, and NEG(f_n, n, m_n) = $n - d(n)$ with

$$w = \begin{pmatrix} n \\ d(n) \end{pmatrix}.$$

The condition of the lemma is easily verified.

4. We can assume without loss of generality that $d(n) \leq n - d(n)$. Take $m_n = n - d(n)$. When Formula (4) is FALSE, all the variables in the shorter conjunct must be zero; therefore, when $d(n) < n - d(n)$, NEG(f_n, n, m_n) = 1, and when $d(n) = n - d(n)$, NEG(f_n, n, m_n) = 2. Since

$$w = \begin{pmatrix} n \\ d(n) \end{pmatrix}$$

the condition of the lemma is easily verified.

5. Take $m_n = d(n)$. Since Formula (5) is false only if $x_1 = \ldots = x_{d(n)} = 1$, $\mathrm{NEG}(f_n, n, m_n) = 1$ with

$$w = \begin{pmatrix} n \\ d(n) \end{pmatrix},$$

and the condition of the lemma is easily verified.

6. Take $m_n = n - d(n)/2$. Formula (6) is FALSE only when all the 0 entries of a vector with $d(n)/2$ 0 coordinates are among the first $d(n)$ variables. Therefore,

$$\mathrm{NEG}(f_n, n, m_n) = \begin{pmatrix} d(n) \\ d(n)/2 \end{pmatrix} \approx 2^{d(n)},$$

and

$$w = \begin{pmatrix} n \\ d(n)/2 \end{pmatrix}.$$

When $d(n) = \log n$, $\mathrm{NEG}(f_n, n, m_n) \approx n$, $w \approx n^{\log n}$, and the condition of the lemma is easily verified. When $d(n) = k \cdot n$ for a constant k, $\mathrm{NEG}(f_n, n, m_n) \approx 2^{d(n)} = (2^k)^n$, and

$$w = \begin{pmatrix} n \\ \frac{k}{2} n \end{pmatrix} \approx \left(\frac{(2/k - 1)^{k/2}}{1 - \frac{k}{2}} \right)^n$$

The condition of the lemma holds for values of k for which:

$$2^k < \frac{\left(\frac{2}{k} - 1 \right)^{k/2}}{1 - \frac{k}{2}}$$

and this can be shown to hold for all values of $0 < k < 1$. ■

Let G be the set of graphs with p vertices. The edges of a graph $g \in G$ can be defined by the Boolean variables $\{g_{i,j}\}$, where $g_{i,j} = 1$ if there is an edge between vertices i, j and 0 otherwise. $g \in G$ is characterized by the $n = p(p - 1)/2$ variables $g_{i,j}$ where $0 < i \leq j \leq p$.

THEOREM. Let $c_n(g_{1,1}, g_{1,2}, \ldots, g_{p,p})$ stand for: *"the graph $g \in G$ has a connected component of size greater than p/2."* The family of concept classes

$$\{\text{PERM}(c_n) : n = p(p-1)/2, p \geq 1\}$$

is nonlearnable from positive-only examples.

Proof. The following fact, which can be easily proved, is needed for the proof: let q be the p-vertex graph with the maximum number of edges that has no connected components of size greater than $p/2$. q is the graph of two unconnected cliques, each of size $p/2$. It is unique up to isomorphism.

To apply the nonlearnability lemma take $m(p) = p/2 \cdot (p/2 - 1)$. Since this is exactly the number of edges in the graph q, the number of vectors of $p(p-1)/2$ coordinates with $m(p)$ 1 coordinates that are negative examples of c_n is the number of graphs isomorphic to q, which is

$$\frac{1}{2}\binom{p}{p/2} \approx 2^p.$$

Therefore, we have

$$\text{NEG}(c_n, p(p-1)/2, m(p)) \approx 2^p,$$

and the denominator in the condition of the lemma is:

$$w = \binom{\frac{p(p-1)}{2}}{\frac{p}{2}\left(\frac{p}{2} - 1\right)} = \binom{\frac{p^2}{2} - \frac{p}{2}}{\frac{p^2}{4}} \approx 2^{p^2/2}.$$

The condition of the lemma is easily verified. ■

Consider a neural network with a single threshold unit such as the one in Figure 3. The ability of this network to learn from both positive and negative examples was extensively analyzed in [Minsky & Papert, 1969]. The network can compute all the Boolean functions of the type:

$$N(x_1, \ldots, x_n) = \begin{cases} \text{TRUE} & \Sigma_{i=1}^{n} w_i \cdot x_i > \theta \\ \text{FALSE otherwise} & \end{cases} \tag{7}$$

for all possible choices of coefficients w_i, θ.

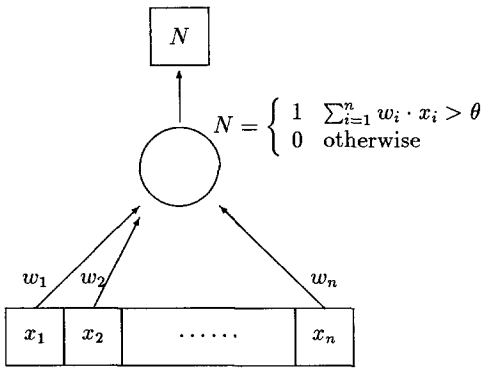

Figure 3. A network with one threshold node.

THEOREM. A family of concept classes consisting of all the Boolean functions expressible in the form of Equation (7) is nonlearnable from positive-only examples.

Proof. The proof follows from the fact that all the functions in PERM(f_n) where $f_n(x_1, \ldots, x_n)$ is the Boolean formula (1) can be realized by the network by choosing appropriate coefficients w_i, θ. For example, by choosing $w_i = 1$, for $i = 1, \ldots, d$, and $w_i = 0$ for $i = d + 1, \ldots, n$, with $\theta = 0$:

$$\sum_{i=1}^{d} x_i > 0 \quad \Leftrightarrow \quad x_1 \vee \ldots \vee x_d = \text{TRUE}.$$

(Alternatively, we could choose the majority function 6.) ■

5. Concluding remarks

From the proof of the nonlearnability lemma in Section 3 it follows that when the condition of the lemma holds, the probability of a mistake approaches 1 as $n \to \infty$. This is stronger than what is required for nonlearnability in the sense of Valiant. If we think of the distributions of positive and negative examples as the *working environment* of a learning system, the nonlearnability lemma can be understood as follows:

> If a learning system is trained with positive-only examples to recognize a concept class that contains PERM(c_n), and the condition of the nonlearnability lemma holds, then there is a target concept $c' \in$ PERM(c_n) and an environment, in which the sytem almost certainly fails to classify almost all the examples.

The nonlearnability lemma provides a sufficient, but not a necessary condition, for learnability from positive-only examples. In general, nothing follows from the fact that the condition of the lemma does not hold. However, sometimes it is possible to show that when the lemma does not hold the family of permutation classes is learnable, and thus, obtain a complete characterization. Consider, as an example, the results that were obtained for formulae (1)–(5) in the first theorem in Section 4. In each of these cases when $d(n)$ or $n - d(n)$ are bounded the permutation class has only polynomially many formulae, and is, therefore, trivially learnable from positive examples [Natarajan, 1987].

The families of concept classes that were shown nonlearnable in Section 4 were chosen to demonstrate the simplicity of proofs that use the nonlearnability lemma. Other applications of the lemma to the learnability of visual concepts can be found in [Shvaytser, in press].

Acknowledgment

I would like to thank Professor Dexter Kozen for fruitful discussions.

References

Blumer, A., Ehrenfeucht, A., Haussler, D., & Warmuth, M. (1987). Occam's razor. *Information Processing Letters, 24*: 377–380.

Blumer, A., Ehrenfeucht, A., Haussler, D., & Warmuth, M. (1989). Learnability and the Vapnik–Chervonenkis dimension. *J. of the ACM 36:*929–965.

Bollobas, A. (1978). *Extremal Graph Theory.* New York: Academic Press.

Haussler, D., Kearns, M., Littlestone, N., & Warmuth, M. (1988). Equivalence of models for polynomial learnability. *Proceedings of the first workshop on computational theory* (pp. 42–55).

Kearns, M., Li, M., Pitt, L., & Valiant, L.G. (1987). On the learnability of Boolean formulae. *Proceedings of the nineteenth annual ACM symposium on theory of computing* (May) (pp. 285–295).

Minsky, M., & Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry.* Cambridge: MIT Press.

Natarajan, B.K. (1987). On learning Boolean functions. *Proceedings of the nineteenth annual ACM symposium on theory of computing* (May) (pp. 296–304).

Shvaytser, H. (in press). Learnable and nonlearnable visual concepts. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

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

Valiant, L.G. (1985). Learning disjunctions of conjunctions. *Proceedings of the 9th IJCAI,* (Aug.) (pp. 550–556).