INFORMATION AND COMPUTATION 113, 230–252 (1994)

Composite Geometric Concepts and Polynomial Predictability

PHILIP M. LONG AND MANFRED K. WARMUTH*

University of California at Santa Cruz, Santa Cruz, California 95064

We study the predictability of geometric concepts, in particular those defined by boolean combinations of simple geometric objects. First, we give a negative results, showing that the problem of predicting the class of convex polytopes encoded by listing their vertices is prediction complete for P. Thus, an efficient solution to this prediction problem implies the existence of efficient solutions to all prediction problems whose associated evaluation problems are in P. Assuming the existence of a one-way function that is hard on iterates, there are such prediction problems which do not admit efficient solutions. Thus we show under minimal cryptographic assumptions that the class of convex polytopes encoded by listing their vertices is not predictable. As a side effect, we show that determining membership in the convex hull of a given set of points is complete for P with respect to log space reductions. Next, we establish the predictability of the class consisting of unions of a fixed number of flats by reducing its prediction problem to that of the class of flats, which has previously been shown to be predictable. Finally, we give an Occam algorithm for predicting fixed finite unions of boxes. Both constructive results for flats and boxes hold if the dimension is variable.

1. INTRODUCTION

We study the problem of predicting membership in a hidden subset c of Euclidian space (called the target concept), after being told whether each element of a finite set of points chosen independently at random according to some fixed distribution is in c. We measure the accuracy of a prediction algorithm by its probability of misclassifying a point chosen from the same distribution. We wish to find an algorithm which, given few examples, is able to achieve any desired accuracy in a small amount of time, where the amount of time, as well as the number of examples, is allowed to grow polynomially with the inverse of the desired accuracy as well as with the size of the examples and with some measure of the complexity of the hidden subset. The model of polynomial predictability used here is related to and in some sense equivalent to the PAC model introduced by Valiant

* This work was supported by ONR Grants N00014-86-K-0454 and N00014-91-J-1162. Phil Long's present address is: Computer Science Department, Duke University, P.O. Box 90129, Durham, NC 27708. E-mail address: plong@cs.duke.edu. Manfred Warmuth's E-mail address: manfred@mira.ucsc.edu.

0890-5401/94 $6.00

[Val84].[1] The prediction algorithm assumes that the target concept is chosen by an adversary from some class of subsets of Euclidian space (called the target concept class), and we ask the question of whether this assumption is strong enough to admit acceptable performance. If so, we say that this concept class is predictable. In the model treated in this paper, we assume not only that the target concept is chosen from a particular class, but that the concepts of this class are encoded using a particular representation language, and we allow the time and the number of examples required by our prediction algorithms to grow polynomially in the length of the target representation, which we take to be a measure of the complexity of the hidden concept. A more formal definition of the model (which was introduced in [HLW88, PW90]) will be given in the following section.

Since any set of points on a sphere can be shattered by the class of convex polytopes, this class has infinite Vapnik–Chervonenkis (VC) dimension [VC71],[2] and therefore, if we do not allow the algorithm's resources to grow with the complexity of the target concept, this class is not predictable [BEHW89]. To address the question of the predictability of this class when resources are allowed to grow with the length of the representation of the hidden concept, we must choose a representation language for the class of convex polytopes.

As in [PW90] we are only interested in representation classes and their associated prediction problems for which the following question can be answered in polynomial time: Given a point and a representation, is the point in the concept defined by the representation? Call the set of all such prediction problems $\mathscr{B}_P$.

Since the resources of the prediction algorithm are allowed to grow polynomially in the length of the representation of the target, positive results for representation languages which encode their concept classes concisely are stronger than those for less concise representations. For hardness results, the opposite relationship holds. Two natural representation languages (both in $\mathscr{B}_P$) for convex polytopes are to list the coefficients of the bounding hyperplanes and to list the vertices of the convex polytope. Since hypercubes have exponentially more vertices than facets and their duals have exponentially more facets than vertices, neither encoding scheme "dominates" the other in terms of conciseness.

Using the tool of prediction preserving reductions we show in Section 3 that the class of polytopes represented by listing their vertices is prediction

[1] In the original model proposed by Valiant, the algorithm is required to output a hypothesis from the target class. The notion of polynomial predictability is equivalent to that of PAC learnability in terms of any "reasonable" hypothesis class [HKLW91].

[2] Called *capacity* in [Vap82].

complete for $\mathscr{B}_P$. Thus if vertex-represented convex polytopes are predictable, then so is every prediction problem in $\mathscr{B}_P$. However, any one-way function that is hard on its iterates [GKL88, Lev87] leads to a problem in $\mathscr{B}_P$ that is not predictable [PW90]. Thus modulo the minimal cryptographic assumption that such functions exist, any prediction complete problem for $\mathscr{B}_P$ (including vertex-represented convex polytopes) is not predictable.

Baum [Bau89] gives an algorithm for predicting the class of unions of halfspaces which requires resources polynomial in the number of halfspaces and the inverse of the accuracy, but exponential in the domain dimension. The problem of whether the dependence on the domain dimension can be made polynomial as well remains open.[3]

Consider the following more complex prediction problem associated with a hidden convex polytope (contained in the unit cube) represented by a conjunction of halfspaces. The examples do not consist of points labeled according to whether they are in the hidden polytope. Instead they consist of encodings of hypercubes labeled according to whether they intersect the hidden polytope. In [PW90] this prediction problem was proven to be prediction complete for $\mathscr{B}_P$. We give a dual transformation from a problem related to the above to the prediction problem for vertex-represented convex polytopes (see the beginning of Section 3 for a high-level discussion of this reduction).

Our second result, proved in Section 4, establishes the predictability of the class of unions of a fixed number of flats. Flats are translations of subspaces of Euclidean space. Our proof of the predictability of fixed finite unions of flats consists of reducing this prediction problem to that of predicting flats. The class of flats was shown to be predictable in [Shv88]. In [VW89] and independently in [Blu89], a similar technique was applied to show that the class of "border augmented symmetric differences of halfspaces"[4] is predictable. Our result for predicting a fixed number of flats holds even if the dimension varies with the target concept.

Finally, in Section 4, we give an Occam algorithm [BEHW87] for predicting unions of a fixed number of "boxes" (Cartesian products of intervals). Again the dimension is allowed to vary. When given a sequence of examples in n dimensional space labeled consistently with some k boxes, it produces a union of up to $k(2n)^k$ boxes consistent with the examples.

[3] Baum [Bau90] also gave an elegant learning algorithm for a union of two homogeneous halfspaces that requires resources which grow only polynomially in domain dimension. Unfortunately, his method does not appear to generalize to unions of nonhomogeneous halfspaces or to unions of more than two homogeneous halfspaces. It also assumes that the distribution is symmetric about the origin.

[4] The border augmented symmetric difference of a set of halfspaces is the union of their symmetric difference and all of their bordering hyperplanes.

Provided that the example sequence is large enough, the hypothesis produced is an accurate predictor. The class of single boxes was shown to be predictable in [BEHW89] using single boxes as hypotheses.

Note that the class of intersections of halfspaces is a generalization of CNF (boolean formulae in conjunctive normal form). Also, the class of unions of axis-parallel rectangles and the class of unions of flats are generalizations of DNF (boolean formulae in disjunctive normal form). The question of whether DNF and CNF are predictable is one of the major open problems in Computational Learning Theory. As discussed in Section 4, our algorithm for predicting unions of a fixed number of boxes induces an algorithm for predicting k-term DNF using DNF as hypotheses. The hypotheses produced are $k(2n)^k$-term DNF, where n is the number of variables. The "standard" algorithm for k-term DNF uses k-CNF as hypotheses [PV88]. Another algorithm for learning k-term DNF in terms of DNF was discovered in parallel [BS90].

In Section 5, we summarize the paper and give a number of open problems.

2. Preliminary Definitions

We begin by formalizing the definition of predictability discussed in the introduction [HLW88, PW90]. Let Σ and Γ be finite alphabets. If σ is a string, let $|\sigma|$ denote the length of σ. If n is a natural number, let $\Sigma^{[n]}$ be the set of all strings over Σ of length at most n. A *concept* is any subset of Σ^*. A *prediction problem* is a pair (R, c), where $R \subseteq \Gamma^*$, and c is a function from R to 2^{Σ^*}. Elements of R are representations of concepts, and c maps representations to the concepts they represent, so $c(R)$ is the associated concept class.

Throughout the paper, we assume that integers are encoded in binary, so that encoding the integer n requires space $\Theta(\log |n|)$. Let $\mathbf{N}$ denote the positive integers, $\mathbf{Q}$ denote the rationals, and $\mathbf{R}$ denote the reals. We assume that rationals are encoded by representing their numerators and denominators. Therefore, the space to encode the rational p/q (with p and q relatively prime) is assumed to be $\Theta(\log |pq|)$.

If $y, a^{(1)}, ..., a^{(n)} \in \mathbf{R}^d$, we say that y is in the convex hull of $\{a^{(1)}, ..., a^{(n)}\}$ if and only if there exist nonnegative $\lambda_1, ..., \lambda_n \in \mathbf{R}$ such that $\sum_{i=1}^n \lambda_i = 1$ and

$$y = \sum_{i=1}^{n} \lambda_i a^{(i)}.$$

The convex hull of a finite set is a convex polytope [Gru67].

Our hardness result is for the prediction problem $B_{\text{CHULL}} = (R, c)$, where each representation of R consists of the encoding of a dimension d and a

finite set of points in $\mathbf{Q}^d$ and for each $r \in R$, $c(r)$ is the convex hull of the points represented by r.

If (R, c) is a prediction problem and $r \in R$, an *example* of $c(r)$ is a pair $(w, \text{label}(w, c(r)))$, where $w \in \Sigma^*$ and $\text{label}(w, c(r))$ is 1 if $w \in c(r)$ and 0 otherwise.

A *prediction algorithm* A is an algorithm that takes as inputs $s \in \mathbf{N}$ (interpreted as an upper bound on the length of the representation of the hidden concept), $n \in \mathbf{N}$ (an upper bound on the length of the elements of Σ^* that the learning algorithm will see), a desired probability $\varepsilon \in [0, 1]$ of making a mistake, a collection of elements of $\Sigma^{[n]} \times \{0, 1\}$, and an element $w \in \Sigma^{[n]}$. The output of A is either 1 or 0. We say A is a *polynomial time prediction algorithm* if there exists a polynomial t such that the run time of A is at most $t(s, n, 1/\varepsilon, l)$, where l is the total length of the input to A.

We say a prediction problem (R, c) is *polynomially predictable* if and only if there exist a polynomial time prediction algorithm A and a polynomial p such that for all input parameters s, n, and $\varepsilon > 0$, for all $r \in R$, $|r| \leqslant s$, and all probability distributions P on $\Sigma^{[n]}$, if A is given at least $p(s, n, 1/\varepsilon)$ examples of $c(r)$ generated according to P and $w \in \Sigma^{[n]}$, also chosen from P, then the probability that A's output differs from $\text{label}(w, c(r))$ is at most ε. Throughout the paper, we use predictable as a synonym for polynomially predictable. A number of equivalent models are described in [HKLW91].

Let $B_0 = (R_0, c_0)$ and $B_1 = (R_1, c_1)$ be prediction problems. We say that B_0 reduces to B_1 (denoted by $B_0 \trianglelefteq B_1$) if there exist f: $\Sigma^* \to \Sigma^*$ (called the *instance transformation*) and g: $R_0 \to R_1$ (called the *concept transformation*) and polynomials t and q such that

1. For all $r \in R_0$ and $w \in \Sigma^*$, $w \in c_0(r)$ iff $f(w) \in c_1(g(r))$, or for all $r \in R_0$ and $w \in \Sigma^*$, $w \in c_0(r)$ iff $f(w) \notin c_1(g(r))$.
2. For all $w \in \Sigma^*$, f is computable in time $t(|w|)$.
3. For all $r \in R_0$, $|g(r)| \leqslant q(|r|)$.

This notion of reducibility is more restrictive than that introduced in [PW90], but is all that is required for the reductions of this paper. The fact that $\trianglelefteq$ is transitive and preserves predictability was proven in [PW90].

For a prediction problem (R, c), its associated (*membership*) *evaluation problem* is defined as follows: Given $w \in \Sigma^*$, $r \in R$, is $w \in c(r)$? Let $\mathscr{B}_P$[5] be the set of all prediction problems whose associated membership evaluation problem can be computed in polynomial time. We say a prediction problem B is *prediction complete for* $\mathscr{B}_P$ if for every $B' \in \mathscr{B}_P$, B' reduces to B using the more general definition of reduction given in [PW90]. Many examples are given in [PW90].

[5] This set is called $\mathscr{R}_P$ in [PW90].

Our proof that B_{CHULL} is prediction complete for $\mathscr{B}_{\text{P}}$ consists of a reduction from the following prediction problem, which was shown to be prediction complete for $\mathscr{B}_{\text{P}}$ in [PW90]: $B_{\text{CIRC}} = (R, c)$, where $R = \{r : r$ encodes an acyclic boolean circuit with AND, NOT, and OR gates$\}$, and if r has n inputs, $c(r)$ is the set of boolean strings of length n which are accepted by the circuit encoded by r. Similar prediction problems consisting of circuits using only AND and NOT gates or AND and OR gates are also prediction complete for $\mathscr{B}_{\text{P}}$, since there is a trivial reduction between any two such prediction problems.

3. Convex Polytopes are Hard

In this section we prove that B_{CHULL} is prediction complete for $\mathscr{B}_{\text{P}}$. Our approach is motivated by the concept of a dual relationship between points and hyperplanes. Edelsbrunner [Ede84] describes the following mapping $\mathscr{D}$ from nonzero points to hyperplanes and nonhomogeneous hyperplanes to points:[6]

- If p is a nonzero point in $\mathbf{R}^d$, then $\mathscr{D}(p) = \{x \in \mathbf{R}^d : p \cdot x = 1\}$.
- If H is a nonhomogeneous hyperplane in $\mathbf{R}^d$ and $h \in \mathbf{R}^d$ is such that $H = \{x : h \cdot x = 1\}$, then $\mathscr{D}(H) = h$.

Note that a point p is on the same side of a hyperplane H as the origin if and only if the point $\mathscr{D}(H)$ is on the same side of the hyperplane $\mathscr{D}(p)$ as the origin and p is contained in H if and only if $\mathscr{D}(H)$ is contained in $\mathscr{D}(p)$. For any geometric problem involving only points and hyperplanes, there is an equivalent dual problem in which the roles of points and hyperplanes are reversed. Our proof that B_{CHULL} is prediction complete for P was motivated by the observation that such a relationship exists between the problem of determining whether a hyperplane intersects each of a finite set of open nonhomogeneous halfspaces and whether a point is not in the convex hull of a set of points. This can be seen be observing that if $H^{(1)}$, $H^{(2)}$, ..., $H^{(n)}$ are nonhomogeneous hyperplanes bounding a polyhedron P containing the origin and $H^{(*)}$ is a nonhomogeneous hyperplane, then the following are quivalent:

- $H^{(*)}$ intersects the interior of P.
- $H^{(*)}$ contains a point on the same side of each of $H^{(1)}$, $H^{(2)}$, ..., $H^{(n)}$ as the origin.

[6] A geometric interpretation of the mapping $\mathscr{D}$ which might give some intuition is described in [Ede84].

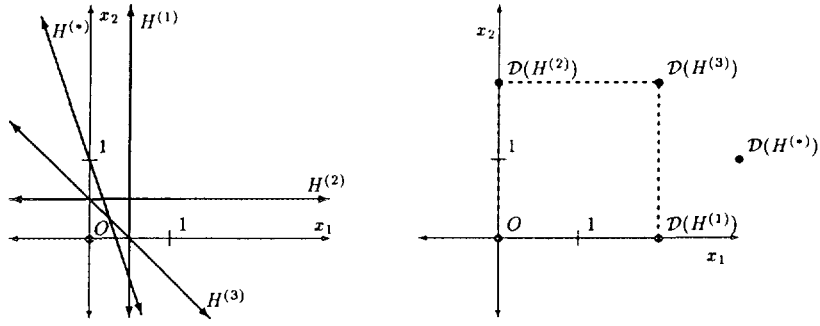


FIG. 1. An illustration of the dual mapping $\mathcal{D}$. We have that $\mathcal{D}(H^{(*)})$ is not in the convex hull of $\{\mathcal{D}(H^{(1)}), \mathcal{D}(H^{(2)}), \mathcal{D}(H^{(3)}), 0\}$, corresponding to the fact that $H^{(*)}$ intersects the interior of the polyhedron containing the origin and bounded by $H^{(1)}$, $H^{(2)}$, and $H^{(3)}$.

- There is a hyperplane G containing the point $\mathcal{D}(H^{(*)})$ such that each of $\mathcal{D}(H^{(1)}), \mathcal{D}(H^{(2)}), \ldots, \mathcal{D}(H^{(n)})$ is on the same side of G as the origin (by the properties of the dual mapping $\mathcal{D}$ described above).
- The point $\mathcal{D}(H^{(*)})$ is not in the convex hull of $\{\mathcal{D}(H^{(1)}), \mathcal{D}(H^{(2)}), \ldots, \mathcal{D}(H^{(n)}), 0\}$.

An algebraic formalization of this argument is given in Lemma 3 and an example is given in Fig. 1. Consider the following prediction problem (call it $B_{\text{HYP/POLY}}$) also described in the introduction. The instances are subcubes of the unit cube. The instances are labeled according to whether they intersect with a hidden convex polytope (contained in the unit cube) represented by a conjunction of halfspaces. Since for any subcube C there is a hyperplane H_C whose intersection with the unit cube is C, and since all targets are contained in the unit cube, we may substitute hyperplanes for subcubes in a reduction to $B_{\text{HYP/POLY}}$ and obtain hardness results for a similar class in which hyperplanes replace subcubes as instances. Since $B_{\text{HYP/POLY}}$ is known to be prediction complete for $\mathcal{B}_{\text{P}}$ [PW90], we hoped to construct a prediction preserving reduction from $B_{\text{HYP/POLY}}$ to B_{CHULL} based on duality.

The first obstacle was that the halfspaces in $B_{\text{HYP/POLY}}$ were closed halfspaces and potentially homogeneous as well, so we were forced to prove the following strange problem prediction complete for $\mathcal{B}_{\text{P}}$ which is a restriction of $B_{\text{HYP/POLY}}$: $B_{\text{PLUS}} = (R, c)$, where each representation $r \in R$ consists of the encoding of a dimension d and a finite set A of points in $\mathbf{Q}^d$, and the concept $c(r)$ represented by r is

$$\{b \in \mathbf{Q}^d : \exists x \in \mathbf{R}^d,\ b \cdot x = 1 \text{ and } \forall a \in A,\ a \cdot x < 1\}.$$

Applying the dual transformation to this class gives B_{CHULL}. Since simulating a circuit with open, nonhomogeneous halfspaces is more difficult than with

closed arbitrary halfspaces, our reduction from B_{CIRC} to B_{PLUS} is more complex than the original proof of the hardness of $B_{\text{HYP/POLY}}$.

First, we give the reduction from B_{CIRC} to B_{PLUS}.

THEOREM 1. $B_{\text{CIRC}} \trianglelefteq B_{\text{PLUS}}$.

Proof. Since any circuit using AND, OR, and NOT gates can be trivially simulated by a monotone circuit with constant blowup using double-railed logic, assume without loss of generality that all circuits in B_{CIRC} are monotone.

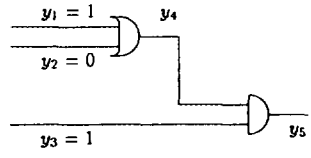
Suppose $B_{\text{CIRC}} = (R_0, c_0)$ and $B_{\text{PLUS}} = (R_1, c_1)$. We give a concept transformation g and an instance transformation f satisfying the requirements of a prediction preserving reduction. Let r be a representation of an acyclic monotone circuit C_r, with y_i, $1 \leqslant i \leqslant s$, where the input gates are y_i, $1 \leqslant i \leqslant n$, and for all i, $n+1 \leqslant i \leqslant s$, y_i is an AND or OR gate taking inputs from some y_j and y_k, such that $j < i$ and $k < i$.

Define the concept transformation g by creating linear inequalities in $\mathbf{Q}^s$ as follows. When reading these, it is useful to keep in mind that we intend x_i to be "near" 1 when y_i evaluates to 1 and "near" 0 when y_i evaluates to 0. For each AND gate $y_i = y_j \wedge y_k$, include the following inequalities. We explain the purpose of these inequalities in parenthesis following each inequality and give an example in Fig. 2:

$$x_i - x_j < 2^{-(s+2)} \qquad (\bar{x}_j \Rightarrow \bar{x}_i) \tag{1}$$

$$x_i - x_k < 2^{-(s+2)} \qquad (\bar{x}_k \Rightarrow \bar{x}_i) \tag{2}$$

$$x_j + x_k - x_i < 1 + 2^{-(s+2)} \qquad (x_j \wedge x_k \Rightarrow x_i). \tag{3}$$



		Constraints	In B_{PLUS} form	Dual points
$g(r)$	y_4	$x_1 - x_4 < \frac{1}{32}$	$32x_1 - 32x_4 < 1$	$(32, 0, 0, -32, 0)$
		$x_2 - x_4 < \frac{1}{32}$	$32x_2 - 32x_4 < 1$	$(0, 32, 0, -32, 0)$
		$x_4 - x_1 - x_2 < \frac{1}{32}$	$32x_4 - 32x_1 - 32x_2 < 1$	$(-32, -32, 0, 32, 0)$
	y_5	$x_5 - x_4 < \frac{1}{32}$	$32x_5 - 32x_4 < 1$	$(0, 0, 0, -32, 32)$
		$x_5 - x_3 < \frac{1}{32}$	$32x_5 - 32x_3 < 1$	$(0, 0, -32, 0, 32)$
		$x_3 + x_4 - x_5 < \frac{33}{32}$	$\frac{32}{33}x_3 + \frac{32}{33}x_4 - \frac{32}{33}x_5 < 1$	$(0, 0, \frac{33}{32}, \frac{33}{32}, \frac{-33}{32})$
	⋮	$\forall i, x_i > \frac{-1}{96}$	$\forall i, -96x_i < 1$	$(0, \ldots, -96, \ldots, 0)$
	⋮	$\forall i, x_i < \frac{97}{96}$	$\forall i, \frac{96}{97}x_i < 1$	$(0, \ldots, \frac{96}{97}, \ldots, 0)$
$f(w)$		$(1-x_1)+x_2+(1-x_3)+(1-x_5)=0$	$\frac{-1}{3}x_1 + \frac{1}{3}x_2 - \frac{1}{3}x_3 - \frac{1}{3}x_5 = 1$	$(\frac{-1}{3}, \frac{1}{3}, \frac{-1}{3}, 0, \frac{-1}{3})$

FIG. 2. An example of the reductions $B_{\text{CIRC}} \trianglelefteq B_{\text{PLUS}} \trianglelefteq B_{\text{CHULL}}$: y_5 is 1 corresponding to the fact that $(-1/3, 1/3, -1/3, 0, -1/3)$ is not in the convex hull of the other dual points together with the origin.

For each OR gate $y_i = y_j \vee y_k$, include the following inequalities:

$$x_j - x_i < 2^{-(s+2)} \qquad (x_j \Rightarrow x_i) \tag{4}$$

$$x_k - x_i < 2^{-(s+2)} \qquad (x_k \Rightarrow x_i) \tag{5}$$

$$x_i - x_j - x_k < 2^{-(s+2)} \qquad (\bar{x}_j \wedge \bar{x}_k \Rightarrow \bar{x}_i). \tag{6}$$

In addition, for each x_i, $1 \leqslant i \leqslant s$, add the inequalities

$$\frac{-1}{n2^{s+2}} < x_i < 1 + \frac{1}{n2^{s+2}}.$$

Note that by multiplying by a constant, each of the inequalities given above can be transformed into the form $a \cdot x < 1$. Also note that since $n \leqslant s$, each of the inequalities given above can be written using $O(s)$ bits, so g is polynomially length preserving.

Define the instance transformation f as follows. Let $w \in \{0, 1\}^n$ be an input to a circuit with n input gates. Let $Z = \{i : w_i = 0\}$, $N = (i : w_i = 1\}$. Let $f(w)$ represent b such that $\{x : b \cdot x = 1\}$ is equal to

$$\left\{x : \left[\sum_{i \in Z} x_i\right] + \left[\sum_{i \in N} (1 - x_i)\right] + (1 - x_s) = 0\right\}. \tag{7}$$

Note that the guaranteed appearance of the $(1 - x_s)$ component ensures that this hyperplane can be written in the form $b \cdot x = 1$. Also note that f is trivially polynomially computable.

Choose $w \in \Sigma^*$ and r, a representation of an acyclic monotone boolean circuit. Let H be the hyperplane represented by $f(w)$ and let L be the set of points satisfying the inequalities of $g(r)$. Let y_i, $1 \leqslant i \leqslant s$, be the values of the circuit represented by r computing w.

First, we can establish that if $w \in c_0(r)$, then $f(w) \in c_1(g(r))$ by constructing $x \in \mathbf{R}^s$ witnessing this fact: set $x_i = y_i$ for $1 \leqslant i \leqslant s$. One can methodically verify that x satisfies all the constraints defining L and that $x \in H$, which together imply that $f(w) \in c_1(g(r))$.

We prove that $f(w) \in c_1(g(r))$ implies $w \in c_0(r)$ by choosing $x \in H \cap L$ and proving by induction that for all i, $1 \leqslant i \leqslant s$, x_i is close to y_i; i.e., that x simulates the execution of the input circuit. Choose $x \in H \cap L$, so x satisfies all of (1) through (7). We claim that for all i, $|x_i - y_i| \leqslant (2^{i+1} - 1)/(2^{s+2})$. Note that by (3), we need only show that if $y_i = 0$ then $x_i \leqslant (2^{i+1} - 1)/(2^{s+2})$, and if $y_i = 1$ then $x_i \geqslant 1 - (2^{i+1} - 1)/(2^{s+2})$.

For our base case, in which $1 \leqslant i \leqslant n$, we show that $|x_i - y_i| \leqslant 1/2^{s+2}$. Note that all of these are input gates; i.e., $y_i = w_i$ for all i, $1 \leqslant i \leqslant n$. Choose i_0, $1 \leqslant i_0 \leqslant n$. Suppose that $w_{i_0} = 0$. We have

$$x_{i_0} = \left[\sum_{i \in Z - \{i_0\}} -x_i \right] + \left[\sum_{i \in N} (x_i - 1) \right] + (x_s - 1) \qquad \text{(by (7))}$$

$$\leqslant \left[\sum_{i \in Z - \{i_0\}} \frac{1}{n2^{s+2}} \right] + \left[\sum_{i \in N} \frac{1}{n2^{s+2}} \right] + \frac{1}{n2^{s+2}} \qquad \text{(by (3))}$$

$$= \frac{1}{2^{s+2}}.$$

We can handle the case $w_{i_0} = 1$ similarly, showing that in this case $x_{i_0} \geqslant 1 - 1/2^{s+2}$. This completes the proof of the base case.

Also, it is easy to prove that $x_s \geqslant 1 - 1/2^{s+2} > 1/2$ using (3) and (7) as above.

For the induction step, choose i such that $n < i \leqslant s$ and make the inductive assumption that for all $j < i$, $|x_j - y_j| \leqslant (2^{j+1} - 1)/2^{s+2}$.

Assume as a first case that y_i is an AND-gate with inputs y_j and y_k, and that $y_i = 0$. Assume w.l.o.g. that $y_j = 0$. Then by (1),

$$x_i < \frac{1}{2^{s+2}} + x_j$$

$$\leqslant \frac{1}{2^{s+2}} + \frac{2^{j+1} - 1}{2^{s+2}}$$

$$\leqslant \frac{1}{2^{s+2}} + \frac{2^i - 1}{2^{s+2}}$$

$$< \frac{2^{i+1} - 1}{2^{s+2}}.$$

Assume as a second case that y_i is an AND-gate with inputs y_j and y_k, and that $y_i = 1$. Then by (3),

$$x_i > x_j + x_k - 1 - \frac{1}{2^{s+2}}$$

$$\geqslant \left(1 - \frac{2^i - 1}{2^{s+2}}\right) + \left(1 - \frac{2^i - 1}{2^{s+2}}\right) - 1 - \frac{1}{2^{s+2}}$$

$$= 1 - \frac{2^{i+1} - 1}{2^{s+2}}.$$

Assume as a third case that y_i is an OR-gate with inputs y_j and y_k, and that $y_i = 0$. Then by (6),

$$\begin{aligned} x_i &< x_j + x_k + \frac{1}{2^{s+2}} \\ &\leqslant \frac{2^i - 1}{2^{s+2}} + \frac{2^i - 1}{2^{s+2}} + \frac{1}{2^{s+2}} \\ &= \frac{2^{i+1} - 1}{2^{s+2}}. \end{aligned}$$

Assume as a fourth case that y_i is an OR-gate with inputs y_j and y_k, and that $y_i = 1$. Assume w.l.o.g. that $y_j = 1$. Then by (4),

$$\begin{aligned} x_i &> x_j - \frac{1}{2^{s+2}} \\ &\geqslant 1 - \frac{2^i - 1}{2^{s+2}} - \frac{1}{2^{s+2}} \\ &> 1 - \frac{2^{i+1} - 1}{2^{s+2}}. \end{aligned}$$

This completes the induction. So for all i, $1 \leqslant i \leqslant s$, we have

$$|x_i - y_i| \leqslant \frac{2^{i+1} - 1}{2^{s+2}} \leqslant \frac{2^{s+1} - 1}{2^{s+2}} < \frac{1}{2}.$$

Since $x_s > 1/2$, this implies that $y_s = 1$, which in turn implies that the input circuit evaluates to 1, i..e, $w \in c_0(r)$. ∎

We next reduce B_{PLUS} to B_{CHULL}, for which we need the following simple lemma. Our proof is similar to that of a related theorem in [Gru67, p. 11].

LEMMA 2. *Let C be a closed, convex subset of $\mathbf{R}^d$ and let $y \in \mathbf{R}^d$ be an element outside of C. Then there exists a hyperplane H containing y such that $H \cap C = \varnothing$.*

Proof. Since C is closed, there is a point $c \in C$ closest to y. Let

$$H = \{x : (c - y) \cdot x = (c - y) \cdot y\}.$$

Clearly, $y \in H$.

First, assume for contradiction that $c \in H$. Then

$$(c - y) \cdot c = (c - y) \cdot y,$$

which implies that

$$(c - y) \cdot c - (c - y) \cdot y = 0,$$

which in turn gives

$$(c - y) \cdot (c - y) = 0.$$

This implies that $c = y$, which in turn implies that $y \in C$, which is a contradiction. Thus, we have that $c \notin H$.

Now, choose $z \in H - \{y\}$. Note that c, y, and z are distinct. Assume for contradiction that $z \in C$. Trivially, the triangle with vertices at c, y, and z is a right triangle with the right angle at y, so if w is the element of the segment between c and z closest to y, then $w \notin \{c, z\}$, and thus w is closer to y than c. But since c and z are in the convex set C, w is in C also, which contradicts the assumption that c is the closest element to y in C.

Since z was chosen arbitrarily, $H \cap C = \varnothing$. ∎

The following technical lemma basically amounts to the observation discussed above that under certain assumptions a hyperplane intersects the interior of a polyhedron if and only if its dual point is not a member of the convex hull of the duals of the bounding hyperplanes of the polyhedron together with the origin. Similar facts are proved in [Ede84].

LEMMA 3. *Let $A = \{a^{(i)} : 1 \leqslant i \leqslant n\} \subseteq \mathbf{Q}^d$. Let $b \in \mathbf{Q}^d$. There exists $x \in \mathbf{R}^d$ such that $b \cdot x = 1$ and $a^{(i)} \cdot x < 1$ for all i, $1 \leqslant i \leqslant n$, if and only if b is not a member of the convex hull of $A \cup \{0\}$.*

Proof. First, assume for contradiction that there exists $x \in \mathbf{R}^d$ such that $b \cdot x = 1$, and $a^{(i)} \cdot x < 1$ for all i, $1 \leqslant i \leqslant n$ and b is in the convex hull of $A \cup \{0\}$. Since b is in the convex hull of $A \cup \{0\}$, there exists a sequence λ_i, $1 \leqslant i \leqslant n$, such that for all i, $\lambda_i \in \mathbf{R}$ and $\sum \lambda_i \leqslant 1$ and for all i, $1 \leqslant i \leqslant n$, $0 \leqslant \lambda_i \leqslant 1$ and $b = \sum_{i=1}^{n} \lambda_i a^{(i)}$. So we have

$$b \cdot x = \left(\sum_{i=1}^{n} \lambda_i a^{(i)} \right) \cdot x = \sum_{i=1}^{n} \lambda_i (a^{(i)} \cdot x) < \sum_{i} \lambda_i \leqslant 1.$$

This contradicts the assumption that $b \cdot x = 1$, and thereby proves the forward direction of the lemma.

Now, suppose that b is not in the convex hull of $A \cup \{0\}$. Let H be a hyperplane containing b and avoiding the convex hull of $A \cup \{0\}$. The

existence of such an H is established by Lemma 2. Let $x = \mathcal{D}(H)$; i.e., let x be such that

$$H = \{z : x \cdot z = 1\}.$$

Since $x \cdot 0 < 1$, and all points of A are in the same halfspace of H as the origin, we have $a \cdot x = x \cdot a < 1$ for all $a \in A$. ∎

THEOREM 4. $B_{\text{PLUS}} \trianglelefteq B_{\text{CHULL}}$.

Proof. Let $B_{\text{PLUS}} = (R, c)$ and $B_{\text{CHULL}} = (R', c')$. Define $g: R \to R'$ as follows. Suppose r represents the constraints $a^{(i)} \cdot x < 1$, $1 \leqslant i \leqslant s$. Then let $g(r)$ be the representation of the set $\{a^{(i)} : 1 \leqslant i \leqslant s\} \cup \{0\}$. Define f as follows. Suppose w represents the hyperplane $b \cdot x = 1$; then let $f(w)$ be a representation of b. By Lemma 3, $w \in c(r)$ if and only if $f(w) \notin c'(g(r))$. ∎

By the transitivity of $\trianglelefteq$, together with the fact that one can test whether a point is a convex combination of a finite set of points in polynomial time using linear programming [Kar84], we get the main result (see Fig. 2 for an example tracing both reductions used).

THEOREM 5. B_{CHULL} *is prediction complete for* $\mathcal{B}_{\text{P}}$.

We can easily extend the preceding argument to establish that the following problem is log space complete for P: given a finite set $S \subseteq \mathbf{Q}^d$, and $x \in \mathbf{Q}^d$, is x in the convex hull of S? First, it was established in [Gol77] that the evaluation problem for monotone circuits is log space complete for P.[7] Using the reduction of the previous section, we can now prove the following.

THEOREM 6. *The problem of determining whether a point is in the convex hull of a finite set of points is log space complete for P.*

Proof. The only question is whether the inequalities of the reduction of Theorem 1 (with right hand sides normalized to 1) can be output using log space. Each inequality is represented with a constant number of fraction chosen from

$$\left\{\pm 2^{s+2},\ -n2^{s+2}, \pm\frac{2^{s+2}}{2^{s+2}+1}, \frac{n2^{s+2}}{n2^{s+2}+1}\right\}.$$

[7] Surveys of P-complete problems can be found in [HR85, MSS89].

Recall that we assumed that rationals are represented by writing their numerator and denominator in binary.[8] Given n and s, all these numbers can trivially be output using $O(\log ns)$ space. The theorem now easily follows. ∎

4. Positive Results

In this section we give proofs of the polynomial predictability of two classes. We list below some of the prediction problems treated in this section.

- $B_{\square} = (R, c)$, where $R = \{r : \exists d \in \mathbf{N}$ such that r encodes $(l, u) \in \mathbf{Q}^d \times \mathbf{Q}^d\}$ and $c(r) = \prod_{i=1}^{d} [l_i, u_i]$.
- $B_{\mathrm{FLAT}} = (R, c)$, where R consists of encodings of coefficients of elements of $\{A \subseteq \mathbf{Q}^d$: A finite, $d \in \mathbf{N}\}$ and $c(r)$ is defined as follows: If r encodes A, then

$$c(r) = \{x \in \mathbf{Q}^d : \forall a \in A,\ a \cdot x = 0\}.$$

If (R, c) is a prediction problem, define $\mathbf{U}_k(R, c) = (R', c')$, where R' consists of encodings of all finite sequences of elements of R of length no greater than k and for each $r' \in R'$, if r' represents $(s_1, ..., s_l)$, $c'(r') = \bigcup_{i=1}^{l} s_i$. Define $\mathbf{U}$ similarly for unbounded finite unions.

As our first positive result, we show that $\mathbf{U}_k(B_{\mathrm{FLAT}})$ is predictable by reducing this prediction problem to B_{FLAT}. The fact that B_{FLAT} is predictable was proved in [Shv88, HSW91]. Our reduction is similar to that of [Blu89, VW89] which showed that the class of border augmented symmetric differences of halfspaces reduces to the class of halfspaces.

THEOREM 7. $\mathbf{U}_2(B_{\mathrm{FLAT}}) \trianglelefteq B_{\mathrm{FLAT}}$.

Proof. Let $(R_0, c_0) = \mathbf{U}_2(B_{\mathrm{FLAT}})$ and $(R_1, c_1) = B_{\mathrm{FLAT}}$.
Suppose $w \in \Sigma^*$ represents $(x_1, ..., x_d) \in \mathbf{Q}^d$. Let $f(w)$ represent

$$(x_1^2, ..., x_1 x_d, x_2 x_1, ..., x_2 x_d, ..., x_d x_1, ..., x_d^2).$$

Let $r_0 \in R_0$ encode A, $B \subseteq \mathbf{Q}^d$. Then let $g(r_0)$ represent

$$\{(a_1 b_1, ..., a_1 b_d, a_2 b_1, ..., a_2 b_d, ..., a_d b_1, ..., a_d b_d) : a \in A, b \in B\}.$$

[8] Note that our results are not sensitive to which integer basis $b \geq 2$ is used for representing integers. We can simply substitute the fractions $\{\pm b^{s+2}, -nb^{s+2}, \pm(b^{s+2})/(b^{s+2}+1), (nb^{s+2})/(nb^{s+2}+1)\}$ in the reduction of Theorem 1 and the proof goes through essentially without modification.

We have that $w \in c_0(r_0)$ if and only if

$$\left[\bigwedge_{a \in A} (a \cdot x = 0)\right] \vee \left[\bigwedge_{b \in B} (b \cdot x = 0)\right],$$

which is true if and only if

$$\bigwedge_{a \in A, b \in B} [(a \cdot x = 0) \vee (b \cdot x = 0)],$$

which in turn holds if and only if

$$\bigwedge_{a \in A, b \in B} [(a \cdot x)(b \cdot x) = 0].$$

This is true if and only if

$$\bigwedge_{a \in A, b \in B} \left[\sum_{i,j} a_i b_j x_i x_j = 0\right],$$

which, finally, holds if and only if $f(w) \in c_1(g(r_0))$.

Since f is clearly polynomially computable and g is clearly polynomially length preserving, this theorem holds. ▮

COROLLARY. 8. *For all* $k \in \mathbf{N}$, $\mathbf{U}_k(B_{\mathrm{FLAT}}) \trianglelefteq B_{\mathrm{FLAT}}$.

Sketch of Proof. By an argument similar to the above, taking f to be as above and letting g operate on pairs of halfspaces as above, we can easily verify that $\mathbf{U}_k(B_{\mathrm{FLAT}}) \trianglelefteq \mathbf{U}_{\lceil k/2 \rceil}(B_{\mathrm{FLAT}})$. The corollary then easily follows by induction. ▮

COROLLARY 9. *For all* $k \in \mathbf{N}$, $\mathbf{U}_k(B_{\mathrm{FLAT}})$ *is predictable.*

Proof. Follows from Corollary 8, together with the fact that $\trianglelefteq$ preserves predictability [PW90] and B_{FLAT} is predictable [Shv88, HSW91]. ▮

Note that there is a trivial prediction preserving reduction to $\mathbf{U}_k(B_{\mathrm{FLAT}})$ from the corresponding prediction problem in which the flats are not restricted to be homogeneous, so our result extends to unions of arbitrary flats.

For our second positive result, we give a prediction algorithm for $\mathbf{U}_k(B_\square)$ for each $k \in \mathbf{N}$. For $k = 1$, this problem has been solved in [BEHW89]. Our algorithm consists of finding a concept h of $\mathbf{U}_{k(2d)^k}(B_\square)$ consistent with the sample, and using h for prediction. We make use of the following lemma.

LEMMA 10 [BEHW89]. *Let B be any prediction problem whose associated concept class has finite VC dimension $d' \geq 1$. For all $k \geq 1$. Then the VC dimension of the concept class of $\mathbf{U}_k(B)$ is no more than $2d\,k \log(3k)$.*[9]

Our algorithm for finding a concept of $\mathbf{U}_{k(2d)^k}(B_\square)$ consistent with the sample works by calling a subroutine which takes as input a sample S and a target bound k on the number of boxes in the target, and either returns a hypothesis in $\mathbf{U}_{k(2d)^k}(B_\square)$ consistent with the sample, or correctly informs the caller that the sample was not consistent with any concept of $\mathbf{U}_k(B_\square)$.

Loosely speaking, the algorithm works as follows. For each axis, it sweeps two axis-aligned hyperplanes, "squeezing" toward each other from $-\infty$ and ∞. As a hyperplane passes sample points, the subroutine calls itself recursively to see if the examples that have been passed are consistent with some union of $k-1$ boxes, and stops when this is no longer true, saving the no more than $(k-1)(2d)^{k-1}$ hyperrectangles obtained from the last "successful" call.

After this "squeezing" has been completed for all d axes, all of the "stopped" axis-aligned hyperplanes form a box. The algorithm returns this box, together with all those saved just prior to stopping the various sweeps.

The following lemma establishes the correctness of this algorithm, which is formally described in Fig. 3.

LEMMA 11. *The algorithm given in Fig. 3 either returns a hypothesis in $\mathbf{U}_{k(2d)^k}(B_\square)$ consistent with the sample, or correctly informs the caller that the sample was not consistent with any concept of $\mathbf{U}_k(B_\square)$.*

Proof. Choose a sample S arbitrarily. Let d be the dimension of the space containing the points of S. Our proof is by induction on the second argument k.

For the base case, in which $k=1$, we claim the algorithm either returns a single box consistent with the sample, or correctly reports that there is no such box. First, trivially, if the algorithm outputs a concept, it is consistent with the sample. Now, assume for contradiction that the algorithm returns "inconsistent" when in fact $\prod_j [u_j, v_j]$ is consistent with the sample. Clearly, for all j, $z_j \leq v_j$ and $y_j \geq u_j$, which implies $\prod_j [y_j, z_j] \subseteq \prod_j [u_j, v_j]$. This implies that since $\prod_j [y_j, z_j]$ contains a negative example, $\prod_j [u_j, v_j]$ contains a negative example, which is a contradiction. So in the case that $k=1$, the algorithm behaves as described above.

[9] The lemma also holds if unions are replaced by intersections.

```
1.  boxes(S = {(x^(i), l^(i)) ∈ Q^d × {0,1} : 1 ≤ i ≤ m}, k);
2.
3.  if (k = 1)
4.    then begin
5.         for j := 1 to d do begin /* Find smallest box consistent w/ pos. examples */
6.             z_j := max{x_j^(i) : l^(i) = 1};
7.             y_j := min{x_j^(i) : l^(i) = 1};
8.             end;
9.         if ((∏_j[y_j, z_j]) ∩ {x^(i) : l^(i) = 0} ≠ ∅) /* See if hypothesis contains neg. examples */
10.          then return("inconsistent")
11.          else return(∏_j[y_j, z_j])
12.        end
13. else if (boxes(S, k − 1) ≠ "inconsistent")
14.        then return(boxes(S, k − 1))
15.        else begin
16.             for j := 1 to d do begin
17.                 sort S according to the jth entries of the x^(i)'s;
18.                 for j' := 1 step 1 /* Sweep "up" */
19.                     until boxes({(x^(i), l^(i)) : 1 ≤ i ≤ j'}, k − 1) = "inconsistent");
20.                 lower_j := boxes({(x^(i), l^(i)) : 1 ≤ i ≤ j' − 1}, k − 1);
21.                 y_j := x_j^(j') ;
22.                 for j' := m step −1 /* Sweep "down" */
23.                     until boxes({(x^(i), l^(i)) : j' ≤ i ≤ m}, k − 1) = "inconsistent");
24.                 upper_j := boxes({(x^(i), l^(i)) : j' + 1 ≤ i ≤ m}, k − 1);
25.                 z_j := x_j^(j') ;
26.                 end;
27.             if ∏_j[y_j, z_j] contains a negative example
28.               then return("inconsistent")
29.               else return((∪_j lower_j) ∪ (∪_j upper_j) ∪ {∏_j[y_j, z_j]});
30.             end
```

FIG. 3. The subroutine for the algorithm for $\mathbf{U}_k(B_\square)$.

Choose k. Make the inductive assumption that the algorithm is correct when its second argument is $k-1$.

We may assume without loss of generality that on line 13, boxes $(S, k-1)$ returns "inconsistent," since, by the inductive assumption, the lemma is trivial otherwise.

First, we claim that if the algorithm returns a concept, it contains no negative examples. By the inductive hypothesis, $(\bigcup_j \text{lower}_j) \cup (\bigcup_j \text{upper}_j)$ contains no negative examples, and the algorithm tests to ensure that $\prod_j [y_j, z_j]$ contains no negative examples prior to outputting its hypothesis.

Next, we claim that if the algorithm returns a concept, that it covers all positive examples. Choose a positive example x. If $x \in \prod_j [y_j, z_j]$, then trivially x is covered. Suppose $x \notin \prod_j [y_j, z_j]$. Choose j such that $x_j \notin [y_j, z_j]$. If $x_j < y_j$, then by the inductive hypothesis, x is covered by lower_j. Similarly, if $x_j > z_j$, x is covered by upper_j. Since x was chosen arbitrarily, if the algorithm returns a concept, every positive example is covered.

We have now established that if the algorithm returns a concept, it is consistent with the sample. We now claim that the output concept contains at most $k(2d)^k$ boxes. By the inductive hypothesis, each of the lower$_j$'s and the upper$_j$'s contains at most $(k-1)(2d)^{k-1}$ boxes, so the output hypothesis contains at most

$$2d[(k-1)(2d)^{k-1}]+1=(k-1)(2d)^k+1\leqslant k(2d)^k$$

boxes.

Finally, we show that if the algorithm returns "inconsistent," that the sample is in fact not consistent with any concept in $\mathbf{U}_k(B_\square)$. Assume for contradiction that boxes(S, k) returns "inconsistent" and the sample is consistent with

$$\bigcup_{i=1}^{k}\prod_{j=1}^{d}[u_j^{(i)}, v_j^{(i)}].$$

Since boxes(S, k) returns "inconsistent," $\prod_j [y_j, z_j]$ must contain a negative example from S. For each j, $1\leqslant j\leqslant d$, let

$$y_j^*=\max\{u_j^{(i)}: 1\leqslant i\leqslant k\}$$
$$z_j^*=\min\{v_j^{(i)}: 1\leqslant i\leqslant k\}.$$

Choose j. Let

$$(x^{(1)}, l^{(1)}), \ldots, (x^{(m)}, l^{(m)})$$

be the enumeration returned by the jth sort performed by the algorithm. Thus $x_j^{(1)}, \ldots, x_j^{(m)}$ is a nondecreasing sequence. Let

$$j'=\min\{i': \text{boxes}(\{x^{(i)}, l^{(i)}): 1\leqslant i\leqslant i'\}, k-1)=\text{"inconsistent"}\}.$$

Trivially, $y_j=x_j^{(j')}$. Assume for contradiction that $x_j^{(j')}<y_j^*$. Let i' be such that $u_j^{(i)}\leqslant u_j^{(i')}$ for all i, $1\leqslant i\leqslant k$. So $y_j^*=u_j^{(i')}$, which implies $x_j^{(j')}<u_j^{(i')}$, so none of the first j' sample points intersect box i'; that is

$$\{x^{(i)}: 1\leqslant i\leqslant j'\}\cap\left(\prod_j [u_j^{(i')}, v_j^{(i')}]\right)=\varnothing,$$

which implies that the first j' examples are consistent with the other $k-1$ boxes, which contradicts the inductive assumption, so $y_j=x_j^{(j')}\geqslant y_j^*$.

Similarly, we can show that $z_j\leqslant z_j^*$. Since the algorithm returns "inconsistent," $\prod_j [y_j, z_j]$ contains a negative example. Call it x. Choose j. Then

$$u_j^{(1)}\leqslant y_j^*\leqslant y_j\leqslant x_j\leqslant z_j\leqslant z_j^*\leqslant v_j^{(1)}.$$

Since j was chosen arbitrarily, $x \in \prod_j [u_j^{(1)}, v_j^{(1)}]$, which contradicts the assumption that the sample is consistent with

$$\bigcup_i \prod_j [u_j^{(i)}, v_j^{(i)}].$$

So if the algorithm returns "inconsistent," then the sample truly is not consistent with any k boxes. This completes the induction. ∎

The above algorithm clearly requires time polynomial in m, but exponential in k. Since the output hypothesis is in the concept class of $\mathbf{U}_{k(2d)^k}(B_\square)$ boxes, and the VC-dimension of $B_\square$ is $2d$ [BEHW89], by Lemma 10, the VC dimension of the hypothesis class of this algorithm is no more than $2^{k+2}kd^{k+1} \log 2^k 3kd^k$, which is polynomial in d for fixed k, which, by the results of [BEHW89, HKLW91], implies that only polynomially many examples are required for any desired accuracy ε of prediction. This gives the following theorem.

THEOREM 12. *For all $k \in \mathbf{N}$, $\mathbf{U}_k(B_\square)$ is predictable.*

Note that increased efficiency could be obtained by replacing the interior for loops with binary searches. Also, some redundant recursive calls could be avoided. Finally, immediately after lower_j and upper_j are assigned their values, some examples can be removed from S. That is, after line 20, we can add the statement

$$S := S - \{(x^{(i)}, l^{(i)}) : 1 \leqslant i \leqslant j' - 1\}.$$

Similarly, after line 24, we can add

$$S := S - \{(x^{(i)}, l^{(i)}) : j' + 1 \leqslant i \leqslant m\}.$$

The algorithm is presented in the given form for clarity.

Since the intersection of any box with the vertices of the unit cube consists of the vertices of some subcube, our algorithm for learning a fixed number of boxes leads to an Occam algorithm [BEHW89] for learning k-term DNF using hypotheses in DNF. The number of terms in the DNF expression returned by our algorithm is bounded by $k(2n)^k$, where n is the number of variables. In contrast, it is NP-hard to produce a consistent DNF with no more than $2k-3$ terms.

Note that the standard algorithm for learning k-term DNF [PV88] uses hypotheses in k-CNF (CNF expressions with at most k literals per clause). Along the same lines one can construct an algorithm for learning unions of k boxes by an appropriate generalization of k-CNF (We presented the algorithm of Figure 3 because of its implications for learning k-term DNF

in terms of DNF). The clauses generalize to unions of no more than k "axis-aligned" halfspaces; i.e., halfspaces of the form

$$\{x \in \mathbf{R}^d : \pm x_i \leqslant a\}$$

where $1 \leqslant i \leqslant d$ and $a \in \mathbf{R}$. As with k-term DNF, unions of up to k boxes can be expressed as intersections of at most $(2d)^k$ generalized clauses, since

$$\bigcup_{j=1}^{k} \prod_{i=1}^{d} [l_i^{(j)}, u_i^{(j)}] = \bigcup_{j=1}^{k} \bigcap_{i=1}^{d} (\{x : -x_i \leqslant -l_i^{(j)}\} \cap \{x : x_i \leqslant u_i^{(j)}\}) \quad (8)$$

$$= \bigcup_{j=1}^{k} \bigcap_{i=1}^{2d} H_i^{(j)}, \quad (9)$$

where

$$H_i^{(j)} = \begin{cases} \{x : -x_i \leqslant -l_i^{(j)}\} & \text{if } 1 \leqslant i \leqslant d \\ \{x : x_{i-d} \leqslant u_{i-d}^{(j)}\} & \text{otherwise} \end{cases}$$

and we can "distribute out" the expression in (9) to get

$$\bigcap_{\vec{i} \in \{1, \ldots, 2d\}^k} \bigcup_{j=1}^{k} H_{i_j}^{(j)}$$

which is an intersection of at most $(2d)^k$ generalized clauses. Thus we can use a standard greedy covering algorithm [BEHW89], to obtain an intersection of at most $(2d)^k \ln m + 1$ generalized clauses consistent with any sample of size m. The greedy algorithm iteratively finds a generalized clause consistent with all the positive examples and at least a fraction $(2d)^{-k}$ of the as yet "uncovered" negative examples. By the results of [BEHW89], this implies the predictability of $\mathbf{U}_k(B_\square)$ if the greedy algorithm requires polynomial time. Note that there are infinitely many generalized clauses but that if $S \subseteq \mathbf{R}^d$ is the set of m sample points, the algorithm need only consider clauses formed by the union of at most k axis-aligned halfspaces in

$$\{\{x : x_i \leqslant s_i\} : 1 \leqslant i \leqslant d,\ s \in S\} \cup \{\{x : -x_i \leqslant -s_i\} : 1 \leqslant i \leqslant d,\ s \in S\}.$$

Thus for each iteration, the algorithm need only consider $(2md)^k$ clauses, which is polynomial in the relevant parameters for fixed k. Since there

are at most $(2d)^k \ln m + 1$ iterations, this algorithm required polynomial time.

5. Conclusion

We have shown that the problem of predicting membership in convex polytopes where the polytopes are encoded by listing their vertices is prediction complete for P, and therefore almost certainly intractable. The question of whether the same concept class encoded by listing the facets is predictable remains open. The associated membership evaluation problem for the latter problem is in NC^1 [PW90] which suggests that this problem might be easier, and that it is unlikely to be prediction complete for P. However, even the problem of whether intersections of two halfspaces can be predicted for arbitrary distributions remains open (Note that the boolean restriction of this problem to 2-clause CNF is predictable [PV88]). The fact that the class of border augmented symmetric differences of halfspaces reduces to halfspaces might lead one to believe that the class of intersections of two halfspaces reduces to halfspaces. In [War89], it was conjectured that no such reduction exists, and if the instance transformation is restricted to be continuous, there is provably no such reduction [HP89]. Still, the question of whether there is a reduction with a discontinuous instance transformation remains open.

On the positive side, we showed that unions of a fixed number of subspaces, and therefore of a fixed number of flats, are predictable. It is an open problem whether the class of unbounded unions of flats is predictable. Another interesting open question is whether the class of unions of a fixed number of integer lattices [HSW91] is predictable. In addition, we showed that unions of a fixed number of boxes are predictable. The problem of predicting the class of unbounded unions of boxes remains open.

By applying Lemma 10 together with the fact that the VC-dimension of halfspaces in d dimensions is $d+1$, we can see that the VC-dimension of intersections of k halfspaces in $\mathbf{R}^d$ grows polynomially in both k and d. It is an open problem whether the class of convex polytopes in $\mathbf{R}^d$ with k vertices (i.e., convex hulls of k points in $\mathbf{R}^d$) has VC dimension which grows polynomially in k and d.

6. Acknowledgments

We thank Naoki Abe, David Cohn, Andrzej Ehrenfeucht, Yoav Freund, David Haussler, David Helmbold, Michael Kearns, Nick Littlestone, Shlomo Moran, Giulia Pagallo, and Leslie Valiant for valuable conversations. We also thank an anonymous referee for comments.

Received September 4, 1990; Final Manuscript Received March 31, 1992.

REFERENCES

[Bau89] BAUM, E. B. (1990), On learning a union of halfspaces, *J. Complexity* **6**, 67–101.

[Bau90] BAUM, E. B. (1990), Polynomial time algorithms for learning neural nets, *in* "Proceedings of the Third Workshop on Computational Learning Theory."

[Blu89] BLUM, A. (1989), "On the Computational Complexity of Training Simple Neural Networks," Technical Report MIT/LCS/TR-445 (Master's Thesis), MIT.

[BEHW87] BLUMER, A., EHRENFEUCHT, A., HAUSSLER, D., AND WARMUTH, M. K. (1987), Occam's razor, *Inform. Process. Lett.* **24**, 377–380.

[BEHW89] BLUMER, A., EHRENFEUCHT, A., HAUSSLER, D., AND WARMUTH, M. K. (1989), Learnability and the Vapnik–Chervonenkis dimension, *J. Assoc. Comput. Mach.* **36** (4).

[BS90] BLUM, A., AND SINGH, M. (1990), Learning functions of k terms, *in* "Proceedings of the Third Workshop on Computational Learning Theory."

[Ede84] EDELSBRUNNER, E. (1984), "Algorithms in Combinatorial Geometry," Springer-Verlag, New York.

[GKL88] GOLDREICH, O., KRWACZYK, H., AND LUBY, M. (1988), On the existence of pseudorandom generators, *in* "Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science," pp. 12–24.

[Gol77] GOLDSCHLAGER, L. M. (1977), The monotone and planar circuit value problems are log space complete for P, *SIGACT News* **9**, 25–29.

[Gru67] GRÜNBAUM, B. (1967), "Convex Polytopes," Interscience, New York.

[HKLW91] HAUSSLER, D., KEARNS, M., LITTLESTONE, N., AND WARMUTH, M. K. (1991), Equivalence of models for polynomial learnability, *Inform. and Comput.* **95**, 129–161. An extended abstract appeared, *in* "Proceedings of the 1st Workshop on Computational Learning Theory," Morgan Kaufmann, San Mateo, CA.

[HLW88] HAUSSLER, D., LITTLESTONE, N., AND WARMUTH, M. K. (1988), Predicting $\{0, 1\}$ functions on randomly drawn points, *in* "Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science," pp. 100–109.

[HR85] HOOVER, H. J., AND RUZZO, W. L. (1985), "A Compendium of Problems Complete for P," Technical Report, University of Washington.

[HSW91] HELMBOLD, D., SLOAN, R., AND WARMUTH, M. K. (1991), Learning integer lattices, *SIAM J. Comput.*, to appear.

[HP89] HELMBOLD, D., AND PAGALLO, G. (1989), "There is No Continuous Prediction Preserving Reduction Between the Intersection of Two Halfspaces and a Single Halfspace."

[Kar84] KARMARKAR, N. (1984), A new polynomial-time algorithm for linear programming, *Combinatorica* **4**, 373–395.

[Lev87] LEVIN, L. (1987), One-way functions and pseudorandom generators, *Combinatorica* **7**(4), 357–363.

[MSS89] MIYANO, S., SHIRAISHI, S., AND SHOUDAI, T. (1989), "A List of P-Complete Problems," Technical Report RIFIS-TR-CS-17, Kyushu University, Japan.

[Par87] PARBERRY, I. (1987), "Parallel Complexity Theory," Pitman, London.

[PV88] PITT, L., AND VALIANT, L. G. (1988), Computational limitations on learning from examples, *J. Assoc. Comput. Mach.* **35**(4), 965–984.

[PW90] PITT, L., AND WARMUTH, M. K. (1990), Prediction preserving reducibility, *J. Comput. System Sci.* **41**, 430–467.

[Shv88] SHVAYTSER, H. (1988), Linear manifolds are learnable from positive examples. Manuscript.

[Val84] VALIANT, L. G. (1984), A theory of the learnable, *Comm. ACM* **27**(11), 1134–1142.

[Vap82] VAPNIK, V. N. (1982), "Estimation of Dependences Based on Empirical Data," Springer-Verlag, New York.

[VC71] VAPNIK, V. N., AND CHERVONENKIS A. Y. (1971), On the uniform convergence of relative frequencies of events to their probabilities, *Theoret. Probab. Appl.* **16**(2), 264–280.

[VW89] VALIANT, L. G., AND WARMUTH, M. K. (1989), The border-augmented symmetric difference of halfspaces is learnable, manuscript.

[War89] WARMUTH, M. K. (1989), Towards representation independence in PAC learning, *in* "Analogical and Inductive Inference: International Workshop AII 1989," Springer-Verlag, Berlin/New York.