Report Number: WUCS-94-2

1994-01-01

Learning From a Consistently Ignorant Teacher

Michael Frazier, Sally Goldman, Nina Mishra, and Leonard Pitt

One view of computational learning theory is that of a learner acquiring the knowledge of a teacher. We introduce a formal model of learning capturing the idea that teachers may have gaps in their knowledge. The goal of the learner is still to acquire the knowledge of the teacher, but now the learner must also identify the gaps. This is the notion of learning from a consistently ignorant teacher. We consider the impact of knowledge gaps on learning, for example, monoton DNF and d-dimensional boxes, and show that leraning is still possible. Negatively, we show that knowledge gaps make... **Read complete abstract on page 2.**

Recommended Citation

Frazier, Michael; Goldman, Sally; Mishra, Nina; and Pitt, Leonard, "Learning From a Consistently Ignorant Teacher" Report Number: WUCS-94-2 (1994). *All Computer Science and Engineering Research.* https://openscholarship.wustl.edu/cse_research/341

Learning From a Consistently Ignorant Teacher

Michael Frazier, Sally Goldman, Nina Mishra, and Leonard Pitt

Complete Abstract:

One view of computational learning theory is that of a learner acquiring the knowledge of a teacher. We introduce a formal model of learning capturing the idea that teachers may have gaps in their knowledge. The goal of the learner is still to acquire the knowledge of the teacher, but now the learner must also identify the gaps. This is the notion of learning from a consistently ignorant teacher. We consider the impact of knowledge gaps on learning, for example, monoton DNF and d-dimensional boxes, and show that leraning is still possible. Negatively, we show that knowledge gaps make learning conjunctions of Horn clauses as hard as learning DNF. We also present general results describing when known learning algorithms can be used to obtain learning algorithms using a consistently ignorant teacher.

Learning From a Consistently Ignorant Teacher

Michael Frazier, Sally Goldman, Nina Mishra and
Leonard Pitt

WUCS-94-02

May 1994

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
St. Louis MO 63130-4899

Supported in part by NSF Grant IRI-9014840, NASA grant NAG 1-613, NSF Grant CCR-9110108, NSF NYI Grant CCR-9357707 and NSF Grant IRI-9014840.

Learning From a Consistently Ignorant Teacher

Michael Frazier[*]
Dept. of Computer Science
University of Illinois
Urbana, IL 61801
mfrazier@uiuc.edu

Sally Goldman[†]
Dept. of Computer Science
Washington University
St. Louis, MO 63130
sg@cs.wustl.edu

Nina Mishra
Dept. of Computer Science
University of Illinois
Urbana, IL 61801
nmishra@uiuc.edu

Leonard Pitt[‡]
Dept. of Computer Science
University of Illinois
Urbana, IL 61801
pitt@cs.uiuc.edu

WUCS-94-02

May 12, 1994

Abstract

One view of computational learning theory is that of a learner acquiring the knowledge of a teacher. We introduce a formal model of learning capturing the idea that teachers may have gaps in their knowledge. The goal of the learner is still to acquire the knowledge of the teacher, but now the learner must also identify the gaps. This is the notion of learning from a consistently ignorant teacher. We consider the impact of knowledge gaps on learning, for example, monotone DNF and d-dimensional boxes, and show that learning is still possible. Negatively, we show that knowledge gaps make learning conjunctions of Horn clauses as hard as learning DNF. We also present general results describing when known learning algorithms can be used to obtain learning algorithms using a consistently ignorant teacher.

> *"Consistency requires you to be as ignorant today as you were a year ago."*
> – Bernard Berenson (1865-1959)

[*] Supported in part by NSF Grant IRI-9014840, and by NASA grant NAG 1-613.

[†] Supported in part by NSF Grant CCR-9110108 and an NSF NYI Grant CCR-9357707.

[‡] Supported in part by NSF Grant IRI-9014840.

1 Introduction

Most of the theoretical work in concept learning models the interaction between the learner and the environment by an omniscient oracle (or teacher) that classifies all objects as positive or negative instances of the concept to be learned. Thus, it is assumed that there is a well-defined border separating positive instances from negative ones. In practice, though, classification is often unclear. For example, an algorithm designed to read handwritten cheques will likely encounter many handwritten characters that look somewhat like a "4", and somewhat like a "9". In such cases, where even an expert does not have the knowledge to classify all objects, determining which objects are unclassifiable seems at least as important as determining the classifications of objects which *are* classifiable. From the learner's perspective, the regions of the instance space that defy classification create a blurry border between the positive and negative examples that the learner must determine.

The main contributions of this paper involve (1) development of a new learning model for learning such "blurry" concepts, and alternate characterizations of this model; (2) general techniques for obtaining positive results in the new model and applications to specific problems; (3) a specific negative result for the model.

The Model of a Consistently Ignorant Teacher: We introduce a formal learning model in which a learner interacts with a teacher who has incomplete information about the target function due to intrinsic uncertainty or due to gaps in the teacher's knowledge. The key requirement we place on the teacher is that all examples labeled with "?" (indicating unknown classification) are consistent with the teacher's background knowledge about the class to which the unknown function belongs. In particular, the classification of any instance labeled with "?" should not be determinable from the positive and negative instances, and knowledge of the concept class. (Thus the teacher is "consistently ignorant".) The goal of the learner will be to learn a good approximation to the knowledge of the teacher. Namely, the learner must construct a *ternary* function (i.e. with values {0,1,?}) that, with high probability, classifies most randomly drawn instances exactly as the teacher does.

Let $\mathcal{C}$ be a concept class with each concept $c \in \mathcal{C}$ defined over example space $\mathcal{X}$. A *blurry* ternary concept $f_?$ is created by taking any f from the *base class* $\mathcal{C}$ and changing a set of instances $Q \subseteq \mathcal{X}$ from their current value to "?" indicating that the teacher does not know their classifications. Further, we require that this be consistent with the knowledge that f was chosen from $\mathcal{C}$: If every concept $f \in \mathcal{C}$ consistent with the labels of examples from $\mathcal{X} - Q$, labels q as positive (respectively, negative), then $f_?$ cannot label q as "?". More formally:

Definition 1 *Let $f_? : \mathcal{X} \to \{0, 1, ?\}$, and let*

$$P = \{x \mid f_?(x) = 1\}, N = \{x \mid f_?(x) = 0\},$$

$$\text{and } Q = \{x \mid f_?(x) = ?\}.$$

Then $f_?$ is a blurry concept *for $\mathcal{C}$ if for every $q \in Q$, there exist functions f_0 and f_1 in $\mathcal{C}$ such that:*

1. *for all $x \in P, f_0(x) = f_1(x) = 1$,*
2. *for all $x \in N, f_0(x) = f_1(x) = 0$, and*
3. *$f_0(q) = 0 \neq 1 = f_1(q)$.*

We define the blurry concept class

$$\mathcal{C}_? = \{f_? \mid f_? \text{ is a blurry concept for } \mathcal{C}\}$$

Thus for any concept class $\mathcal{C}$, the class $\mathcal{C}_?$ contains exactly those blurry concepts that can be generated from some $f \in \mathcal{C}$. We assume that random examples are chosen (by nature) from an unknown, arbitrary, distribution D, and are then given a label from $\{0, 1, ?\}$ by the teacher, and presented to the learner. We say that the learner has successfully learned $f_? \in \mathcal{C}_?$ if with probability at least $1 - \delta$, the (ternary) hypothesis output by the learner has probability at most ϵ of disagreeing with $f_?$ on a randomly drawn example from D each labeled $+, -$, or ?. If such a polynomial-time algorithm exists for learning any $f_?$ in $\mathcal{C}_?$, we say that the blurry class $\mathcal{C}_?$ is PAC or PAC-MEMB[1]

[1] We assume familiarity with the basic definitions of PAC learning, exact learning with equivalence queries, and each of these models enhanced with membership queries. By PAC-MEMB, we mean the PAC learning model with membership queries. While our results hold in this modified PAC model (examples now labeled "+","–", and "?"), with the exception of the material on learning unions of boxes, our results extend easily to a suitably modified exact model. When we say "learnable" we assume membership queries are allowed; it is easily shown that the problems we attack are hard without membership queries (where "hard" means at least as hard as standard open problems (e.g., DNF) in learning theory).

learnable, or equivalently, that the class $\mathcal{C}$ is learnable from a consistently ignorant teacher. Finally, note that one way a hypothesis h might err is if $f_?(x) = ?$ and $h(x) \neq ?$. Thus, "?" does *not* mean "don't care".

An Alternate Formulation of Our Model: To understand some complexity issues involved in learning from consistently ignorant teachers, we consider when $\mathcal{C}$ is the class of pure conjunctive concepts (monomials)—each concept is a simple conjunction of variables or their negations. Let P, Q, and N be the set of examples labeled "+", "?", and "−", respectively, for some blurry monomial. In this case, it is straightforward to show that P must be representable as a (nonblurry) monomial m. Further, it is not difficult to show that $P \cup Q$ can be represented by a unate[2] DNF that contains only those literals appearing in m (provided P is not empty). These observations are sufficient to construct a PAC-MEMB algorithm to learn the class of blurry monomials (for which P is nonempty): run a known algorithm for learning (nonblurry) monomials [41] to learn the set P of positive examples, and at the same time run a known learning algorithm for (nonblurry) unate DNF [8] to learn the set $P \cup Q$ of nonnegative examples. Then Q and N can be easily determined from knowledge of P and $P \cup Q$. (See Corollary 5 for more details.)

Is this an efficient learning algorithm? It depends on our choice of complexity parameters. As we observed, the learning problem is not that of determining some underlying boolean concept, but that of determining the ternary blurry concept, which requires learning both P and $P \cup Q$. A particularly nasty choice of "?" instances can result in a unate DNF describing the set $P \cup Q$ that has a number of terms exponential in n (hence, exponential in the size of any monomial from $\mathcal{C}$). So an appropriate measure, in the case of blurry monomials, might be the number of boolean monomials needed to describe $P \cup Q$ disjunctively. In fact, for any blurry concept, we can represent the set $P \cup Q$, as well as the set P alone, by reformulating the notion of a blurry concept as that of an *agreement* of base concepts. Below, we define the complexity of a blurry concept in terms of the complexity of the boolean concepts forming the corresponding agreement.

Definition 2 *Let F be a finite set of boolean functions. The function $Agree_F$ is a*

[2] A *unate* formula is one in which no variable appears both negated and unnegated.

ternary function whose classification on instance $x \in \mathcal{X}$ is given by

$$Agree_F(x) = \begin{cases} 1 & \text{if } f(x) = 1 \text{ for each } f \in F, \\ 0 & \text{if } f(x) = 0 \text{ for each } f \in F, \\ ? & \text{otherwise.} \end{cases}$$

The following lemma states that the problem of learning agreements of concepts from $\mathcal{C}$ is equivalent to learning $\mathcal{C}$ from a consistently ignorant teacher, or equivalently, learning the blurry class $\mathcal{C}_?$. The notion of an agreement of base concepts has independent interest, as it models a type of unanimous vote of independent agents.

Lemma 1 *For a class $\mathcal{C}$ of boolean concepts, the blurry class $\mathcal{C}_? = \{Agree_F \mid F \subseteq \mathcal{C}\}$.*

Proof Sketch: It can be shown that if $f_? \in \mathcal{C}_?$ then $f_? = \text{Agree}_F$ where for each x for which $f(x) = ?$, F contains the pair of functions f_0 and f_1 as described in Definition 1. Containment in the other direction can also be shown. □

Hence, the problem of learning blurry concepts $\mathcal{C}_?$ generated from a base class $\mathcal{C}$ is equivalent to the problem of learning the agreements of sets of concepts from the base class $\mathcal{C}$, (and equivalently, learning $\mathcal{C}$ from a consistently ignorant teacher). Using this correspondence, we obtain a complexity measure for the size of $f_?$. First, define the representation size of a finite subset of concepts F to be $\sum_{f \in F} |f|$. Now define the size of $f_? \in \mathcal{C}_?$ (denoted by $|f_?|$) to be the minimum, over all $F \subseteq \mathcal{C}$ for which $\text{Agree}_F = f_?$, of the representation size of F.

General techniques: We show how known efficient PAC-learning algorithms for a concept class $\mathcal{C}$ can be used to build an efficient algorithm for learning the agreement of nested concepts from $\mathcal{C}$. For the problem of learning the agreement of concepts from $\mathcal{C}$ that are not necessarily nested, we show that if the intersection and union of arbitrarily many concepts from $\mathcal{C}$ is learnable, then $\mathcal{C}$ is learnable from a consistently ignorant teacher (that is, the blurry class $\mathcal{C}_?$ is learnable).

Positive results: We apply the above techniques to show that blurry monomials (hence, agreements of polynomially many monomials) are learnable if there is at least one positive example. As seen above, this is closer in spirit to learning (nonblurry) unate DNF than that of learning (nonblurry) monomials. The techniques are also applied to show that for each class $\mathcal{C} \in \{$monotone DNF (CNF) formulas, k-term DNF

(k-clause CNF) formulas, decision trees, DFAs}, those blurry concepts representable by an agreement of at most a constant number of elements of $\mathcal{C}$, are learnable.

We also give an algorithm for learning blurry boxes in d-dimensional Euclidean space. By our characterization, learning blurry boxes is the same as learning agreements of (standard) boxes, which can be expressed as one of learning unions and intersections of standard boxes. Thus, the problem of learning the agreement of blurry boxes is a specialization of the widely studied problem of learning the s-fold union of boxes in E^d. Within the PAC model, Long and Warmuth [35] have given an algorithm to learn this class that runs in time polynomial in d for s constant, and Blumer et al. [15] have given an algorithm for this class that runs in time polynomial in s for d constant. Recently, Goldberg, Goldman and Mathias [24] have given an efficient algorithm to exactly learn the union of discretized boxes over the domain $\{1, \ldots, n\}^d$ with membership and equivalence queries when either d or s are constant. In addition, there has been work on learning unions of boxes in the discretized plane (i.e. when $d = 2$) [21, 19, 20]. (See Section 3.3 for a brief discussion of such work.)

Our algorithm to PAC-MEMB learn the *agreement* of s boxes in E^d runs in time polynomial in $1/\epsilon$, $1/\delta$, s, and 9^d. Consequently, the algorithm runs in polynomial time without demanding that one of s and d be constant: s can be arbitrary, and d can be as large as $O(\log s)$. (There is an additional assumption required to prove the result: that the set of positive examples is samplable.)

Negative results: To illustrate the limits of our approaches we show that learning the agreement of an arbitrary number of Horn sentences is as hard as learning DNF. Thus learning propositional Horn sentences, while learnable from omniscient teachers, is as hard as learning DNF from consistently ignorant teachers.

2 Related Work

Most previous research in concept learning assumes examples are labeled either positive or negative. In these situations the border between the positive and negative examples is well defined. There has been work addressing the issue of mislabeled training examples [11, 33, 40, 30] and some addressing the issue of noise in the attributes [39, 26, 34].

In these situations, the border between the positive and negative examples may appear blurry to the learner, but this is just the result of the noise process that has been applied to the properly labeled example. There has also been some work considering learning from noisy membership queries [25, 38].

Angluin and Slonim [12] introduced a model of *incomplete membership queries* in which each membership query is answered "don't know" with a given probability. Furthermore, this information is persistent—repeatedly making a query that was answered "don't know" always results in a "don't know" answer. As in their work, one of our goals is to model the situation in which the teacher responding to the learner's queries is not omniscient. Observe, that in Angluin and Slonim's model since the teacher is *randomly* fallible, there is no guarantee that all of the teacher's knowledge about the target concept is used in answering queries. For example, it is possible that their teacher knows that a french poodle is a poodle and that poodles are mammals, but responds with "don't know" when asked if a french poodle is a mammal.[3]

In the context of *monotone* DNF, our consistency requirement manifests itself as follows: The teacher should know that adding positive attributes to an already positive example yields a positive example. (Dually for negative examples.) Thus, in the standard boolean lattice defined over variable assignments, all positive instances are above all unknown instances, which, in turn, are above all negative instances. In Angluin and Slonim's algorithm for learning monotone DNF, if the teacher replies "don't know" to a membership query then the learner samples below x in the boolean lattice for some (known) positive example y, implying that x is a positive example. If none are found, the learner concludes with high probability that x is a negative example. Thus, the teacher's ignorance is not consistent with the knowledge that the target function is monotone; the learner can determine the underlying boolean function by deducing what the teacher does not (but should) know.

More recent investigations have considered learning concept classes when membership query responses are incorrect (as opposed to "don't know"): Angluin and

[3] In our view, the notion of an incomplete membership oracle seems to better model noise than it models incomplete knowledge. Indeed, they note that their algorithm for learning monotone DNF with an incomplete membership oracle can be used to learn monotone DNF with random 1 → 0 one-sided errors.

Krikis [10], and Angluin [6] consider learning with a bounded number of such erroneous responses, and Frazier and Pitt [23] consider learning when such incorrect responses occur randomly with probability at most $\frac{1}{2}$.

In other related work, Kearns and Schapire [32] generalized the PAC setting to non-binary values using Haussler's framework [28]. They define a *p-concept* in which each instance $x \in \mathcal{X}$ has some probability $p(x)$ of being classified as positive. In their model, the goal of the learner is to make optimal predictions, or more commonly, to accurately predict $p(x)$ for all $x \in \mathcal{X}$. One way to compare our model to theirs is to consider blurry concepts as p-concepts, but the learner's goal is only that of determining whether $p(x) = 0$, $p(x) = 1$, or $0 < p(x) < 1$. (If a written numeral is sometimes identified as "4" and sometimes as "9", the learner just wants to know this–it does not need to determine what percentage of the population calls the numeral each value.)

3 Positive Results for Learning Agreements

We show that efficient PAC and PAC-MEMB learning algorithms can be designed to learn from consistently ignorant teachers. We first consider the problem of learning the agreement of a pair of nested concepts. We show that if both concepts are chosen from classes for which efficient learning algorithms exist, then we can use these algorithms to obtain an efficient algorithm for learning the agreement of the functions. We then present a general result addressing how known algorithms for learning from omniscient teachers can be applied to learn from consistently ignorant teachers even when the base functions are not nested.

3.1 Learning Agreements of Nested Concepts

Recall that a concept $f \in \mathcal{C}$ is simply the subset of instances from $\mathcal{X}$ that f classifies as positive. Thus for two concepts f_1 and f_2, we write $f_1 \subseteq f_2$ if the set of positive examples of f_1 is a subset of the positive examples of f_2. Given a set of concepts $F = \{f_1, \ldots, f_k\}$ we say that these concepts are *nested* if $f_1 \subseteq f_2 \subseteq \cdots \subseteq f_k$. Observe that $\text{Agree}_{\{f_1,\ldots,f_k\}} = \text{Agree}_{\{f_1,f_k\}}$ and thus, without loss of generality, we consider learning the agreement, $\mathrm{Agree}_{\{f_s,f_g\}}$, of two nested functions f_s and f_g (s and g for "specific" and "general"). Suppose these are chosen, respectively, from known polynomial-time learnable concept classes $\mathcal{C}_S$ and $\mathcal{C}_G$. Then the learning algorithms for $\mathcal{C}_S$ and $\mathcal{C}_G$ can be used to learn the following blurry concept class :

$\mathrm{Nested}_?\langle\mathcal{C}_S,\mathcal{C}_G\rangle =$

$$\{\mathrm{Agree}_{\{f_s,f_g\}} \mid f_s \in \mathcal{C}_S, f_g \in \mathcal{C}_G, \text{and } f_s \subseteq f_g\}.$$

(See Figure 1 for the algorithm.)

Theorem 2 *If $\mathcal{C}_S$ and $\mathcal{C}_G$ are polynomially PAC-MEMB (respectively PAC) learnable concept classes, then the class* $\mathrm{Nested}_?\langle\mathcal{C}_S,\mathcal{C}_G\rangle$ *is polynomially PAC-MEMB (respectively PAC) learnable.*

Proof Sketch: If the target is $\mathrm{Agree}_{\{f_s,f_g\}}$ for f_s in $\mathcal{C}_S$ and f_g in $\mathcal{C}_G$, note that a positive (resp., negative) example of $\mathrm{Agree}_{\{f_s,f_g\}}$ is classified as positive (resp., negative) by both f_s and f_g and a "?" example is classified as negative by f_s and positive by f_g. Thus, algorithm $\mathcal{A}$ (Figure 1) learns $\mathrm{Agree}_{\{f_s,f_g\}}$ by running the learning algorithm for $\mathcal{C}_S$ treating "?" as "$-$" to obtain h_S, and running the algorithm for $\mathcal{C}_G$ treating "?" as "+" to obtain h_G, and outputs $h = \mathrm{Agree}_{\{h_S,h_G\}}$ as the final hypothesis.

Since h_S and h_G both have error at most $\epsilon/2$ with probability at least $1-\delta/2$, it is easily shown that h has error at most ϵ with probability at least $1-\delta$. Finally, since $\mathcal{A}_S$ and $\mathcal{A}_G$ run in polynomial time, it follows that $\mathcal{A}$ runs in polynomial time. Note that $\mathcal{A}$ only makes a membership query when either $\mathcal{A}_S$ or $\mathcal{A}_G$ does. □

3.2 A General Technique for Learning Agreements

We now show how an arbitrary agreement of concepts from a class $\mathcal{C}$ (and hence, an arbitrary blurry concept $f_?$ from $\mathcal{C}_?$), can be represented, without significant increase in size, as the agreement of two nested concepts, one of which is an intersection of concepts from $\mathcal{C}$, and the other a union of concepts from $\mathcal{C}$.[4] Thus when unions and intersections of concepts from $\mathcal{C}$ are learnable, the blurry class $\mathcal{C}_?$ is learnable.

[4] There is an interesting relationship between the definition of agreements and Mitchell's definition of a version space [36] that is discussed in a more complete version of this paper.

```
Learn-Agreement-Nested-Concepts($\boldsymbol{F}, \boldsymbol{\epsilon}, \boldsymbol{\delta}$)

1. Let $F := \{f_s, f_g\}$ such that $f_s \subseteq f_g$.
2. Let $\mathcal{A}_S$ be the PAC-MEMB learning algorithm for $\mathcal{C}_S$.
3. Let $\mathcal{A}_G$ be the PAC-MEMB learning algorithm for $\mathcal{C}_G$.
4. Simulate $\mathcal{A}_S$ (with parameters $\epsilon/2$ and $\delta/2$) as follows:
    (a) If $\mathcal{A}_S$ requests an example, then draw a random labeled example $(x, f(x))$ from $D$.
    (b) If $\mathcal{A}_S$ performs a membership query on $x$, then perform a membership query on $x$ to obtain $f(x)$.
    (c) If $f(x) = 1$ then give $(x, 1)$ to $\mathcal{A}_S$,
    (d) Else give $(x, 0)$ to $\mathcal{A}_S$.
5. Let $h_S$ be the hypothesis output by $\mathcal{A}_S$.
6. Simulate $\mathcal{A}_G$ (with parameters $\epsilon/2$ and $\delta/2$) as follows:
    (a) If $\mathcal{A}_G$ requests an example, then draw a random labeled example $(x, f(x))$ from $D$.
    (b) If $\mathcal{A}_G$ performs a membership query on $x$, then perform a membership query on $x$ to obtain $f(x)$.
    (c) If $f(x) = 1$ or $f(x) =$ "?" then give $(x, 1)$ to $\mathcal{A}_G$,
    (d) Else give $(x, 0)$ to $\mathcal{A}_G$.
7. Let $h_G$ be the hypothesis output by $\mathcal{A}_G$.
8. Return the hypothesis $\text{Agree}_{\{h_S, h_G\}}$.
```

Figure 1: A method for learning the agreement of nested concepts

We then apply these techniques to show that the agreement of monomials (with some restrictions) is learnable and for each class $\mathcal{C} \in$ {monotone DNF (CNF) formulas, k-term DNF (k-clause CNF) formulas, decision trees, DFAs}, those blurry concepts representable by an agreement of at most a constant number of elements of $\mathcal{C}$, are learnable.

We begin with the following definition.

Definition 3 *Let F be a finite set of boolean functions. The function $Union_F$ is a boolean function whose classification of instance x is given by*

$$Union_F(x) = \begin{cases} 1 & \textit{if } f(x) = 1 \textit{ for some } f \in F \\ 0 & \textit{otherwise} \end{cases}$$

Likewise, the function $Intersect_F$ is a boolean function whose classification of instance x is given by

$$Intersect_F(x) = \begin{cases} 1 & \textit{if } f(x) = 1 \textit{ for each } f \in F \\ 0 & \textit{otherwise} \end{cases}$$

To discuss the efficiency of algorithms to learn unions or intersections of concepts from a given class, we must provide a size measure for each concept in the class. As we defined $|\text{Agree}_F|$, we define $|\text{Union}_F|$ to be the minimum, taken over all $F' \subseteq \mathcal{C}$ for which $\text{Union}_{F'} = \text{Union}_F$, of the representation size of F'. (Similarly for $|\text{Intersect}_F|$.) Each concept Agree_F in the class $\mathcal{C}_?$ is equivalent to the agreement of two concepts – Union_F and Intersect_F, and the size of this representation is at most twice the size of the original representation.

Lemma 3 *Let F be a finite subset of concept class $\mathcal{C}$. Then $|Agree_{\{Intersect_F, Union_F\}}|$ $\leq 2 \cdot |Agree_F|$, and*
$Agree_F = Agree_{\{Intersect_F, Union_F\}}$.

Proof: The assertion about size holds by definition since a function f in F appears once in Agree_F and twice in $\text{Agree}_{\{\text{Intersect}_F, \text{Union}_F\}}$.

To see that the functions are equivalent, note that for any example x, $\text{Agree}_{\{\text{Intersect}_F, \text{Union}_F\}}$ labels x positive if and only if Intersect_F labels x positive since Intersect_F is more specific than $\text{Union}_F(x)$. So the function $\text{Agree}_{\{\text{Intersect}_F, \text{Union}_F\}}$ labels x positive iff every f in F labels x positive. But, by definition, this is when Agree_F labels x positive. An analogous argument shows that the two functions are identical when x is a negative example. Finally, since Agree_F and $\text{Agree}_{\{\text{Intersect}_F, \text{Union}_F\}}$ are equal on positive and negative examples, they are also equal on "?" examples. □

We now use this alternate characterization to obtain an efficient algorithm for learning from a consistently ignorant teacher when finite sets of unions and intersections from the given class are known to be learnable. To aid the exposition, we introduce the notation $\mathcal{C}_\cap$ and $\mathcal{C}_\cup$:

$$\mathcal{C}_\cap = \{\text{Intersect}_F : F \text{ a finite subset of } \mathcal{C}\} \text{ and}$$

$$\mathcal{C}_\cup = \{\text{Union}_F : F \text{ a finite subset of } \mathcal{C}\}.$$

Theorem 4 *Let $\mathcal{C}$ be a concept class for which $\mathcal{C}_\cap$ and $\mathcal{C}_\cup$ are PAC-MEMB (respectively PAC) learnable in polynomial time. Then $\mathcal{C}_?$ is PAC-MEMB (respectively PAC) learnable in polynomial time.*

Proof Sketch: For any collection F of concepts, note that $\text{Intersect}_F \subseteq \text{Union}_F$, and thus by Theorem 2, if $\mathcal{C}_\cap$ and $\mathcal{C}_\cup$ are polynomially PAC-MEMB learnable then so is $\{\text{Agree}_{\{\text{Intersect}_F, \text{Union}_F\}} \mid F \subseteq \mathcal{C}\}$. Combining this with Lemma 3 we get the desired result. □

Theorems 2 and 4 can be strengthened to hold in a suitably modified exact learning model (with membership queries).. The following corollary shows that blurry monomials with nonempty intersection is learnable.

Corollary 5 *Let $\mathcal{C}$ be the class of monomials, and let $\mathcal{C}_?^+ = \{c : c \in \mathcal{C}_?, \exists x\ c(x) = 1\}$. Then $\mathcal{C}_?^+$ is polynomially PAC-MEMB learnable.*

Proof: The class $\mathcal{C}_\cap$ is learnable since $\mathcal{C}$ is closed under intersection and known to be learnable [41]. If $F \subseteq \mathcal{C}$ is a subset for which there is some example x such that $\text{Intersect}_F(x) = 1$, then x satisfies every monomial in F and so it cannot be the case that some variable appears both negated and unnegated in F. Thus Union_F is a unate DNF formula, that is efficiently PAC-MEMB learnable [8]. It is also easily verified that for any finite $F \subseteq \mathcal{C}$, the size of the representation of Intersect_F as a monomial and the size of the representation of Union_F as a unate DNF formula are each $O(\sum_{f \in F} |f|)$. Thus by Theorem 4, $\mathcal{C}_?^+$ is learnable. □

In the above corollary, had we not discarded those blurry concepts of $\mathcal{C}_?$ with no positive examples, our proof would fail because Union_F is not necessarily unate. Thus, Union_F would be arbitrary DNF formulas (and the learnability of this class in the PAC-MEMB model is open). Also observe the corollary applies to the dual of monomials (i.e. 1-DNF) when we discard blurry concepts with no negative examples.

We now consider learning agreements of at most a constant number k of concepts from a class $\mathcal{C}$. Let $\mathcal{C}_?(k) = \{\text{Agree}_F \mid F \subseteq \mathcal{C}, |F| \leq k\}$. Applying the known learning results for monotone DNF [41] and DNF formulas with a constant number of terms [4, 14] and the known learning results for decision trees [16] and DFAs [4], we obtain the following corollary. (The corollary follows because the intersection and union of a constant number of concepts from each of the preceding classes can be represented by a single concept in the corresponding class that is at most polynomially larger.)

Corollary 6 *Let $\mathcal{C}$ be the class of monotone DNF (ℓ-term DNF, decision trees, DFAs) formulas. Then for each constant k, $\mathcal{C}_?(k)$ is polynomially PAC-MEMB learnable. The dual results for monotone CNF formulas and ℓ-clause CNF formulas also hold. For decision trees, the hypothesis space is conjunctions of unate DNF.*

3.3 Learning Unions of Boxes in Euclidean Space

In this section we give an algorithm to learn the agreement of a set of s axis-parallel boxes (henceforth referred to as boxes) in d-dimensional Euclidean space (E^d) when the set of boxes have a samplable intersection. It is easy to show that this class is a generalization of unate DNF formulas, and a specialization of the class of unions of boxes in E^d.

Blumer et al. [15] present an algorithm to PAC-learn an s-fold union of boxes in E^d by drawing a sufficiently large sample of size $m = poly\left(\frac{1}{\epsilon}, \log\frac{1}{\delta}, s, d\right)$, and then performing a greedy covering over the at most $\left(\frac{em}{2d}\right)^{2d}$ rectangles defined by the sample. Thus for d constant this algorithm runs in polynomial time. Long and Warmuth [35] present an algorithm to PAC-learn this same class by again drawing a sufficiently large sample and constructing a hypothesis consistent with the sample that consists of at most $s(2d)^s$ boxes. Thus both the time and sample complexity of their algorithm depend polynomially on $(2d)^s, \frac{1}{\epsilon}$, and $\log \frac{1}{\delta}$. So for s constant this yields an efficient PAC algorithm.

There has also been work on learning unions of s boxes in the discretized space $\{1, \ldots, n\}^d$. Most of this work has focused on the special case in which $d = 2$. Chen and Maass [21] gave an algorithm to learn the union of two axis-parallel rectangles in the discretized space $\{1, \ldots, n\} \times \{1, \ldots, m\}$ in time polynomial in $\log n$ and $\log m$, where one rectangle has a corner in the top left corner and the other has a corner in the bottom right corner. While learning the *union* of these two rectangles within these time bounds was difficult, learning the *agreement* of the rectangles is quite simple since the learner needs only learn the intersection of the two rectangles which is easily achieved.

Chen [19] gave an algorithm that uses $O(\log^2 n)$ equivalence queries to learn the union of two rectangles in the discretized plane (i.e. $\{1, \ldots, n\}^2$). Also, Chen and Homer [20] gave an algorithm to learn the union of s rectangles in the discretized plane using $O(s^3 \log n)$ membership and equivalence queries and $O(s^5 \log n)$ time.

More recently, Goldberg, Goldman and Mathias [24] have presented two algorithms to learn the union of s discretized boxes in $\{1, \ldots, n\}^d$. The first makes at most $sd + 1$ equivalence queries and uses $O((4s)^d + sd \log n)$ time and membership queries. Their second algorithm uses time and queries (both equivalence and membership) that are both polynomial in $\log n$ and s for d constant and polynomial in $\log n$ and d for s constant. Thus their algorithm uses polynomial computation time and queries when either s or d are constant.

The algorithm we present here PAC-MEMB learns the *agreement* of s boxes in E^d runs in time polynomial in $1/\epsilon$, $1/\delta$, s, and 9^d. Thus, the algorithm runs in polynomial time without demanding that one of s and d be constant (e.g. d can be $O(\log s)$). A key algorithm we use in learning the agreement of boxes is an algorithm to PAC-MEMB learn the union of a set of boxes that all lie in the same quadrant of E^d, and for which the intersection region contains the origin. We call each box in such a set an *origin-incident* box. Our algorithm to learn the union of s origin-incident boxes runs in time polynomial in both d and s.

To aid in learning the agreement of boxes, we also use the known algorithm for computing the intersection of boxes [15]. Namely, we first learn an approximation for the intersection region by applying the standard algorithm with all "?" examples treated as negative. Since the boxes have a non-empty intersection, we can subdivide E^d into at most 3^d sub-regions based on this common intersection. Each sub-region can be translated and relabeled so that we can apply our algorithm for learning the union of origin-incident boxes. In the worst case, some piece of each of the s boxes will lie in each of the 3^d regions of the sub-divided problem forcing us to learn $O(s3^d)$ boxes.

It is important to note that in obtaining our algorithm to learn the agreement of boxes we take advantage of our ability to efficiently compute the intersection region and then use this information to aid in more efficiently learning the union of the boxes. It is uncommon for *both* intersections and unions of concepts to be learnable, and thus, the possibility that information from one could be used to help learn the other is of particular interest.

3.3.1 Learning the Union of Origin-incident Boxes

We present an algorithm to learn the union of s origin-incident (nonblurry) boxes in E^d where all of the boxes are in the same quadrant (for simplicity we only present the algorithm, Figure 2, for the positive quadrant). We refer to the class of origin-incident boxes in the positive quadrant as BPQ.

We define the *upper* corner of a box $b \in$ BPQ to be the corner of the box diametrically opposed to the origin. Since any box in BPQ is uniquely identified by its upper corner, we denote an origin-incident box by box(p) where p is its upper corner. We define *maxCorner* to be a function that takes a set of points in the positive quadrant of E^d and returns the upper corner of the smallest box in BPQ that contains every point in the set.

Theorem 7 *Let $BPQ_{\cup}(s)$ be the union of at most s origin incident boxes in the positive quadrant. The class $BPQ_{\cup}(s)$ is PAC-MEMB learnable with time and sample*

```
                         LearnBPQ($S$)

/* $S$ is a labeled sample. */
/* This algorithm will, with probability at least $1 - \delta$, output a hypothesis with
error at most $\epsilon$
        given that $|S| \geq \max\{\frac{4}{\epsilon}\log\frac{2}{\delta}, \frac{16dslog3s}{\epsilon}\log\frac{13}{\epsilon}\}$. */

  1. $h := \emptyset$ /* The set of boxes in the hypothesis; represented as upper corners */
  2. $P := \{x : x \in S, x \text{ is a positive example}\}$
  3. while there exists an example $x \in P$
      (a) $P := P - \{x\}$
      (b) for each $y \in P$ if member(maxCorner$\{x, y\}$) = "yes" then
           i. $x :=$ maxCorner$\{x, y\}$
          ii. $P := P - \{y\}$
      (c) add box($x$) to $h$
  4. return $h$ /* That is, output the union of boxes in $h$*/
```

Figure 2: Algorithm to learn a union of origin-incident boxes.

complexity polynomial in s, d, $1/\epsilon$, $1/\delta$.

Proof: To prove the theorem, we show that

1. Algorithm LearnBPQ (Figure 2), takes as input a sample S, runs in time polynomial in S, and outputs a union of at most s origin-incident boxes (that is, an element of $\mathrm{BPQ}_{\cup}(s)$) that is consistent with the sample.

2. The VC-dimension[5] of $\mathrm{BPQ}_{\cup}(s)$ grows polynomially with s and d (namely, it is at most $2ds \log 3s$).

It then follows from Theorem 2.1 of Blumer et al. [15] that if LearnBPQ is given a sample of cardinality at least $m = \max\left\{\frac{4}{\epsilon}\log\frac{2}{\delta}, \frac{16ds\log(3s)}{\epsilon}\log\frac{13}{\epsilon}\right\}$, then with probability at least $1 - \delta$, it will output a hypothesis h with error at most ϵ.

To see that (2) is true, note that the VC-dimension of BPQ is at most d (this is easily shown), and by Lemma 3.2.3 of Blumer et al. [15], the VC-dimension of $\mathrm{BPQ}_{\cup}(s)$ is at most $2ds\log(3s)$. To complete the proof, it remains to be shown that (1) holds. We first show that LearnBPQ produces a hypothesis that is consistent with the sample S. The hypothesis produced is consistent with the positive examples of S since the algorithm does not terminate until all positive examples of S have been removed from P and no point is removed unless the box about to be placed in h contains it. Furthermore, if box(x) was placed in h, then x was a positive example (either it was in P or verified to be positive with the membership query **member**(x)). Since x is a positive example, box(x) is contained within some box of the target. Thus no negative points (even those not in S) can be contained in any of the boxes placed in h.

We now prove the hypothesis h output by LearnBPQ contains at most s boxes. Suppose h contained more than s boxes. Since each box of h is contained within a box of the target, it follows that there must be at least two boxes (say b_i and b_j) in h that are contained within the same box (say b_k^*) of the target. Assume, without loss of generality, that b_i was placed in h first. Let p_i be the point from P selected in step 3 during the iteration of the while loop in which b_i was added to h. Thus p_i must be contained within b_i. Likewise, let p_j be the point from P selected during

[5] The VC-dimension is a combinatorial parameter of a concept class that directly relates to the number of examples necessary (and sufficient) for sufficient generalization [15].

the iteration of the while loop in which b_j was added to h. (So p_j is in b_j.) Since $p_j \in P$ after b_i was placed in h, a membership query must have been performed on maxCorner$\{p'_i, p_j\}$, where box(p'_i) contains p_i. Furthermore, since p_j was not removed during the construction of b_i, it follows that maxCorner$\{p'_i, p_j\}$ is a negative example. Since p_i is contained within box(p'_i) it must be that maxCorner$\{p_i, p_j\}$ is also a negative example. Recall that the box b^*_k of the target contains b_i and b_j and thus b^*_k contains p_i and p_j. However, this contradicts the fact that maxCorner$\{p_i, p_j\}$ is a negative example. Thus h contains at most s boxes.

Finally, LearnBPQ runs in polynomial time, since there are at most s iterations of the while loop, each taking $O(|S|)$ time. This completes the proof of (1) above, and hence of the theorem. □

3.3.2 Learning the Agreement of Boxes with Samplable Intersection

We now give an algorithm to learn the agreement of s boxes in E^d (hence, an algorithm to learn boxes from a consistently ignorant teacher) when the intersection region is samplable. Our algorithm, Figure 7, has polynomial time and sample complexity in both d and s when $d = O(\log s)$. The intuition behind our algorithm lies in the way in which the non-empty intersection of a set of boxes can be used to partition E^d into 3^d sub-regions. Let B be the set of boxes for which we are computing the agreement. Figure 3 illustrates the effect of this partitioning on a typical box $b \in B$. The large, transparent box is b, and the solidly shaded box b_I in the center is the intersection of all boxes in B (and thus contained in b). By infinitely extending the faces of b_I we decompose b into a set of sub-boxes that are also axis-parallel. In general, there are 3^d sub-regions in E^d as seen informally by first observing that the bounds of the intersection region are d pairs of parallel hyperplanes, one pair of parallel hyperplanes for each dimension. Thus, in each dimension the sub-region lies either above both of the hyperplanes, lies between the pair, or lies beneath both of the hyperplanes.

Sub-regions and Sub-boxes: A useful way to categorize these 3^d sub-regions is by the dimension of the boundaries they share with the intersection region.

Since, by definition, the intersection region is contained in every box in B, the manner in which these boxes can overlap in a sub-region is restricted based on the dimension

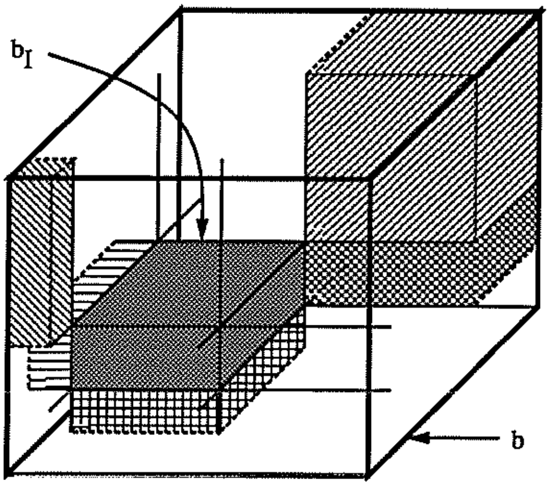

Figure 3: Decomposition of an axis-parallel box with respect to the intersection box b_I. (Note that only some of the 27 sub-regions induced by b_I are shaded.)

of the boundary the sub-region shares with the intersection region. Figure 4 illustrates the constraints imposed by the dimension of the shared boundary—the higher the dimension of the shared boundary, the greater the number of dimensions constrained by the intersection.

To eliminate the dimensions of a sub-region that are already constrained by the intersection we introduce the following notation. Let $p = \langle x_1, x_2, \ldots, x_d \rangle$ be a point in E^d, and let I be a set of indices $\{i_1, i_2, \ldots, i_k\}$ such that $1 \leq i_1 < i_2 < \ldots < i_k \leq d$. The point $\pi_I(p) = \langle x_{i_1}, x_{i_2}, \ldots, x_{i_k} \rangle$ in E^k is the *projection* of p with respect to I. In general, if a sub-region shares a k-dimensional boundary with a d-dimensional intersection region, then for any sub-box in that sub-region we need only determine the sub-box's extent in the remaining $d-k$ dimensions. Then boxes in the same sub-region can be translated to all be origin-incident boxes in a $d-k$ dimensional space for which we can apply LearnBPQ.

Removing Intersection Box Estimation Error:

There remains a subtle point that we must address. So far we have assumed that we know the intersection region *exactly*. However, in reality, we apply a known PAC-algorithm [15] to obtain a good approximation of the intersection region; the approximation box is contained in the intersection region. To obtain an approximation with error at most ϵ with probability at least $1 - \delta$, this algorithm draws a sample of size

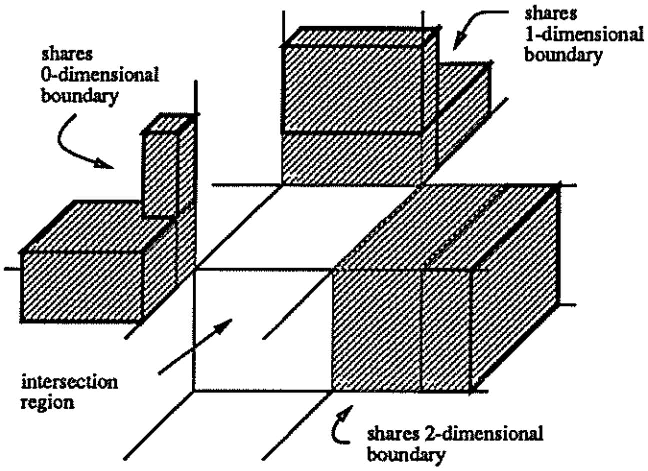

Figure 4: Sub-region constraints imposed by the dimension of the boundary shared with the intersection region

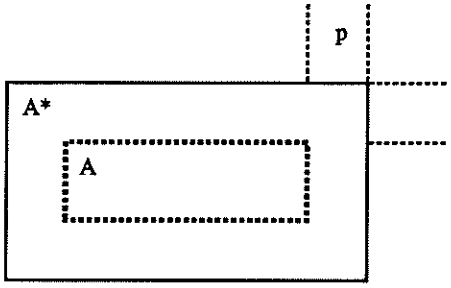

Figure 5: An example assigned to the wrong sub-region.

$\max\left(\frac{4}{\epsilon}\log\frac{2}{\delta}, \frac{16d}{\epsilon}\log\frac{13}{\epsilon}\right)$ and returns the smallest box that is consistent with the sample. Let IBox(S) be a procedure that takes a sample S and returns the smallest box consistent with S. To apply this algorithm to learn the intersection region of the boxes in our model, we simply modify the sample by changing all "?" examples to negative examples.

The difficulty here is that the sub-region in which a point p lies may differ when subdividing based on the true intersection region versus sub-dividing based on the underestimate for the intersection region. Figure 5 illustrates how this may happen. A^* is the true intersection region and A is an underestimate of A^*; the point p lies between the vertical boundaries of A^*, but lies to the right of the vertical boundaries of A. We handle this by discretizing E^d with an irregular Cartesian grid.

```
Expand($A$,$S$)
/* $A$ is a non-empty underestimate of the true intersection region. */
/* $S$ is a set of labeled example points. */

1. Let $\langle a_1, a_2, \ldots, a_d \rangle$ be any point contained in $A$.
2. For $i := 1$ to $d$
    (a) /* Use membership queries to obtain a lower bound for dimension $i$. */
        Set $\ell_i$ to be the smallest $v$ such that $v = \pi_{\{i\}}(p)$ for some $p \in S$ and
        $\langle a_1, a_2, \ldots, a_{i-1}, v, a_{i+1}, \ldots, a_d \rangle$ is a positive example.
    (b) /* Use membership queries to obtain an upper bound for dimension $i$. */
        Set $u_i$ to be the largest $v$ such that $v = \pi_{\{i\}}(p)$ for some $p \in S$ and
        $\langle a_1, a_2, \ldots, a_{i-1}, v, a_{i+1}, \ldots, a_d \rangle$ is a positive example.
3. Return the box $A'$ having opposing corners $\langle \ell_1, \ell_2, \ldots, \ell_d \rangle$ and $\langle u_1, u_2, \ldots, u_d \rangle$.
```

Figure 6: Algorithm to expand an underestimate A of the the true intersection region A^* to an estimate A' such that for any point p in S, the sub-region generated by A' in which p lies is the same as the sub-region generated by A^* in which p lies.

Suppose we have a collection S of points from E^d. For each dimension i consider the set $S_i = \{\pi_{\{i\}}(p) : p \in S\}$. Notice that S_i is a collection of points from E^1 and that if we consider labeling the coordinate axis for dimension i of E^d using only values found in S_i, then we will have effectively discretized E^d in such a way that *every* point of the sample S lies at some intersection point of the resulting irregular Cartesian grid. We then expand our estimate A of the true intersection region A^* in such a way that for every point p in S, the sub-region generated by A in which p lies and the sub-region generated by A^* in which p lies are the same. An algorithm to achieve this goal is given in Figure 6. We state the following lemma without proof.

Lemma 8 *Let A be a non-empty underestimate of the true intersection region A^*. The algorithm* Expand(A, S) *outputs a box A' so that for all $p \in S$, the sub-region generated by A' in which p lies and the sub-region generated by A^* in which p lies are the same. Furthermore,* Expand *runs in time polynomial in the size of S.*

The Full Algorithm: Putting the pieces together we obtain our algorithm. We first approximate the intersection region, and then refine this estimate using Expand.

Next we apply a version of LearnBPQ of suitable dimension to the points in the various sub-regions generated by the intersection region. Finally, we combine the hypotheses obtained from the calls to LearnBPQ along with our estimate of the intersection region to obtain our final hypothesis. Our algorithm is shown in Figure 7. We now state the theorem.

Theorem 9 *Let p^+ be the probability of receiving a positive example from the example oracle. LearnBoxesAgreement is a PAC-MEMB algorithm for learning the agreement of s axis-parallel boxes in E^d that has time complexity $O(sm)$, and sample complexity $m - O\left(\frac{9^d}{\epsilon^2}\log\frac{3^d}{\delta} + \frac{9^d}{\epsilon^2}ds\log s\log\frac{3^d}{\epsilon} + \frac{1}{p^+}\log\frac{1}{\delta}\right)$.*

Proof Sketch: Note that by drawing a sample of size $1/p^+ \ln 2/\delta$ with probability at least $1 - \delta/2$ we will obtain a positive example. It follows directly from Blumer et al. [15] that our sample suffices to ensure that the hypothesis output by IBox(T) has error at most $\epsilon/3^d$ with probability at least $1 - \frac{\delta}{2\cdot 3^d}$. Applying Chernoff bounds [13] it can be shown that for each of the $3^d - 1$ remaining sub-regions, the sample is sufficiently large so that with probability at least $1 - \frac{\delta}{2\cdot 3^d}$ there are enough points in any sub-region of weight at least $\epsilon/3^d$ so that the hypothesis output by LearnBPQ for that region has error at most $\epsilon/3^d$. (Note that sub-regions with weight less than $\epsilon/3^d$ can contribute at most $\epsilon/3^d$ to the total error.) It follows from Lemma 8 that the total error of our final hypothesis is the sum of the errors of the hypotheses we generate for each of the 3^d regions. Thus the probability that the error of the final hypothesis is more than $3^d \cdot \epsilon/3^d = \epsilon$ is at most $3^d \frac{\delta}{2\cdot 3^d} = \delta/2$. Finally, it is easily shown that the time complexity is polynomial in the sample complexity. □

4 A Negative Result

The class of conjunctions of Horn clauses (Horn sentences) is known to be PAC-MEMB-learnable [7]. Furthermore, Frazier and Pitt [22] have shown that Horn sentences are efficiently learnable using membership and equivalence queries from a different model in which entailed examples are provided.

We provide evidence that this result cannot be strengthened to allow learning blurry

```
LearnBoxesAgreement()

1. Draw a sample $S$ of size $m := \max\left\{\frac{8\cdot 9^d}{\epsilon^2}\lg\frac{4\cdot 3^d}{\delta}, \frac{32\cdot 9^d\cdot ds\lg(3s)}{\epsilon^2}\lg\frac{13\cdot 3^d}{\epsilon}, \frac{1}{p^+}\lg\frac{1}{\delta}\right\}$.
2. If there are no positive examples halt and report failure.
3. Let $T$ be set of examples obtained by relabeling all "?" examples of $S$ as negative.
4. Set $A :=$ Expand(IBox($T, S$)).
5. Let $R$ be the set of sub-regions generated by $A$ (excluding $A$ itself).
6. For each sub-region $r \in R$
   (a) Choose any point $p_r$ in the boundary shared by $A$ and the sub-region $r$ such
       that $p_r$ is an extreme point in every dimension of the boundary. Let $f_r$ be the
       coordinate transformation that translates $f_r$ to the origin of $E^d$.
   (b) /* Identify dimensions for which we already know the extent of any sub-box
       lying in $r$ */
       Let $I_r$ be the dimensions for which $r$ is not bounded between a pair of parallel
       hyperplane bounds for $A$.
   (c) /* Project out those dimensions for any point of $S$ that lies in $r$, relabel "?"
       examples */
       Let $S_r := \{p' \mid p \in r \text{ and } p' = \pi_{I_r}(f_r(p))\}$.
       If $p' \in S_r$ is labeled with "?" then relabel it as positive.
   (d) Set $B_r$ to be the set of boxes returned by LearnBPQ($S_r$).
7. Given any unlabeled example $x$, predict
   $\begin{cases} 1 & \text{if } x \text{ lies in } A \\ ? & \text{if } \exists r \in R, b \in B_r \text{ such that } x \text{ lies in sub-region } r \text{ and } \pi_{I_r}(f_r(x)) \text{ lies in } b \\ 0 & \text{otherwise} \end{cases}$
```

Figure 7: The algorithm LearnBoxesAgreement for learning the agreement of a set of axis-parallel boxes with samplable intersection region.

Horn sentences, by showing that such an algorithm could be used to learn the class of (nonblurry) DNF formulas.

Let DHF represent the class of disjunction of Horn Sentences. We now demonstrate that the class of DNF formulas is a subset of the class of DHF formulas.

Claim 10 *For any DNF formula f there exists a logically equivalent DHF formula f' for which $|f'|$ is polynomial in $|f|$.*

Proof: Observe that every unnegated literal v is equivalent to the Horn clause $(T \rightarrow v)$ and every negated literal $\bar{v}$ is equivalent to the Horn clause $(v \rightarrow F)$. For example, $a\bar{b}c \equiv (T \rightarrow a)(b \rightarrow F)(T \rightarrow c)$. Thus we can represent each term by a Horn sentence and take the disjunction of these Horn sentences to build a DHF formula that is logically equivalent to the given DNF formula. Finally, observe that the size of the DHF formula created by this transformation has size polynomial in the DNF formula from which it was created. □

Using the above observation, it is easily shown that the problem of learning an agreement of Horn sentences (without any restrictions) is as hard as learning DNF. However, as demonstrated by our algorithm to learn the agreement of boxes, if the intersection of the Horn sentences in the agreement were non-empty then it may be possible to use the intersection information to successfully learn the disjunction. We claim the stronger negative result that learning the agreement of Horn sentences even when the intersection region is samplable is as hard as learning the class of DNF formulas.

Theorem 11 *PAC-MEMB learning the agreement of Horn sentences for which the intersection region is samplable is as hard as PAC-MEMB learning the class of DNF formulas.*

Proof: We prove this through a sequence of prediction preserving reductions [9, 37]. Let DHF-1pos be the class of DHF formulas with exactly one positive example p that satisfies every disjunct. Let agree-Horn-1pos be the agreement of Horn sentences that have exactly one example in their intersection. Finally, let agree-Horn be the agreement of Horn sentences with a samplable intersection region.

Applying Claim 10, it follows that the learnability of DHF implies the learnability of DNF. We now give a reduction showing that the learnability of DHF-1pos (even when the learner knows the single positive example) implies the learnability of DHF. Let f be the target of the DHF algorithm and f_p be the target of the DHF-1pos algorithm. Assume without loss of generality the positive example p known to the learner is the zero vector. We construct f_p from f by adding an extra literal v to the antecedent of every Horn clause in f as well as adding the Horn sentence $(v \rightarrow F)(v_1 \rightarrow F)\cdots(v_n \rightarrow F)$. Note that the only example satisfying every disjunct of f_p is the zero vector p.

The DHF algorithm A simulates the queries for the DHF-1pos algorithm A_+ as follows: When A_+ requests an example, A obtains a random example x (that assigns values only to $v_1, \ldots, v_n$), generates the example x' that is like x with the additional variable v set to 1, and gives x' to A_+. When A_+ makes a membership query on x, if $v = 0$ then A returns "1" and if $v = 1$ then A respond with the result of a membership query on the instance x' that is just x with the setting for the variable v eliminated. Once A_+ terminates with hypothesis h_+, A is able to predict the label of any example, x, by setting v to 1 and evaluating f_+ on that example. Note that setting v to 1 causes the added Horn sentence in f_p to evaluate to 0 and the antecedents of all the remaining Horn clauses to not be affected by x.

We now show that the learnability of agree-Horn-1pos implies the learnability of DHF-1pos. It is at this point that we switch from learning a standard boolean concept to learning an agreement. Note that the learning problem for the class DHF-1pos assumes that the learner knows the single positive example that satisfies every disjunct of the the target. Any algorithm for agree-Horn-1pos can be used to learn DHF-1pos by simply providing the sole positive example of agree-Horn-1pos to DHF-1pos as p and changing all "?" examples to positive examples.

Finally, we show that the learnability of agree-Horn implies the learnability of agree-Horn-1pos. Recall that a PAC-MEMB learning algorithm must learn under any distribution D. When the agree-Horn algorithm requests a random example, the simulation algorithm flips a fair coin. With probability 1/2, the simulation provides the agree-Horn algorithm with the single positive point in agree-Horn-1pos (and thus the positive region is samplable). Otherwise, a random example drawn from the oracle is given to the agree-Horn algorithm. Clearly agree-Horn is a generalization of agree-Horn-1pos and thus at least as hard.

Thus it follows from this sequence of reductions that PAC-MEMB learning the agreement of Horn sentences with a samplable positive region is as hard as PAC-MEMB learning the class of DNF formulas. □

Finally, we strengthen this result by using the hardness result of Angluin and Kharitonov [9] which shows, under the assumption that public key encryption is secure, membership queries do not help in learning DNF formulas (with an unbounded number of terms).

Corollary 12 *PAC-MEMB learning the agreement of Horn sentences for which the intersection region is samplable is as hard as PAC-learning the class of DNF formulas assuming that public key encryption is secure.*

Acknowledgements

We thank Dana Angluin and Haym Hirsh for helpful conversations regarding this work.

References

[1] Howard Aizenstein, Lisa Hellerstein, and Leonard Pitt. Read-thrice DNF is hard to learn with membership and equivalence queries. In *33nd Annual Symposium on Foundations of Computer Science*, pages 523–532, October 1992.

[2] Howard Aizenstein and Leonard Pitt. Exact learning of read-twice DNF formulas. In *32nd Annual Symposium on Foundations of Computer Science*, pages 170–179, October 1991.

[3] Dana Angluin. Learning k-term DNF formulas using queries and counterexamples. Technical Report YALEU/DCS/RR-559, Yale University, August 1987.

[4] Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75:87–106, November 1987.

[5] Dana Angluin. Negative results for equivalence queries. *Machine Learning*, 5:121–150, 1990.

[6] Dana Angluin. Exact learning of μ-DNF formulas with malicious membership queries. Technical Report YALEU/DCS/TR-1020, Yale University, March 1994.

[7] Dana Angluin, Michael Frazier, and Leonard Pitt. Learning conjunctions of Horn clauses. *Machine Learning*, 9:147–164, 1992.

[8] Dana Angluin, Lisa Hellerstein, and Marek Karpinski. Learning read-once formulas with queries. *Journal of the Association for Computing Machinery*, 40(1):185–210, 1993.

[9] Dana Angluin and Michael Kharitonov. When won't membership queries help? In *Proceedings of the Twenty Third Annual ACM Symposium on Theory of Computing*, pages 444–454, May 1991.

[10] Dana Angluin and Martin Krikis. Learning with malicious membership queries and exceptions. In *Proceedings of the Seventh Annual Workshop on Computational Learning Theory*, July 1994.

[11] Dana Angluin and Philip Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

[12] Dana Angluin and Donna K. Slonim. Randomly fallible teachers: Learning monotone DNF with an incomplete membership oracle *Machine Learning*, 14:7–26, 1994.

[13] Dana Angluin and Leslie G. Valiant. Fast probabilistic algorithms for Hamiltonian circuits and matchings. *Journal of Computer and System Sciences*, 18(2):155–193, April 1979.

[14] Avrim Blum and Steven Rudich. Fast learning of k-term DNF formulas with queries. In *Proceedings of the Twenty Fourth Annual ACM Symposium on Theory of Computing*, pages 382–389, May 1992.

[15] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association for Computing Machinery*, 36(4):929–965, October 1989.

[16] Nader H. Bshouty. Exact Learning via the Monotone Theory. In *34th Annual Symposium on Foundations of Computer Science*, pages 302–311, November 1993.

[17] Nader H. Bshouty, Thomas R. Hancock, and Lisa Hellerstein. Learning arithmetic read-once formulas. In *Proceedings of the Twenty Fourth Annual ACM Symposium on Theory of Computing*, pages 370–381, May 1992.

[18] Nader H. Bshouty, Thomas R. Hancock, and Lisa Hellerstein. Learning Boolean read-once formulas with arbitrary symmetric and constant fan-in gates. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 1–15, August 1992.

[19] Zhixiang Chen. Learning unions of two rectangles in the plane with equivalence queries. In *Proceedings of the Sixth Annual ACM Conference on Computational Learning Theory*, pages 243–252. ACM Press, July 1993.

[20] Zhixiang Chen and Steven Homer. Fast learning unions of rectangles with queries. Unpublished manuscript, July 1993.

[21] Zhixiang Chen and Wolfgang Maass. On-line learning of rectangles. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 16–27. ACM Press, July 1992.

[22] Michael Frazier and Leonard Pitt. Learning from entailments: an application to propositional Horn sentences. In *Machine Learning Proceedings of the Tenth International Conference*, pages 120–127, June 1993.

[23] Michael Frazier and Leonard Pitt. CLASSIC Learning. In *Proceedings of the Seventh Annual Workshop on Computational Learning Theory*, July 1994.

[24] Paul W. Goldberg, Sally A. Goldman, and H. David Mathias. Learning unions of rectangles with membership and equivalence queries. In *Proceedings of the Seventh Annual Workshop on Computational Learning Theory*, July 1994.

[25] Sally A. Goldman, Michael J. Kearns, and Robert E. Schapire. Exact identification of circuits using fixed points of amplification functions. *SIAM Journal on Computing*, 22(4):705–726, August 1993.

[26] Sally A. Goldman and Robert H. Sloan. Can PAC learning algorithms tolerate random attribute noise? Technical Report WUCS–92–25, Washington University, Department of Computer Science, July 1992. To appear in Algorthmica.

[27] Thomas R. Hancock. Learning 2μ DNF formulas and $k\mu$ decision trees. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 199–209, August 1991.

[28] David Haussler. Generalizing the PAC model: sample size bounds from metric dimension-based uniform convergence results. In *30th Annual Symposium on Foundations of Computer Science*, pages 40–45, October 1989.

[29] David Haussler, Michael Kearns, Nick Littlestone, and Manfred K. Warmuth. Equivalence of models for polynomial learnability. *Information and Computation*, 25(2):129–161, December 1991.

[30] Michael Kearns and Ming Li. Learning in the presence of malicious errors. *SIAM Journal on Computing*, 22(4):807–837, August 1993.

[31] Michael Kearns and Leslie Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, pages 433–444, May 1989. To appear in JACM.

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

[33] Philip D. Laird. *Learning from Good and Bad Data.* Kluwer international series in engineering and computer science. Kluwer Academic Publishers, Boston, 1988.

[34] Nicholas Littlestone. Redundant noisy attributes, attribute errors, and linearythreshold learning using winnow. In *Fourth Workshop on Computational Learning Theory*, pages 147–156, 1991.

[35] Philip M. Long and Manfred K. Warmuth. Composite geometric concepts and polynmomial predictability. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 273–287. Morgan Kaufmann, August 1990.

[36] Thomas Mitchell. Generalization as Search. *Artificial Intelligence*, 18:203–226, 1982.

[37] Leonard Pitt and Manfred K. Warmuth. Prediction preserving reducibility. *Journal of Computer and System Sciences*, 41(3):430–467, December 1990.

[38] Yasubumi Sakakibara. On learning from queries and counterexamples in the presence of noise. *Information Processing Letters.* To appear.

[39] George Shackelford and Dennis Volper. Learning k-DNF with noise in the attributes. In *Proceedings of the 1988 Workshop on Computational Learning Theory*, pages 97–103. Morgan Kaufmann, August 1988.

[40] Robert H. Sloan. Types of noise in data for concept learning. In *Proceedings of the 1988 Workshop on Computational Learning Theory*, pages 91–96. Morgan Kaufmann, 1988.

[41] Leslie Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.