Machine Learning, 45, 123–145, 2001
© 2001 Kluwer Academic Publishers. Manufactured in The Netherlands.

Learning with Maximum-Entropy Distributions*

YISHAY MANSOUR mansour@math.tau.ac.il
MARIANO SCHAIN mariano@math.tau.ac.il
Computer Science Department, Tel-Aviv University, Tel-Aviv, Israel

Editor: Lisa Hellerstein

Abstract. We are interested in distributions which are derived as a maximum entropy distribution from a given set of constraints. More specifically, we are interested in the case where the constraints are the expectation of individual and pairs of attributes. For such a given maximum entropy distribution (with some technical restrictions) we develop an efficient learning algorithm for read-once DNF. We extend our results to monotone read-k DNF following the techniques of (Hancock & Mansour, 1991).

Keywords: PAC-learning, maximum entropy, learning algorithms

1. Introduction

The PAC learning model (Valiant, 1984) is a basic model in computational learning theory. Its introduction brought forth a simple set of assumptions and raised many challenging problems. Initially, the main goal was a computational one, to develop new algorithms within this framework and show the learnability of different concept classes. The PAC model has been very successful in the study of the tradeoff between sample size versus accuracy and confidence, but less successful in the algorithmic study. Only a few algorithmic techniques were developed, and many simple concept classes have proven to be computationally intractable. We recall two intractability results in the PAC model, namely, efficient learning 3-term DNF (Pitt & Valiant, 1988) (for learning using a hypothesis which is a 3-term DNF) and efficient learning polynomial size formula (Kearns & Valiant, 1989) (for learning using any hypothesis).

In addition, since the distribution may be arbitrary, it is rather simple to show equivalence in difficulty of related concept classes, such as general DNF and monotone read-once DNF (Haussler et al., 1991). A much more complicated and non-intuitive result is showing that if DNF is learnable with membership queries in the PAC model, then DNF can also be learned without membership queries (Angluin & Kharitonov, 1991).

Researchers were aware that the need to learn a concept class with respect to an *arbitrary* distribution is one of the main sources of intractability in the PAC model. Research was done on learnability with respect to a fixed distribution, and a model, similar to the PAC model, was suggested, where the learning algorithm may depend on the underlying distribution

*This research was supported in part by a grant from the Israel Science Foundation. A preliminary version of this paper appeared in the 10th Annual Conference on Computational Learning Theory.

(Benedek & Itai, 1991; Natarajan, 1987). They also gave examples where such models allowed the learning of classes, not learnable before.

When looking for learning with respect to a fixed distribution rather than an arbitrary one, it is most important to choose "natural" distributions. The most common distributions are the uniform and product distributions. For example, with respect to such distributions one can learn read-once DNF (Kearns, Li, & Valiant, 1994), although with respect to an arbitrary distribution it is equivalent to learning general DNF, which is a major open problem in computational learning theory.

This was the starting point of our research; we were interested in expanding the set of "natural" distributions that are considered. One huge difference between an arbitrary distribution and a uniform/product distribution is the independence assumption. In a product distribution, by definition, the values of the various attributes are independent. This is a very severe handicap when considering "natural" distributions, and clearly helps the learning algorithm. We are interested in considering a wide class of distributions where the independence assumption does not hold. First let us deviate and discuss maximum entropy distributions.

The entropy is a functional over probability distributions. Intuitively it measures the uncertainty of an observer over the outcome of an experiment (see, e.g., Cover and Thomas (1991)). Maximizing the entropy may be viewed in many cases as trying to adopt the "most permissive" distribution that obeys certain constraints. We view our setting as being given a set of constraints on the distribution, each constraint being an expectation of a predicate of the attributes. Later, assuming that there are distributions that satisfy a set of constraints, we focus on the distribution that maximizes the entropy.

To motivate this, let us consider a few special cases. First, if there are no constraints, then all the distributions are possible, and the maximum entropy distribution is the uniform distribution. Second, if we are given the expectations of individual attributes as constraints and no other constraint, the maximum entropy is the product distribution, where each attribute has the desired expectation. The case that we choose to concentrate on in this work is that in which we have expectations of individual variables and pairs of variables (i.e., probability that two attributes simultaneously have the value 1).

There is a vast literature on the subject of maximum entropy distributions, a subject of special interest in physics (Smith & Grandy, 1985). One of the basic results is the form of the maximum entropy distribution, given a set of constraints. Unfortunately, from a computational point of view, it is NP-complete to decide if there is any distribution that satisfies the set of pair-wise constraints (Koller & Megiddo, 1993; Kilian & Naor, 1995).

As stated before, learning the read-once DNF concept class over an arbitrary distribution is equivalent to learning DNF (Haussler et al., 1991). Constraining the underlying probability distribution to uniform or product distribution enables one to learn read-once DNF (Kearns, Li, & Valiant, 1994). However, the algorithms for learning read-once DNF over uniform or product distributions depend heavily on the fact that the attributes are independent. This is why it is interesting to develop an efficient learning algorithm for read-once DNF under maximum entropy distributions. This enables us to extend our learnability horizon and include many distribution classes which do not have the independence assumption.

We present an efficient learning algorithm for read-once DNF over the maximum entropy distribution for pairwise expectations (assuming that the expectations are bounded from 0 and 1). Our algorithm receives the parameters of the maximum entropy distribution, and based on them and on random examples from the target concept, finds a read-once DNF that approximates the target read-once DNF.

Our algorithm is based on a definition of the *influence* of a variable on the value of a DNF, and on the observation that for maximum entropy distributions the sign of a variable in a DNF is the same as the sign of that *influence*. We show that given a method to retrieve the sign of the influence of a variable, the target DNF can be reconstructed. An efficient method to retrieve the sign of the influence is then described, implying the PAC learnability result.

Intuitively, the influence of a variable on a boolean function is the probability that flipping the variable's value changes the output of the function. By our definition, the *influence* is composed of two expected values—the difference between the expectations of the function with the variable fixed for each of its two possible values. The first of these expectations can be directly estimated. For the second expectation, since the variables are not independent, a special *decorrelating* transformation was designed and applied.

Having a method to approximately measure the influence, we prove a separation result. We first define a measure of the relative influence of a variable on the other variables. We show that for maximum entropy distributions with bounded relative influence (denoted α-bounded, and defined later), either the *influence* is zero or its value is bounded away from zero by a constant that depends polynomially on the desired accuracy parameters. This means that only a reasonable number of samples is needed to recover the sign of the influence up to the desired accuracy—establishing the efficient PAC learnability result of read-once DNF over α-bounded maximum entropy distributions.

We extend the results to monotone read-k DNF. The work of Hancock and Mansour (1991) gives an algorithm that efficiently learns monotone read-k DNF over uniform or bounded product distributions. We follow their techniques and prove that a similar algorithm works for α-bounded maximum entropy distributions as well.

Our algorithm requires only statistical queries. For this reason we use the Statistical Query model (Kearns, 1993) to describe the algorithm. We use a variant of the statistical query model (Aslam & Decatur, 1994) where we have the ability to query the expectation of a real-valued function with a bounded range, instead of only boolean predicates. This variant preserves the fact that learning in it implies PAC learnability even in the presence of random classification noise. As a result, our algorithm is tolerent of random classification noise.

The paper is organized as follows. Section 2 gives the basic definitions and notation to be used in subsequent sections. Section 3 presents the maximum-entropy method and properties of maximum-entropy distributions. The learning algorithm for read-once DNF is described in Section 4. A similar result for monotone read-k DNF is given in Section 5.

2. Model and notation

2.1. PAC learning model

A concept c is a Boolean function $c : \mathcal{X} \to \{0, 1\}$, where $\mathcal{X}$ is the learning domain. An example for c is a pair $(\mathbf{x}, c(\mathbf{x}))$, where $\mathbf{x} \in \mathcal{X}$.

Let D be a fixed probability distribution over $\mathcal{X}$ and $\mathcal{C}$ be a set of concepts. A learning algorithm for concept class $\mathcal{C}$ with respect to distribution D is an algorithm that has access to examples, $(\mathbf{x}, c(\mathbf{x}))$ of an unknown target concept $c \in \mathcal{C}$ where the input $\mathbf{x}$ is drawn independently according to D. The output of the learning algorithm is a hypothesis $h \in \mathcal{C}$. The *error* of h with respect to the distribution D and the target concept c is

$$Error_D(h, c) \overset{\text{def}}{=} Pr_D(h \neq c).$$

Similarly to Benedek and Itai (1991), we say that an algorithm *PAC*-learns the concept class $\mathcal{C}$ over distribution D if for any $c \in \mathcal{C}$ given $\epsilon, \delta > 0$ the algorithm outputs a hypothesis $h \in \mathcal{C}$, such that

$$Pr(Error_D(h, c) \geq \epsilon) \leq \delta.$$

We say that such a learning algorithm is *efficient* if it runs in time polynomial in n, $\frac{1}{\epsilon}$ and $\log \frac{1}{\delta}$.

2.2. Restrictions

We consider the domain $\mathcal{X} = \{0, 1\}^n$. A vector $\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{X}$ is an assignment to the n variables $x_1, \ldots, x_n$. For $1 \leq i \leq n$ we define:

$$\mathbf{x}^i \overset{\text{def}}{=} (x_1, \ldots x_{i-1}, x_{i+1}, \ldots, x_n).$$

For $b \in \{0, 1\}$ we define:

$$\mathbf{x}_{i \leftarrow b} \overset{\text{def}}{=} (x_1, \ldots x_{i-1}, b, x_{i+1}, \ldots, x_n).$$

Let D be a probability distribution over the n-dimensional Boolean domain $\mathcal{X}$. Let f be a Boolean function $f : \mathcal{X} \to \{0, 1\}$. Restricting a function f by fixing the value of a variable x_i to $b \in \{0, 1\}$, is denoted by

$$f_{i \leftarrow b}(\mathbf{x}^i) \overset{\text{def}}{=} f(\mathbf{x}_{i \leftarrow b}) = f(x_1, \ldots x_{i-1}, b, x_{i+1}, \ldots, x_n).$$

Note that $f_{i \leftarrow b}$ can also be considered as a function over the domain $\mathcal{X}$, one that does not depend on x_i.

When we restrict a probability distribution we also need to normalize the sum of the probabilities to 1:

$$D_{i\leftarrow b}(\mathbf{x}^i) \overset{\text{def}}{=} D(\mathbf{x}^i \mid x_i = b) = \frac{D(\mathbf{x}_{i\leftarrow b})}{Pr_D(x_i = b)}.$$

Denote by $E_D[f(x)]$ the expectation of $f(x)$ where x is drawn according to the distribution D. The following lemma is a simple use of Bayes' formula.

Lemma 1. *Let $f(\mathbf{x})$ be a boolean function $f : \{0, 1\}^n \to \{0, 1\}$, and $D(\mathbf{x})$ be a probability distribution over $\{0, 1\}^n$. Then*

$$E_{D_{i\leftarrow b}}[f_{i\leftarrow b}(\mathbf{x}^i)] = \frac{E_D[I_b(x_i)f(\mathbf{x})]}{E_D[I_b(x_i)]},$$

where $I_b(x)$ is the indicator function (i.e., $I_b(x) = 1$ iff $x = b$, otherwise $I_b(x) = 0$).

Proof: The proof follows directly from the definitions:

$$\begin{aligned}
E_{D_{i\leftarrow b}}[f_{i\leftarrow b}(\mathbf{x}^i)] &= \sum_{\mathbf{z}\in\{0,1\}^{n-1}} D_{i\leftarrow b}(\mathbf{z}) f_{i\leftarrow b}(\mathbf{z}) \\
&= \sum_{\mathbf{x}\in\{0,1\}^n} I_b(x_i) \frac{D(\mathbf{x}_{i\leftarrow b})}{Pr_D(x_i = b)} f_{i\leftarrow b}(\mathbf{x}^i) \\
&= \frac{\sum_{\mathbf{x}\in\{0,1\}^n} I_b(x_i) D(\mathbf{x}) f(\mathbf{x})}{Pr_D(x_i = b)} \\
&= \frac{E_D[I_b(x_i)f(\mathbf{x})]}{E_D[I_b(x_i)]}.
\end{aligned}$$

□

The influence of a variable x_i on the boolean function f is the probability that flipping the variable's value changes the output of f. We define a related quantity,

$$\text{Infl}_D(x_i, f) \overset{\text{def}}{=} E_{D_{i\leftarrow 1}}[f_{i\leftarrow 1}(\mathbf{x}^i)] - E_{D_{i\leftarrow 1}}[f_{i\leftarrow 0}(\mathbf{x}^i)].$$

Note that the absolute value of $\text{Infl}_D(x_i, f)$ computes the correct influence for unate functions (functions that are either monotone or anti-monotone in each variable) under product distributions. However, the absolute value of $\text{Infl}_D(x_i, f)$ always gives a lower bound on the real influence, and this will be sufficient for our purpose.

2.3. *Read-once DNF*

A literal ℓ_i is the positive or negative appearance of a variable, e.g. x_i or $\bar{x}_i$. A term m is a conjunction of literals, e.g. $m = x_6 \wedge \bar{x}_8 \wedge x_3$. A DNF is a disjunction of terms, i.e.

$$g(\mathbf{x}) = m_1 \vee m_2 \vee \cdots \vee m_s.$$

A DNF is *read-once* if each variable appears in at most one term. The sign of the variable x_i in the read-once DNF g, denoted Sign (x_i, g), is 1 if x_i appears in g, -1 if $\bar{x}_i$ appears in g, and 0 otherwise.

Let $g(\mathbf{x})$ be a read-once DNF, and m be the term in which x_i appears. We can rewrite m as[1] $m = g_{i1}(\mathbf{x}^i)\ell_i$ where $g_{i1}(\mathbf{x}^i)$ are the other literals in m. Let $g_{i0}(\mathbf{x}^i)$ be the disjunction of all the terms of $g(\mathbf{x})$ except m. Then

$$g(\mathbf{x}) \equiv g_{i0}(\mathbf{x}^i) \vee g_{i1}(\mathbf{x}^i)\ell_i.$$

Let $b_i = 0$ if $\ell_i = x_i$ and $b_i = 1$ if $\ell_i = \bar{x}_i$. Then $g_{i \leftarrow b_i} \equiv g_{i0}$ and $g_{i \leftarrow \bar{b}_i} \equiv g_{i0} \vee g_{i1}$.

2.4. Learning using statistical queries

The *Statistical-Query* learning model (Kearns, 1993) uses an expectation oracle *STAT* that has as input a function $SQ(\mathbf{x}, b)$ which is a boolean function (a predicate) of $x \in \mathcal{X}$ and a target concept value of $b = c(\mathbf{x})$. We use here a generalized model (Aslam & Decatur, 1994) in which *SQ* is a real-valued function rather than a predicate. In order to maintain the simulation in the PAC model we require that the range of *SQ* be bounded by some constant L, i.e.,

$$SQ : \mathcal{X} \times \{0, 1\} \rightarrow [0, L].$$

As in the PAC model we assume a probability distribution $D(\mathbf{x})$ over the domain $\mathcal{X}$. The oracle *STAT* also receives an accuracy parameter $\mu > 0$. For every input (SQ, μ) the oracle *STAT* outputs a value $\hat{E}_{SQ}$ which is μ-close to the true expectation $E_{SQ} = E_D[SQ(bfx, c)]$, i.e.,

$$|\hat{E}_{SQ} - E_{SQ}| \leq \mu.$$

This implies that $O(\log \frac{L}{\mu})$ bits are sufficient to encode *STAT*'s output.

We say that an algorithm learns *using statistical queries* the concept class $\mathcal{C}$ over distribution D if for any $c \in \mathcal{C}$, given $\epsilon > 0$ and access to an oracle *STAT*, the algorithm outputs a hypothesis $h \in \mathcal{C}$ such that $Error_D(h, c) < \epsilon$. It is worth noting that in the statistical query model there is no confidence parameter and we are guaranteed that the output hypothesis is accurate.

We say that such a statistical query algorithm is *efficient* if every function *SQ* given to *STAT* can be evaluated in time polynomial in n and the algorithm runs in time polynomial in n and $\frac{1}{\epsilon}$ (the running time includes the number of statistical queries the algorithm performs), where the query accuracy μ is also bounded by an inverse of a polynomial in those parameters.

It can be shown (Kearns, 1993) that if a concept class $\mathcal{C}$ is *efficiently* learnable *using statistical queries* over distribution D, then $\mathcal{C}$ is *efficiently PAC*-learnable over the distribution D (even in the presence of random classification noise).

We say that the statistical query oracle is *Exact* when $\mu = 0$ (in this case the oracle returns the exact expectation). Since an *Exact* oracle is an ideal entity we ignore its output encoding complexity. We introduce the exact oracle only as a methodological step in developing our algorithm.

3. Maximum-entropy distributions

3.1. The maximum-entropy formalism

The entropy measures the uncertainty of the outcome of an experiment. Its value can be interpreted as the average number of bits needed to encode the samples generated by D. The entropy of a distribution D is

$$H(D) \stackrel{\text{def}}{=} -\sum_{\mathbf{x}\in\mathcal{X}} D(\mathbf{x}) \log D(\mathbf{x}) = E_D[-\log D(\mathbf{x})].$$

We would like to impose constraints on the distribution. A constraint determines the probability of a certain event. (Such constraints may be viewed as a prior.)

Our goal is to find a distribution that satisfies those constraints while imposing the fewest "additional constraints". One way of achieving this, assuming such a distribution exists, is by choosing the distribution that maximizes the entropy subject to the constraints imposed.

We first examine the form of the maximum entropy distribution, satisfying a given set of constraints. Though this problem has been widely studied, we briefly present here the basic results only. The interested reader may refer to Papoulis (1991), Smith and Grandy (1985). We want to determine the probability distribution $D(\mathbf{x})$ subject to the constraints that the expected value of the function $f_i(\mathbf{x})$ is μ_i, i.e.

$$E[f_i(\mathbf{x})] = \mu_i \quad i = 1, \dots, m.$$

Among all distributions satisfying the above constraints, the one with maximum entropy has the form

$$D(\mathbf{x}) = Ae^{-\sum_{i=1}^{m} \lambda_i f_i(\mathbf{x})},$$

where the parameters λ_i are the solution of the system

$$-\frac{\partial}{\partial \lambda_i} \ln Z = \mu_i \quad i = 1, \dots, m,$$
$$\text{and } Z(\lambda_1, \dots, \lambda_m) \stackrel{\text{def}}{=} \sum_{\mathbf{x}\in\mathcal{X}} e^{-\sum_{i=1}^{m} \lambda_i f_i(\mathbf{x})}.$$

The normalization factor is $A = \frac{1}{Z}$. In our special case, we are given n expectations, $E(x_i) = \mu_i$, and n^2 correlations, $E(x_i x_j) = \mu_{ij}$. One can show (Papoulis, 1991) that the distribution D suggested by the maximum entropy method (denoted MEM) has the following form.

Theorem 2 (Papoulis, 1991)**.** *Let D be the maximum entropy distribution that satisfies the constraints $E(x_i) = \mu_i$ and $E(x_i x_j) = \mu_{ij}$. If $D(x) > 0$ for every x, then,*

$$D(\mathbf{x}) = Ae^{-\mathbf{x}^t\mathbf{M}\mathbf{x}-\mathbf{w}^t\mathbf{x}},$$

where $\mathbf{M}$ is a symmetric $n \times n$ matrix and $\mathbf{w} \in \Re^n$ is a vector (we denote by $\mathbf{x}^t$ the vector $\mathbf{x}$ transposed).

Note that since we assume that the distribution has the form that all the inputs have non-zero probability, i.e., $D(x) > 0$ for every x, this implies that none of the μ_{ij} are either 0 or 1.

As stated in the introduction, the related computational problem is hard since testing the consistency of the given expectations (existence of any distribution satisfying those constraints) is known to be *NP*-Complete (Koller & Megiddo, 1993; Kilian & Naor, 1995).

3.2. Properties of MEM distributions

In this section we derive a few properties of MEM distributions. One basic property that is important to us is relating the probabilities when a variable is fixed to 1 or 0. We show that for MEM distributions these probabilities are somewhat related. (Note that in a product distribution they are identical.)

Claim 3. Let D be an MEM distribution with parameters $\mathbf{M}$ and $\mathbf{w}$. For any i, $1 \le i \le n$, there exists a vector $\mathbf{z}_i$ and a scalar r_i such that

$$D(\mathbf{x}_{i\leftarrow 1}) = e^{-\mathbf{z}_i\mathbf{x}^i - r_i} D(\mathbf{x}_{i\leftarrow 0}),$$

where $\mathbf{z}_i = 2M_i^i$ and $r_i = m_{ii} + w_i$ (M_i^i is the i'th row of the matrix $\mathbf{M}$ with the i'th element deleted, and m_{ii} is the i'th element on the diagonal of $\mathbf{M}$).

Proof: Without loss of generality, and to simplify the notation we prove the claim for $i = 1$. We write the symmetric matrix

$$\mathbf{M} = \begin{pmatrix} a & \mathbf{b}^t \\ \mathbf{b} & \mathbf{M}_{11} \end{pmatrix},$$

then

$$\begin{aligned} \ln D(1\,\mathbf{x}^1) &= \ln A - (1\mathbf{x}^1)\begin{pmatrix} a & \mathbf{b}^t \\ \mathbf{b} & \mathbf{M}_{11} \end{pmatrix}\begin{pmatrix} 1 \\ \mathbf{x}^1 \end{pmatrix} - (1\,\mathbf{x}^1)\begin{pmatrix} w_1 \\ \mathbf{w}^1 \end{pmatrix} \\ &= \ln A - (\mathbf{x}^1)^t\mathbf{M}_{11}\mathbf{x}^1 - (\mathbf{x}^1)^t\mathbf{w}^1 - a - 2\mathbf{b}^t\mathbf{x}^1 - w_1 \\ &= \ln D(0\,\mathbf{x}^1) - a - 2\mathbf{b}^t\mathbf{x}^1 - w_1. \end{aligned}$$

Therefore, $D(\mathbf{x}_{1\leftarrow 1}) = e^{-2\mathbf{b}^t\mathbf{x}^1-(a+w_1)} D(\mathbf{x}_{1\leftarrow 0})$. □

A simple conclusion is the following relation for the conditional distributions

$$D_{i\leftarrow 1}(\mathbf{x}^i) = \beta_i e^{-\mathbf{z}_i \mathbf{x}^i - r_i} D_{i\leftarrow 0}(\mathbf{x}^i), \quad \text{where } \beta_i \overset{\text{def}}{=} \frac{Pr_D(x_i = 0)}{Pr_D(x_i = 1)}.$$

The above property allows us to compute the expectation of the restricted function $f_{i\leftarrow 0}(\mathbf{x}^i)$ over the probability distribution $D_{i\leftarrow 1}(\mathbf{x}^i)$. (Note that examples of $f_{i\leftarrow 0}(\mathbf{x}^i)$ are only available over the conditional distribution $D_{i\leftarrow 0}(\mathbf{x}^i)$!) The following lemma shows how this is done.

Lemma 4. *Let D be an MEM distribution with parameters $\mathbf{M}$ and $\mathbf{w}$, and $f(\mathbf{x})$ be a boolean function. Then*

$$E_{D_{i\leftarrow 1}}[f_{i\leftarrow 0}(\mathbf{x}^i)] = E_{D_{i\leftarrow 0}}\left[\beta_i e^{-\mathbf{z}_i \mathbf{x}^i - r_i} f_{i\leftarrow 0}(\mathbf{x}^i)\right],$$

where $\mathbf{z}_i = 2M_i^i$ and $r_i = m_{ii} + w_i$.

We thus have a method to compute the influence $\mathrm{Infl}_D(x_i, f)$ of a variable x_i on a boolean function f over a given MEM distribution D.

4. Learning read-once DNF

4.1. Learning using Exact Statistical Queries oracle

In this section we develop a statistical query algorithm for the special case where we receive the exact expectation each time we query STAT. We use the exact oracle to highlight some of our ideas, and in Section 4.3 we show how to adapt it to the case where STAT returns the expectation with an additive error.

The algorithm itself is very similar to the one for learning read-once DNF under a uniform/product distribution (Kearns, Li, & Valiant, 1994). We first identify the literals on which the target function depends, and then test whether every two such literals appear in the same term.

The following theorem derives a simple condition which enables one to learn read-once DNF over MEM distributions using Exact Statistical Queries oracle (i.e. $\mu = 0$ in the SQ Learning model).

Theorem 5. *Let $P(\mathbf{x})$ be a probability distribution over $\{0, 1\}^{n-1}$ such that for every $\mathbf{z} \in \{0, 1\}^{n-1}$, $P(\mathbf{z}) > 0$. Let $g(\mathbf{x})$ be read-once DNF. Then the following holds:*
1) $E_P[g_{i\leftarrow 1}(\mathbf{x}^i)] > E_P[g_{i\leftarrow 0}(\mathbf{x}^i)]$ *iff x_i appears in $g(\mathbf{x})$;*
2) $E_P[g_{i\leftarrow 1}(\mathbf{x}^i)] < E_P[g_{i\leftarrow 0}(\mathbf{x}^i)]$ *iff $\bar{x}_i$ appears in $g(\mathbf{x})$;*
3) $E_P[g_{i\leftarrow 1}(\mathbf{x}^i)] = E_P[g_{i\leftarrow 0}(\mathbf{x}^i)]$ *iff $g(\mathbf{x})$ does not depend on x_i.*

Proof: If $g(\mathbf{x})$ does not depend on x_i we have $g_{i\leftarrow 1} \equiv g_{i\leftarrow 0}$, and therefore the expectations are identical. Recall the representation $g(\mathbf{x}) \equiv g_{i0}(\mathbf{x}^i) \vee g_{i1}(\mathbf{x}^i)\ell_i$. If x_i appears in $g(\mathbf{x})$ then

$g_{i\leftarrow 1}(\mathbf{x}^i) \equiv g_{i0}(\mathbf{x}^i) \vee g_{i1}(\mathbf{x}^i) \geq g_{i0}(\mathbf{x}^i) \equiv g_{i\leftarrow 0}(\mathbf{x}^i)$. Since g depends on x_i, there is some $\mathbf{z} \in \{0, 1\}^{n-1}$ such that $g_{i\leftarrow 1}(\mathbf{z}) > g_{i\leftarrow 0}(\mathbf{z})$. The lemma assumes that $P(\mathbf{z}) > 0$ for every $\mathbf{z}$; therefore, $E_P[g_{i\leftarrow 1}(\mathbf{x}^i)] > E_P[g_{i\leftarrow 0}(\mathbf{x}^i)]$. The case of $\bar{x}_i$ is similar. □

Assuming that for each pair of attributes all four combinations are possible, the MEM distribution D has the property that $D(\mathbf{x}) > 0$ for every $\mathbf{x}$.

Corollary 6. *Let D be an MEM distribution and $g(\mathbf{x})$ be a read-once DNF. Then for every variable x*

$$\mathrm{Sign}(x, g) = \mathrm{Sign}(\mathrm{Infl}_D(x, g)).$$

Corollary 6 implies that knowing the signs of the influences $\mathrm{Infl}_D(x_i, g)$ is sufficient to decide which literals appear in a read-once DNF.

The following lemma shows how to compute $\mathrm{Sign}(x, g)$.

Lemma 7. *Given the parameters $\mathbf{M}$ and $\mathbf{w}$ of an MEM distribution D, $Sign(x_i, g)$ can be computed using two queries to an exact statistical queries oracle.*

Proof: Using Lemma 4 we write,

$$\begin{aligned}\mathrm{Infl}_D(x_i, g) &\overset{\mathrm{def}}{=} E_{D_{i\leftarrow 1}}[g_{i\leftarrow 1}(\mathbf{x}^i)] - E_{D_{i\leftarrow 1}}[g_{i\leftarrow 0}(\mathbf{x}^i)] \\ &= E_{D_{i\leftarrow 1}}[g_{i\leftarrow 1}(\mathbf{x}^i)] - E_{D_{i\leftarrow 0}}\left[\beta_i e^{-\mathbf{z}_i\mathbf{x}^i - r_i} g_{i\leftarrow 0}(\mathbf{x}^i)\right].\end{aligned}$$

Recall that $\beta_i = E_D[I_0(x_i)]/E_D[I_1(x_i)]$. Using Lemma 1,

$$\begin{aligned}\mathrm{Infl}_D(x_i, g) &= \frac{E_D[g(\mathbf{x})I_1(x_i)]}{E_D[I_1(x_i)]} - \frac{E_D[I_0(x_i)]}{E_D[I_1(x_i)]}\frac{E_D\left[e^{-\mathbf{z}_i\mathbf{x}^i - r_i} g(\mathbf{x}) I_0(x_i)\right]}{E_D[I_0(x_i)]} \\ &= \frac{E_D[g(\mathbf{x})I_1(x_i)] - E_D\left[e^{-\mathbf{z}_i\mathbf{x}^i - r_i} g(\mathbf{x}) I_0(x_i)\right]}{E_D[I_1(x_i)]}.\end{aligned}$$

Therefore,

$$\begin{aligned}\mathrm{Sign}(\mathrm{Infl}_D(x_i, g)) &= \mathrm{Sign}(\mathrm{Infl}_D(x_i, g) E_D[I_1(x_i)]) \\ &= \mathrm{Sign}\left(E_D[g(\mathbf{x})I_1(x_i)] - E_D\left[e^{-\mathbf{z}_i\mathbf{x}^i - r_i} g(\mathbf{x}) I_0(x_i)\right]\right).\end{aligned}$$

It follows by Corollary 6 that to compute $\mathrm{Sign}(x_i, g)$ the required queries are

$$SQ_1(\mathbf{x}, b) \overset{\mathrm{def}}{=} I_1(x_i) b,$$

and

$$SQ_2(\mathbf{x}, b) \overset{\mathrm{def}}{=} e^{-\mathbf{z}_i\mathbf{x}^i - r_i} I_0(x_i) b.$$

The queries given to the oracle yield the exact expectations $E_1 = E_D[g(\mathbf{x})I_1(x_i)]$ and $E_2 = E_D[e^{-\mathbf{z}_i\mathbf{x}^i - r_i} g(\mathbf{x})I_0(x_i)]$, respectively. Since $\text{Sign}(x_i, g) = \text{Sign}(E_1 - E_2)$ the lemma follows. □

The above proof establishes the following identity

$$\text{Infl}_D(x_i, g)E_D[I_1(x_i)] = E_D[g(\mathbf{x})I_1(x_i)] - E_D\big[e^{-\mathbf{z}_i\mathbf{x}^i - r_i} g(\mathbf{x})I_0(x_i)\big].$$

Lemma 7 allows us to find the variables appearing in a read-once DNF $g(\mathbf{x})$ and their sign, under a given MEM probability distribution $D(\mathbf{x})$. Let S be this set of literals. The next step is to find the terms forming $g(\mathbf{x})$. To do this we have to test if every pair of literals in S appears in the same term or not. The following claim allows us to use the same method as before.

Claim 8. Let $\ell_i, \ell_j \in S$. The literals ℓ_i and ℓ_j appear in the same term of a read-once DNF $g(\mathbf{x})$ if and only if $\text{Sign}(x_j, g_{i\leftarrow b_i}) = 0$ (constraining ℓ_i to zero renders g not dependent on ℓ_j). (Recall from Section 2.3 that $b_i = 0$ if $\ell_i = x_i$ and $b_i = 1$ if $\ell_i = \bar{x}_i$.)

The above claim together with Corollary 6 suggests that in order to test if ℓ_i and ℓ_j appear in the same term of $g(\mathbf{x})$ we should check the sign of $\text{Infl}_{D_{i\leftarrow b_i}}(x_j, g_{i\leftarrow b_i})$.

Lemma 9. *Given the parameters* $\mathbf{M}$ *and* $\mathbf{w}$ *of an MEM distribution* D, $\text{Sign}(x_j, g_{i\leftarrow b_i})$ *can be computed using two queries to an exact statistical queries oracle.*

Proof: We prove the lemma for the case $b_i = 0$ (the other case is similar). Without loss of generality, we can assume that $j > i$. We have

$$\text{Infl}_{D_{i\leftarrow 0}}(x_j, g_{i\leftarrow 0}) = E_{D_{i\leftarrow 0, j\leftarrow 1}}[g_{i\leftarrow 0, j\leftarrow 1}(\mathbf{x}^{ij})] - E_{D_{i\leftarrow 0, j\leftarrow 1}}[g_{i\leftarrow 0, j\leftarrow 0}(\mathbf{x}^{ij})],$$

where $\mathbf{z}_{ij}$ is recursively defined as in Claim 3. By Lemma 1,

$$\text{Infl}_{D_{i\leftarrow 0}}(x_j, g_{i\leftarrow 0}) = \frac{E_D[g(\mathbf{x})I_0(x_i)I_1(x_j)] - E_D\big[e^{-\mathbf{z}_{ij}\mathbf{x}^{ij} - r_{ij}} g(\mathbf{x})I_0(x_i)I_0(x_j)\big]}{E_D[I_0(x_i)I_1(x_j)]}.$$

Therefore,

$$\begin{aligned}
&\text{Sign}(\text{Infl}_{D_{i\leftarrow 0}}(x_j, g_{i\leftarrow 0}))\\
&\quad = \text{Sign}(\text{Infl}_{D_{i\leftarrow 0}}(x_j, g_{i\leftarrow 0})E_D[I_0(x_i)I_1(x_j)])\\
&\quad = \text{Sign}\big(E_D[g(\mathbf{x})I_0(x_i)I_1(x_j)] - E_D\big[e^{-\mathbf{z}_{ij}\mathbf{x}^{ij} - r_j} g(\mathbf{x})I_0(x_i)I_0(x_j)\big]\big)
\end{aligned}$$

It follows by Corollary 6 that to compute $\text{Sign}(x_j, g_{i\leftarrow 0})$ the queries needed are,

$$SQ_1(\mathbf{x}, b) \overset{\text{def}}{=} I_0(x_i)I_1(x_j)b$$

and

$$SQ_2(\mathbf{x}, b) \overset{\text{def}}{=} e^{-\mathbf{z}_{ij}\mathbf{x}^{ij}-r_j} I_0(x_i)I_0(x_j)b.$$

The queries given to the oracle yield the exact expectations $E_1 = E_D[g(\mathbf{x})I_0(x_i)I_1(x_j)]$ and $E_2 = E_D[e^{-\mathbf{z}_{ij}\mathbf{x}^{ij}-r_j} g(\mathbf{x})I_0(x_i)I_0(x_j)]$, respectively. Since $\text{Sign}(x_j, g_{i\leftarrow 0}) = \text{Sign}(E_1 - E_2)$ the lemma follows. □

4.2. Bounded MEM distributions

Recall the general form of an MEM distribution $D(\mathbf{x}) = Ae^{-\mathbf{x}^t\mathbf{M}\mathbf{x}-\mathbf{w}^t\mathbf{x}}$. We call an MEM distribution α-bounded if the inner product of any row of $\mathbf{M}$ with every possible $\mathbf{x} \in \mathcal{X}$ is bounded by a constant α. In addition we require that the elements of $\mathbf{w}$ be bounded by α, i.e., for all $\mathbf{x}$ and i

$$|\mathbf{M}_i\mathbf{x}| \leq \alpha \quad \text{and} \quad |w_i| \leq \alpha.$$

As we shall see, the value of α is a measure of the relative influence of a variable on the other variables. For example, for a product distribution to be α-bounded implies that the probability of each variable being 1 is bounded away from both 1 and 0. The relationship between the value of α and the properties of the related MEM distributions is left for future research.

Unlike product distributions, in MEM distributions the value of one variable influences the distribution of the other variables. Still, we prove that this influence is limited.

Lemma 10. *Let $D(\mathbf{x})$ be an α-bounded MEM distribution. Let $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$ differ only in one place, then*

$$\left| \ln \frac{D(\mathbf{x})}{D(\mathbf{y})} \right| \leq 3\alpha.$$

Proof: Without loss of generality, we can assume that $\mathbf{x}$ and $\mathbf{y}$ differ in the first place, i.e.

$$\mathbf{x} = (1\,\mathbf{z}) \quad \text{and} \quad \mathbf{y} = (0\,\mathbf{z}).$$

Write the matrix

$$\mathbf{M} = \begin{pmatrix} a & \mathbf{b}^t \\ \mathbf{b} & \mathbf{M}_1 \end{pmatrix},$$

and the vector $\mathbf{w} = (w_1\mathbf{w}^1)$. By Claim 3

$$\ln D(\mathbf{x}) = \ln D(\mathbf{y}) - a - 2\mathbf{b}^t\mathbf{z} - w_1;$$

therefore,

$$\left|\ln\frac{D(\mathbf{x})}{D(\mathbf{y})}\right| = |a + 2\mathbf{b}^t\mathbf{z} + w_1| \le |a + \mathbf{b}^t\mathbf{z}| + |\mathbf{b}^t\mathbf{z}| + |w_1| \le 3\alpha\,.$$

The last inequality follows from the fact that D is an α-bounded MEM distribution. □

Building on Lemma 10, we show that two vectors that are close in *Hamming* distance have somewhat similar probabilities.

Lemma 11. *Let $d(\mathbf{x}, \mathbf{y})$ be the number of bits in which $\mathbf{x}$ and $\mathbf{y}$ differ. Then*

$$\left|\ln\frac{D(\mathbf{x})}{D(\mathbf{y})}\right| \le 3\alpha d(\mathbf{x}, \mathbf{y}).$$

Proof: Form a series of vectors, $\mathbf{z}_0 = \mathbf{x}, \mathbf{z}_1, \ldots, \mathbf{z}_{d-1}, \mathbf{z}_d = \mathbf{y}$ such that $d = d(\mathbf{x}, \mathbf{y})$ and $d(\mathbf{z}_i, \mathbf{z}_{i+1}) = 1$ for $i = 0, \ldots, d-1$. It follows that

$$\begin{aligned}\left|\ln\frac{D(\mathbf{x})}{D(\mathbf{y})}\right| &= |\ln D(\mathbf{x}) - \ln D(\mathbf{y})| = \left|\sum_{i=0}^{d-1} \ln D(\mathbf{z}_i) - \ln D(\mathbf{z}_{i+1})\right| \\ &\le \sum_{i=0}^{d-1} |\ln D(\mathbf{z}_i) - \ln D(\mathbf{z}_{i+1})| = \sum_{i=0}^{d-1}\left|\ln\frac{D(\mathbf{z}_i)}{D(\mathbf{z}_{i+1})}\right|.\end{aligned}$$

Now, by Lemma 10,

$$\left|\ln\frac{D(\mathbf{x})}{D(\mathbf{y})}\right| \le \sum_{i=0}^{d-1}\left|\ln\frac{D(\mathbf{z}_i)}{D(\mathbf{z}_{i+1})}\right| \le 3\alpha d\,.$$ □

A case which will be of special interest to us is when the distance between $\mathbf{x}$ and $\mathbf{y}$ is logarithmic.

Corollary 12. *For a given α-bounded MEM distribution D, if $d(\mathbf{x}, \mathbf{y}) \le c\ln\frac{n}{\epsilon}$, then*

$$(n/\epsilon)^{-3c\alpha} \le \frac{D(\mathbf{x})}{D(\mathbf{y})} \le (n/\epsilon)^{3c\alpha}.$$

We show now that for α-bounded MEM distributions, long terms are insignificant. First we prove a general Lemma.

Lemma 13. *Let $f(\mathbf{x})$ depend only on $x_{l+1}\cdots x_n$ and $D(\mathbf{x})$ be an α-bounded MEM distribution. Let v be a conjunction of ℓ literals, including either x_i or $\bar{x}_i$, for $i = 1, \ldots, l$. Then*

$$\frac{Pr_D(f)}{(1+e^{3\alpha})^l} \le Pr_D(f \wedge v) \le \frac{Pr_D(f)}{(1+e^{-3\alpha})^l}.$$

Proof: Divide the elements of $\{0, 1\}^n$ into disjoint subsets such that all the elements of a subset have the same value of $x_{l+1} \cdots x_n$. Note that $f()$ has the same value on all the 2^l elements of each subset S. Denote by S_T the set of subsets whose elements satisfy $f()$. Each subset $S \in S_T$ has a single element $\mathbf{x}_s$ satisfying v. For every $i = 0, \ldots, l$ there are exactly $\binom{l}{i}$ elements y_i in S that differ from x_s in exactly i positions (in $x_1 \cdots x_l$). By Lemma 11, the probability of such a y_i is at least $(e^{-3\alpha})^i D(\mathbf{x}_s)$ and at most $(e^{3\alpha})^i D(\mathbf{x}_s)$. Since the total probability of all the elements in S_T is $Pr_D(f)$, we have

$$Pr_D(f) = \sum_{S \in S_T} \sum_{y \in S} D(y) \geq \sum_{S \in S_T} \sum_{i=0}^{l} \binom{l}{i} (e^{-3\alpha})^i D(\mathbf{x}_s).$$

By the binomial theorem,

$$Pr_D(f) \geq (1 + e^{-3\alpha})^l \sum_{S \in S_T} D(\mathbf{x}_s) = (1 + e^{-3\alpha})^l Pr_D(f \wedge v).$$

The other direction is similar, and the lemma follows. □

Corollary 14. *Let $g(\mathbf{x})$ be a DNF and $D(\mathbf{x})$ be an α-bounded MEM distribution. Let m be a term of g with k literals, then $Pr_D(m = 1) \leq \frac{1}{(1+e^{-3\alpha})^k}$.*

We use Corollary 14 to prove that the learning algorithm can afford to ignore long terms.

Theorem 15. *Let $g(\mathbf{x})$ be a DNF of at most n terms, and $D(\mathbf{x})$ be an α-bounded MEM distribution. Denote by $g_{\alpha,\epsilon}$ the disjunction of all the terms of g of length at most $k = 2e^{3\alpha} \ln \frac{n}{\epsilon}$. Then $Error_D(g, g_{\alpha,\epsilon}) \overset{\text{def}}{=} Pr_D(g \neq g_{\alpha,\epsilon}) < \epsilon$.*

Proof: Let m be a term of length more than $k = 2e^{3\alpha} \ln \frac{n}{\epsilon}$. By Corollary 14, using the inequality $2 \leq (1 + \frac{1}{a})^a < e$ for $a \geq 1$, we have that $Pr_D(m = 1) < \frac{\epsilon}{n}$. Therefore,

$$Error_D(g, g_{\alpha,\epsilon}) = Pr_D(g = 1 \wedge g_{\alpha,\epsilon} = 0) \leq \sum_{m \in g, m \notin g_{\alpha,\epsilon}} Pr_D(m = 1) < \epsilon. \qquad \square$$

4.3. Learning using statistical queries

The existence of an exact statistical query oracle is not a realistic assumption. However, relaxing the requirement to estimate the expectation makes it possible to simulate such an oracle given examples of the target function. We show here how *Statistical Queries* (with an accuracy parameter μ) can be used to efficiently learn read-once DNF over α-bounded MEM distributions. We first concentrate on the special case where all the terms are *short*, of size at most $O(\ln n/\epsilon)$. Later we show how to extend this to the general case.

Corollary 6 states that the target function g depends on a variable x_i if and only if $\text{Infl}_D\ (x_i, g) \neq 0$, and that the sign of that *influence* determines the sign of the variable x_i in g. In the proof of Lemma 7 we described the queries needed to compute the value of $\Gamma_i \overset{\text{def}}{=} \text{Infl}_D\ (x_i, g) E_D[I_1(x_i)]$, an expression having the same sign as the required influence

$\text{Infl}_D\ (x_i, g)$. Finding a lower bound on the value $|\Gamma_i|$ is the main task of this section. Once we establish a lower bound Γ, it is sufficient to run the statistical queries of the algorithm (presented in Lemmas 7 and 9 and with appropriate accuracy parameters), such that the total error is less than $\Gamma/2$. This will allow us to recover the true sign of Γ_i from the values that STAT returns. The main concern is to obtain a bound Γ which is a polynomial in n and $1/\epsilon$, since this is a requirement for the statistical query algorithm to be efficient.

In the analysis we can assume that the target function g satisfies $E_D(\bar{g}) > \epsilon/2$ (otherwise we can estimate g by $h \equiv 1$). Recall the notation $g = \ell_i g_{i1} \vee g_{i0}$. We denote the term in which ℓ_i appears by m_i ($m_i \equiv \ell_i g_{i1}$).

Lemma 16. *Let D be an α-bounded MEM distribution and $g(\mathbf{x})$ be a read-once DNF. If $E_D(\bar{g}) > \frac{\epsilon}{2}$ and m_i is a term of size at most $l \le c \ln \frac{n}{\epsilon}$ then $E_D(m_i \wedge \bar{g}_{i0}) > \frac{\epsilon}{2}(\frac{\epsilon}{n})^{c(3\alpha+1)}$.*

Proof: First we bound $E_D(\bar{g}_{i0}) > E_D(\bar{g}_{i0} \wedge \bar{m}_i) = E_D(\bar{g}) \ge \epsilon/2$. We can use Lemma 13, since m_i and g_{i0} depend on different variables. By Lemma 13, using the inequality $(1 + e^a)^b < e^{b(a+1)}$, we have

$$E_D(m_i \wedge \bar{g}_{i0}) \ge \frac{\epsilon/2}{(1+e^{3\alpha})^l} > \frac{\epsilon}{2}\left(\frac{\epsilon}{n}\right)^{c(3\alpha+1)},$$

and the lemma follows. □

From the above lemma we can deduce the following corollary on the value of Γ.

Corollary 17. *Let D be an α-bounded MEM distribution and let $g(\mathbf{x})$ be a read-once DNF such that $E_D(\bar{g}) > \frac{\epsilon}{2}$. If ℓ_i appears in a term of size at most $c \ln \frac{n}{\epsilon}$ then*

$$|\Gamma_i| \overset{\text{def}}{=} |\text{Infl}_D\ (x_i, g)\ E_D[I_1(x_i)]| > (\epsilon/2)(\epsilon/n)^{c(3\alpha+1)} \overset{\text{def}}{=} \Gamma.$$

Proof: We assume $\ell_i = x_i$ (the other case is similar). From the definition of Γ_i,

$$\begin{aligned}\Gamma_i &= Pr_D(x_i = 1)(E_{D_{i\leftarrow 1}}[g_{i\leftarrow 1}] - E_{D_{i\leftarrow 1}}[g_{i\leftarrow 0}])\\ &= Pr_D(x_i = 1)E_{D_{i\leftarrow 1}}[g_{i\leftarrow 1} - g_{i\leftarrow 0}].\end{aligned}$$

Since ℓ_i is positive and g is read-once, using the notation $g = x_i g_{i1} \vee g_{i0}$ we have

$$\begin{aligned}\Gamma_i &= Pr_D(x_i = 1)E_{D_{i\leftarrow 1}}[g_{i0} \vee g_{i1} - g_{i0}]\\ &= Pr_D(x_i = 1)E_{D_{i\leftarrow 1}}[\bar{g}_{i0} \wedge g_{i1}]\\ &= E_D[\bar{g}_{i0} \wedge g_{i1} \wedge x_i],\end{aligned}$$

where the last equality follows from Lemma 1. Finally,

$$\Gamma_i = E_D[\bar{g}_{i0} \wedge g_{i1} \wedge x_i] = E_D[\bar{g}_{i0} \wedge m_i] > (\epsilon/2)(\epsilon/n)^{c(3\alpha+1)},$$

where the last inequality follows from Lemma 16. □

We can now use the statistical queries as described in the proof of Lemma 7 to identify the significant variables.

Once the significant variables are found we have to test if pairs of literals belong to the same term. We can derive a similar bound on the true expectations in this case as well. (This is because if D is α-bounded then $D_{i\leftarrow b}$ is 2α-bounded, and if the terms of g are of size at most $c \log \frac{n}{\epsilon}$ so are the terms of $g_{i\leftarrow b}$.) This establishes the learnability result for read-once DNF with *short* terms.

Theorem 18. *The concept class read-once DNF with terms of size $O(\log n/\epsilon)$ can be efficiently learned with Statistical Queries over a given α-bounded MEM distribution.*

In what follows we describe the modifications needed to handle an arbitrary read-once DNF. The main concern is that the presence of long terms will hurt the learning process. First note that by using Corollary 17 we are assured of recovering all the terms of size at most $c \ln \frac{n}{\epsilon}$. By definition, this is $g_{\alpha,\epsilon}$. Since, by Theorem 15, longer terms are insignificant, the short terms are sufficient for a good approximation of the target g.

The problem is that the algorithm might construct short terms that are not terms of the target function but part of longer terms. Such terms increase the error when they are 1 and the target function is 0. Of special concern are those terms that increase the error significantly, called harmful terms.

The idea is to identify and delete the harmful terms. For a given term m, this can be done by making a statistical query on the probability that m is 1 and the target function is 0. Once we delete the harmful terms we have a good approximation of the target read-once DNF. This will establish the following theorem.

Theorem 19. *The concept class read-once DNF can be efficiently learned with Statistical Queries over a given α-bounded MEM distribution.*

Before we prove the theorem, examine the implementation of the algorithm in figures 1 and 2. The input parameters of the main procedure **LearnMaxEnt** are the allowed hypothesis error of the SQ model ϵ and the parameters $\mathbf{M}$, $\mathbf{w}$, and α of the α-bounded MEM distribution D. **LearnMaxEnt** first tests whether the target read-once DNF g can be trivially approximated by a constant. If that is not the case then the procedure **Literal** is used to construct a set $\mathcal{S}$ containing all the significant literals.

Literal receives as input the index of the literal to be tested, an accuracy parameter Γ which is the lower bound on the influence of a variable appearing in a short term of g (recall that if x_i does not appear in g then its influence is 0), and the parameters $\mathbf{M}$ and $\mathbf{w}$ of the α-bounded MEM distribution D. **Literal** makes Statistical Queries in order to estimate the influence $\mathrm{Infl}_D(x_i, g)$. By a proper choice of the accuracy parameter of the queries, the true sign of the influence is recovered.

Once the set $\mathcal{S}$ is computed, **LearnMaxEnt** reconstructs the terms of g. Initially, every literal ℓ_i is a term Term_i. The procedure **SameTerm** is used to test whether a pair of literals ℓ_i and ℓ_j (where $i < j$) belongs to the same term of the target g. (For simplicity, the procedure **SameTerm** is presented for the case where both literals ℓ_i and ℓ_j are positive. The other cases can be treated similarly.)

```
/* Find if $g$ depends on $x_i$ or $\overline{x_i}$. */
**Literal**($i, \Gamma, \mathbf{M}, \mathbf{w}$)
Let $z = 2\mathbf{M}_i^i,\ r = m_{ii} + w_i$.
Estimate $\gamma_1 = E_D[g(\mathbf{x}) \wedge I_1(x_i)]$ with accuracy $\frac{\Gamma}{4}$
Estimate $\gamma_0 = E_D[e^{-\mathbf{z}\mathbf{x}^i - r} g(\mathbf{x}) \wedge I_0(x_i)]$ with accuracy $\frac{\Gamma}{4}$
IF $|\gamma_1 - \gamma_0| < \frac{\Gamma}{2}$ RETURN 0
ELSE IF $\gamma_1 > \gamma_0$ RETURN $x_i$
ELSE RETURN $\overline{x_i}$

/* Test is the literals $\ell_i$ and $\ell_j$ appear in the same term. */
**SameTerm**($\ell_i, \ell_j, \Gamma, \mathbf{M}, \mathbf{w}$)
Let $z = 2\mathbf{M}_j^{ij}, r = m_{jj} + w_j$.
Estimate $\gamma_1 = E_D[g(\mathbf{x}) \wedge I_0(x_i) \wedge I_1(x_j)]$ with accuracy $\frac{\Gamma}{4}$
Estimate $\gamma_0 = E_D[e^{-\mathbf{z}\mathbf{x}^{ij} - r} g(\mathbf{x}) \wedge I_0(x_i) \wedge I_0(x_j)]$ with accuracy $\frac{\Gamma}{4}$
IF $|\gamma_1 - \gamma_0| < \frac{\Gamma}{2}$ RETURN True
ELSE RETURN False
```

Figure 1. The procedures used in the algorithm for learning read-once DNF over α-bounded MEM distribution.

SameTerm receives as input the two literals to be tested, an accuracy parameter Γ, and the parameters $\mathbf{M}$ and $\mathbf{w}$ of the α-bounded MEM distribution D. As in **Literal, SameTerm** makes Statistical Queries in order to estimate the influence $\mathrm{Infl}_{D_{i\leftarrow 0}}(x_j, g_{i\leftarrow 0})$. A nonzero influence indicates that the literals are not in the same term. Again, by a proper choice of the accuracy parameter of the queries, the true sign of the influence is recovered.

If ℓ_i and ℓ_j belong to the same term of the target g, then **LearnMaxEnt** deletes ℓ_j from $\mathcal{S}$ and adds it to the term Term_i. Eventually, all the short terms of g are reconstructed.

The final step is to identify and delete the harmful terms (terms reconstructed by **LearnMaxEnt** that are part of longer terms of the target g). This is done with a statistical query, estimating the harmfulness of every term.

Proof of Theorem 19: We will show that the algorithm **LearnMaxEnt**, described in figures 1 and 2, efficiently learns the read-once DNF.

Our first claim is that **Literal** returns the true sign of x_i. By (1) we have,

$$|\gamma_1 - \gamma_0| = \left|E_D[g(\mathbf{x}) \wedge I_1(x_i)] - E_D\left[e^{-\mathbf{z}\mathbf{x}^i - r} g(\mathbf{x}) \wedge I_0(x_i)\right]\right|$$
$$= |\mathrm{Infl}_D(x_i, g) Pr_D(x_i = 1)| \overset{\mathrm{def}}{=} |\Gamma_i|.$$

By Corollary 17, if x_i appears in g then $|\Gamma_i| > \Gamma$. Since γ_0 and γ_1 are each estimated with accuracy $\frac{\Gamma}{4}$ their difference has an error of at most $\frac{\Gamma}{2}$, and we are assured of recovering the true sign.

Our second claim is that **SameTerm** returns True if and only if ℓ_i and ℓ_j belong to the same term of the target read-once DNF g. Again we have,

$$|\gamma_1 - \gamma_0| = \left|E_D[g(\mathbf{x}) \wedge I_0(x_i) \wedge I_1(x_j)] - E_D\left[e^{-\mathbf{z}\mathbf{x}^{ij} - r} g(\mathbf{x}) \wedge I_0(x_i) \wedge I_0(x_j)\right]\right|$$
$$= |\mathrm{Infl}_{D_{i\leftarrow 0}}(x_j, g_{i\leftarrow 0}) Pr_{D_{i\leftarrow 0}}(x_j = 1)|$$

```
/* Find an $\epsilon$-approximation to $g$
with respect to an $\alpha$-bounded distribution. */
**LearnMaxEnt**($\epsilon$, **M**, **w**, $\alpha$)
Estimate $p \equiv Prob[g = 0]$.
IF $p < \epsilon/2$ OR $p > 1 - \epsilon/2$ THEN DONE.
Let $\Gamma = (\epsilon/2)(\epsilon/n)^{e^{3\alpha}(3\alpha+1)}$.
Let $\mathcal{S} = \emptyset$.
/* Find the relevant literals. */
FOR each variable $x_i$,
    Term$_i = \emptyset$.
    $\ell_i$ =**Literal**($i$, $\Gamma$, **M**, **w**)
    IF $\ell_i \neq 0$
        $\mathcal{S} = \mathcal{S} \cup \{\ell_i\}$.
        Term$_i = \{\ell_i\}$.
/* Construct candidates for terms. */
FOR each $i = 1, \ldots, n$   s.t   $\ell_i \in \mathcal{S}$
    FOR each $j > i$   s.t   $\ell_j \in \mathcal{S}$
    **IF SameTerm**($\ell_i, \ell_j, \Gamma$, **M**, **w**)
        **AND SameTerm**($\ell_j, \ell_i, \Gamma$, **M**, **w**)
        Term$_i$ = Term$_i \cup \{\ell_j\}$.
        Term$_j = \emptyset$.
        $\mathcal{S} = \mathcal{S} - \{\ell_j\}$.
/* Delete inaccurate candidate terms.*/
FOR each $i = 1, \ldots, n$ s.t Term$_i \neq \emptyset$
    Estimate $t_i = E_D(\text{Term}_i \wedge \overline{g}(\mathbf{X}))$ with accuracy $\frac{\epsilon}{4n}$
    IF $t_i > \frac{\epsilon}{4n}$ THEN Term$_i = \emptyset$.
RETURN $\bigvee_i \bigwedge_{\ell \in \text{Term}_i} \ell$.
```

Figure 2. Algorithm for learning read-once DNF over α-bounded MEM distribution.

By Corollary 17, if x_j appears in $g_{i\leftarrow 0}$ then $|\text{Infl}_{D_{i\leftarrow 0}}(x_j, g_{i\leftarrow 0})Pr_{D_{i\leftarrow 0}}(x_j = 1)| > \Gamma$. Since γ_0 and γ_1 are each estimated with accuracy $\frac{\Gamma}{4}$ their difference has an error of at most $\frac{\Gamma}{2}$, and we are assured of the correct answer.

We conclude that at the end of the first loop of **LearnMaxEnt** the set $\mathcal{S}$ contains all the literals of short (size $O(\ln \frac{n}{\epsilon})$) terms of the target g (but may include some more literals from other terms).

We now claim that all short terms of g are reconstructed correctly. First note that **SameTerm** tests whether the literal ℓ_j remains significant when ℓ_i is set to 0 (it is actually a significance test, similar to the test done by **Literal**). Both literals belong to the same term of g only if the conditional significance vanishes. Our main concern is the literals in $\mathcal{S}$ from long terms of g. The double call to **SameTerm** in the second loop of **LearnMaxEnt** ensures that the algorithm will not combine a literal ℓ_s from a short term of g and a literal ℓ_l from a long term of g into the same term. (By Corollary 17, ℓ_s will remain significant even when ℓ_l is set to 0.) It follows that at the end of the second loop of **LearnMaxEnt** all short terms of g are reconstructed correctly ($g_{\alpha,\epsilon}$). However, there might be more reconstructed terms (whose literals are from long terms of g), some of which may be harmful.

Our next claim is that the last loop of **LearnMaxEnt** removes all the harmful reconstructed terms. Since all the terms of $g_{\alpha,\epsilon}$ were reconstructed correctly, a term m can be harmful only when m is 1 and g is 0 (we define the damage of m to be $Pr_D(m \wedge \bar{g})$—the increase in the error caused by m). All the terms with damage more than $\frac{\epsilon}{2n}$ are deleted by **LearnMaxEnt** (the damage is estimated with accuracy $\frac{\epsilon}{4n}$, so if a term is not deleted its damage is at most $\frac{\epsilon}{4n} + \frac{\epsilon}{4n} = \frac{\epsilon}{2n}$). Since there are at most n terms (g is read-once) the total increase in error due to harmful terms is at most $\frac{\epsilon}{2}$.

All the queries are done with accuracy bounded by an inverse of a polynomial in m and $\frac{1}{\epsilon}$, implying the efficiency of the algorithm. □

5. Learning monotone $k\mu$ DNF

In the previous sections we showed how a result about the learnability of read-once DNF over product distributions can be extended to α-bounded MEM distributions. We now continue and describe how another result can be extended in a similar way.

In Hancock and Mansour (1991) it was shown how to efficiently learn over product distributions the class of monotone DNF where each variable appears in no more than k terms (denoted $k\mu$-DNF). The learning algorithm did not reconstruct the DNF, but instead, efficiently identified all the minimal blocking sets for each variable (a set of variables ρ blocks the variable x_i over a function f iff $f_{\rho\leftarrow 0}$ does not depend on x_i, i.e., iff Infl $(x_i, f_{\rho\leftarrow 0}) = 0$). It was shown that knowing an approximation of the set of minimal blocking sets (denoted ϵ-blocking sets) for each variable enables a good approximation of the $k\mu$-DNF.

5.1. Computing using blocking sets

We first present (as in Hancock and Mansour (1991)) a function that computes a monotone function $f()$ given its blocking sets. The function *compute*() (see figure 3) receives as input a vector of variables $x_1, \ldots, x_n$ and for each variable a set of minimal blocking sets $\mathcal{B}_1, \ldots \mathcal{B}_n$where $\mathcal{B}_i \subset 2^{\{1,\ldots,n\}}$. The computed value is $f(x_1, \ldots, x_n)$as stated by the following Lemma.

```
FUNCTION $compute(x_1, \ldots, x_n, \mathcal{B}_1, \ldots, \mathcal{B}_n)$
IF $\exists x_i = 1$ such that
    $\forall A \in \mathcal{B}_i$ there exists $j \in A$,
        such that $x_j = 1$.
THEN RETURN 1
ELSE RETURN 0.
```

Figure 3. Computing the value of a monotone function using blocking sets.

Lemma 20 (Hancock & Mansour, 1991). *Let $f(x_1, \ldots, x_n)$ be a monotone function, and $\mathcal{B}_i$, $1 \leq i \leq n$, be a maximal collection of minimal blocking sets of the variable x_i. Then,*

$$compute(x_1, \ldots, x_n, \mathcal{B}_1, \ldots, \mathcal{B}_n) = f(x_1, \ldots, x_n).$$

5.2. *Identifying blocking sets*

By the definition, blocking sets ρ of a variable x_i over the $k\mu$-DNF g can be identified by estimating the influence of the variable x_i on $g_{\rho\leftarrow 0}$. In the next section we will introduce the definition of ϵ-blocking sets ρ over α-bounded MEM distributions. By that definition, a lower bound on the influence of a variable x_i on $g_{\rho\leftarrow 0}$ is polynomial in $\frac{1}{n}$ and ϵ. This will allow us to efficiently identify ϵ-blocking sets. We will prove that over α-bounded MEM distributions using ϵ-blocking sets in *compute* results in a sufficient approximation of the $k\mu$-DNF.

This generalizes the result of Hancock and Mansour (1991) from uniform and product distributions to α-bounded MEM distributions. We start with a generalization of Lemma 16.

Lemma 21. *Let D be an α-bounded MEM distribution and*

$$g = T_0 \vee T_1 \vee T_2 \vee \cdots \vee T_s \vee R_1 \vee R_2 \vee \cdots \vee R_q$$

be a monotone $k\mu$-DNF such that T_0 is a term of size l and $T_1 \cdots T_s$ are all the terms sharing literals with T_0. If $Pr_D(\bar{g}) > \frac{\epsilon}{2}$ then

$$Pr_D(T_0 \wedge \bar{T}_1 \wedge \cdots \wedge \bar{T}_s \wedge \bar{R}_1 \wedge \cdots \wedge \bar{R}_q) > \frac{\epsilon/2}{(1+e^{3\alpha})^{kl}}.$$

Proof: Without loss of generality we can assume that $T_0 = x_1 \wedge \cdots \wedge x_l$. By the monotonicity of g, every term $T_i, i = 1, \ldots, s$, has at least one variable not appearing in T_0. Denote by $S = \{x_{j_1} \cdots x_{j_s}\}$ a set containing one variable from every $T_i, i = 1, \ldots, s$, and none from T_0. Since g is $k\mu$-DNF we have $s \leq (k-1)l$. Denote by $R_{a_1} \cdots R_{a_t}$ the terms among $R_1 \cdots R_q$ with no variable in S. First we bound

$$Pr_D(\bar{R}_{a_1} \wedge \cdots \wedge \bar{R}_{a_t}) \geq Pr_D(\bar{g}) > \epsilon/2.$$

Now, using the above definitions we write

$$\begin{aligned} &Pr_D(T_0 \wedge \bar{T}_1 \wedge \cdots \wedge \bar{T}_s \wedge \bar{R}_1 \wedge \cdots \wedge \bar{R}_q) \\ &\quad \geq Pr_D(T_0 \wedge \bar{x}_{j_1} \wedge \cdots \wedge \bar{x}_{j_s} \wedge \bar{R}_1 \wedge \cdots \wedge \bar{R}_q) \\ &\quad = Pr_D(T_0 \wedge \bar{x}_{j_1} \wedge \cdots \wedge \bar{x}_{j_s} \wedge \bar{R}_{a_1} \wedge \cdots \wedge \bar{R}_{a_t}). \end{aligned}$$

We can now use Lemma 13. This is because $T_0 \wedge \bar{x}_{j_1} \wedge \cdots \wedge \bar{x}_{j_s}$ imposes no more than $l + s \leq kl$ constraints on variables not belonging to any of the terms $R_{a_1} \cdots R_{a_t}$. We further bound

$$Pr_D\big(T_0 \wedge \bar{x}_{j_1} \wedge \cdots \wedge \bar{x}_{j_s} \wedge \bar{R}_{a_1} \wedge \cdots \wedge \bar{R}_{a_t}\big) > \frac{\epsilon/2}{(1+e^{3\alpha})^{kl}} ,$$

which proves the lemma. □

Note that for the case where T_0 is a term of size at most $l = c \ln \frac{n}{\epsilon}$, the lemma generalizes Lemma 16 for monotone $k\mu$-DNF. (The resulting lower bound becomes $\frac{\epsilon}{2}(\frac{\epsilon}{n})^{kc(3\alpha+1)}$.)

Lemma 22. *Let D be an α-bounded MEM distribution and g be a monotone $k\mu$ DNF such that $Pr_D(\bar{g}) > \frac{\epsilon}{2}$. If ρ is not a blocking set for x_i then*

$$\mathrm{Infl}_{D_{\rho\leftarrow 0}}(x_i, g_{\rho\leftarrow 0}) \geq \frac{\epsilon/2}{(1+e^{3\alpha})^{kl}} ,$$

where l is the size of the smallest term that includes x_i in $g_{\rho\leftarrow 0}$.

Proof: Let T_0 be a term of size l that includes x_i and no variable in ρ. We can write

$$\begin{aligned} g &= T_0 \vee T_1 \vee \cdots \vee T_s \vee R_1 \vee \cdots \vee R_q \qquad \text{and} \\ g_{\rho\leftarrow 0} &= T_0 \vee T_{j_1} \vee \cdots \vee T_{j_v} \vee R_{a_1} \vee \cdots \vee R_{a_t}, \end{aligned}$$

where $T_1 \cdots T_s$ are the terms sharing variables with T_0. Without loss of generality we can assume that $T_{j_1} \cdots T_{j_u}$ are the terms of $g_{\rho\leftarrow 0}$ in which x_i appears. We have

$$\begin{aligned} &\mathrm{Infl}_{D_{\rho\leftarrow 0}}(x_i, g_{\rho\leftarrow 0}) \\ &\overset{\text{def}}{=} Pr_{D_{\rho\leftarrow 0,i\leftarrow 1}}(g_{\rho\leftarrow 0,i\leftarrow 1}) - Pr_{D_{\rho\leftarrow 0,i\leftarrow 1}}(g_{\rho\leftarrow 0,i\leftarrow 0}) \\ &= Pr_{D_{\rho\leftarrow 0,i\leftarrow 1}}\big(\big(T_0 \vee T_{j_1} \vee \cdots \vee T_{j_u}\big) \wedge \big(\bar{T}_{j_{u+1}} \wedge \cdots \wedge \bar{T}_{j_v} \wedge \bar{R}_{a_1} \wedge \cdots \wedge \bar{R}_{a_t}\big)\big) \\ &\geq Pr_{D_{\rho\leftarrow 0}}\big(T_0 \wedge \bar{T}_{j_1} \wedge \cdots \wedge \bar{T}_{j_v} \wedge \bar{R}_{a_1} \wedge \cdots \wedge \bar{R}_{a_t}\big). \end{aligned}$$

Since $D_{\rho\leftarrow 0,i\leftarrow 1}$ is an α-bounded MEM and $Pr(\bar{g}) > \epsilon/2$, we can use Lemma 21; hence,

$$\begin{aligned} \mathrm{Infl}_{D_{\rho\leftarrow 0}}(x_i, g_{\rho\leftarrow 0}) &\geq Pr_{D_{\rho\leftarrow 0}}\big(T_0 \wedge \bar{T}_{j_1} \wedge \cdots \wedge \bar{T}_{j_v} \wedge \bar{R}_{a_1} \wedge \cdots \wedge \bar{R}_{a_t}\big) \\ &\geq \frac{\epsilon/2}{(1+e^{3\alpha})^{kl}} , \end{aligned}$$

which completes the proof. □

5.3. ϵ-blocking sets

We say that a set ρ of variables ϵ-blocks the variable x_i of a monotone function f over α-bounded MEM distribution D iff

$$\mathrm{Infl}_{D_{\rho\leftarrow 0}}(x_i, f_{\rho\leftarrow 0}) < \frac{\epsilon}{2}(1+e^{3\alpha})^{-k\lambda} \quad \text{where } \lambda = 2 \ln \frac{2kn}{\epsilon}.$$

Note that every blocking set is an ϵ-blocking set, and by Lemma 22, for a $k\mu$ DNF, if ρ ϵ-blocks x_i, it contains a variable from every term of length $\leq \lambda$ that contains x_i.

Lemma 23. *Let D be an α-bounded MEM distribution, g be a $k\mu$ DNF formula, $\mathcal{B}_i$ the set of all ϵ-blocking sets for x_i, and $\mathcal{B} = \mathcal{B}_1, \ldots, \mathcal{B}_n$. Then,*

$$Pr_D(compute(x, \mathcal{B}) \neq g(x)) \leq \epsilon/2.$$

Proof: As in Hancock and Mansour (1991), since every blocking set is an ϵ-blocking set, $\mathcal{B}_i$ includes all the blocking sets of variable x_i. This implies that if *compute*$(x, \mathcal{B}) = 1$ then $g(x) = 1$. Therefore the only errors are when *compute*$(x, \mathcal{B}) = 0$ while the input x satisfies some terms of g (i.e., $g(x) = 1$).

As noted before, an ϵ-blocking set for x_i must contain a variable from every term containing x_i, whose length is at most λ. Hence an example satisfying such a term must have a variable set to 1 in each ϵ-blocking set for x, and will therefore be correctly evaluated by *compute*.

By Corollary 14, the contribution of all g's terms with length more than λ is at most $kn(1 + e^{3\alpha})^{-\lambda} \leq \epsilon/2$, which completes the proof. □

5.4. *The algorithm*

The resulting algorithm (see figure 4) for identifying ϵ-blocking sets is similar to the one described in Hancock and Mansour (1991).

A set ρ with at most k variables is added to $\mathcal{B}_i$ only if $\text{Infl}_{D_{\rho\leftarrow 0}}(x_i, g_{\rho\leftarrow 0}) < \Gamma$, i.e., only if ρ is an ϵ-blocking set. The estimation of the influence is based on an analysis similar to the one described in the proof of Lemma 7. We conclude with the resulting theorem.

```
**FindBlockingSets**$(k, \epsilon, \mathbf{M}, \mathbf{w}, \alpha)$
Estimate $p \equiv Pr_D[g = 0]$.
IF $p < \epsilon/2$ THEN DONE.
Let $\lambda = 2 \ln \frac{2kn}{\epsilon}$.
LET $\Gamma = p(1 + e^{3\alpha})^{-k\lambda}$.
FOR each variable $x_i$,
    Let $\mathcal{B}_i = \emptyset$.
    FOR each set $\rho$ of at most $k$ variables,
        Estimate $\alpha_{\rho,i} \equiv E_D[g(\mathbf{x}) I_0(\rho) I_1(x_i)]$ with accuracy $\frac{\Gamma}{4}$.
        Estimate $\beta_{\rho,i} \equiv E_D[e^{-\mathbf{z}_\rho \mathbf{x}^\rho - r_\rho} g(\mathbf{x}) I_0(\rho) I_0(x_i)]$
                with accuracy $\frac{\Gamma}{4}$.
        IF $|\alpha_{\rho,i} - \beta_{\rho,i}| < \frac{\Gamma}{2}$
          THEN $\mathcal{B}_i = \mathcal{B}_i \cup \{\rho\}$.
```

Figure 4. Identifying ϵ-blocking sets over an α-bounded MEM distribution.

Theorem 24. *Let g be a $k\mu$-DNF. Let D be an α-bounded MEM distribution. Then $g(\mathbf{x})$ can be efficiently approximated over D using statistical queries.*

Note

1. We denote the boolean expression $a \wedge b$ by ab.

References

Angluin, D. & Kharitonov, M. (1991). When won't membership queries help? In *Proceedings of the 23rd Annual ACM Symposium Theory Computing* (pp. 444–454). New York: ACM.

Aslam, J. A. & Decatur, S. E. (1994). Improved nose-tolerant learning and generalized statistical queries. Tech. rep. TR-17-94, Harvard University.

Benedek, G. M. & Itai, A. (1991). Learnability with respect to fixed distributions. *Theoretical Computer Science*, *86*(2), 377–390.

Cover, T. M. & Thomas, J. A. (1991). *Elements of Information Theory*. New York, NY: John Wiley and Sons Inc.

Hancock, T. & Mansour, Y. (1991). Learning monotone $k\mu$ dnf formulas on product distributions. In *Proceedings of the 2nd Annual Workshop on Computational Learning Theory* (pp. 179–183). San Mateo, CA: Morgan Kaufmann.

Haussler, D., Kearns, M., Littlestone, N., & Warmuth, M. K. (1991). Equivalence of models for polynominal learnability. *Information and Computation*, *95*(2), 129–161.

Kearns, M. (1993). Efficient noise-tolerant learning from statistical queries. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing* (pp. 392–401).

Kearns, M., Li, M., & Valiant, L. G. (1994). Learning boolean formulas. *Journal of the ACM, 41*(6), 1298–1328.

Kearns, M. & Valiant, L. G. (1989). Cryptographic limitations on learning boolean formulae and finite automata. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing* (pp. 433–444).

Kilian, J. & Naor, M. (1995). On the complexity of statistical reasoning. In *Proceedings of the 3rd Israel Symposium on Theory of Computing and Systems* (pp. 209–217).

Koller, D. & Megiddo, N. (1993). Constructing small sample spaces satisfying given constraints. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing* (pp. 268–277).

Natarajan, B. K. (1987). On learning boolean functions. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing* (pp. 296–304). New York.

Papoulis, A. (1991). *Probability, Random Variables, and Stochastic Processes* (3rd edn.), ch. 15. New York, NY: McGraw-Hill.

Pitt, L. & Valiant, L. G. (1988). Computational limitations on learning from examples. *Journal of the ACM, 35*(4), 965–984.

Smith, R. C. & Grandy, W. (1985). *Maximum-Entropy and Bayesian Methods in Inverse Problems*. Dordrecht: D. Reidel Publishing Company.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM, 27*(11), 1134–1142.

Received June 1, 1998
Revised July 28, 2000
Accepted July 28, 2000
Final Manuscript August 20, 2001