Weakly Learning DNF and Characterizing Statistical Query Learning Using Fourier Analysis

(DRAFT — PLEASE DO NOT DISTRIBUTE)

Avrim Blum
Carnegie Mellon U.

Merrick Furst
Carnegie Mellon U.

Jeffrey Jackson
Carnegie Mellon U.

Michael Kearns
AT&T Bell Laboratories

Yishay Mansour
Tel-Aviv U.

Steven Rudich
Carnegie Mellon U.

November 1993

Abstract

We present new results on the well-studied problem of learning DNF expressions. We prove that an algorithm due to Kushilevitz and Mansour [13] can be used to weakly learn DNF formulas with membership queries with respect to the uniform distribution. This is the first positive result known for learning general DNF in polynomial time in a nontrivial model. Our results should be contrasted with those of Kharitonov [12], who proved that AC^0 is not efficiently learnable in this model based on cryptographic assumptions. We also present efficient learning algorithms in various models for the read-k and SAT-k subclasses of DNF.

We then turn our attention to the recently introduced *statistical query* model of learning [9]. This model is a restricted version of the popular Probably Approximately Correct (PAC) model, and practically every PAC learning algorithm falls into the statistical query model [9]. We prove that DNF and decision trees are not even weakly learnable in polynomial time in this model. This result is information-theoretic and therefore does not rely on any unproven assumptions, and demonstrates that no straightforward modification of the existing algorithms for learning various restricted forms of DNF and decision trees will solve the general problem. These lower bounds are a corollary of a more general characterization of the complexity of statistical query learning in terms of the number of uncorrelated functions in the concept class.

The underlying tool for all of our results is the Fourier analysis of the concept class to be learned.

1 Introduction

We present new results on the well-studied problem of learning DNF expressions. The problem of efficiently learning DNF formulas in any nontrivial model of learning has been of central interest in computational learning theory since the seminal paper of Valiant [18] introducing the popular Probably Approximately Correct (PAC) learning model. Despite the importance of this problem, to date no polynomial time algorithm for learning unrestricted DNF has been discovered.

In this paper we prove that an algorithm due to Kushilevitz and Mansour [13] can be used to weakly learn DNF with respect to the uniform distribution using membership queries. This is the first positive result for learning general DNF in a nontrivial model of learning. In particular, this result provides a contrast between DNF formulas and more general AC^0 circuits, which Kharitonov [12] proved were not learnable in this model based on cryptographic assumptions. (In fact, at the time of Kharitonov's result, it appeared possible that his results would soon be extended to DNF; our result shows otherwise.)

Due to the lack of positive results for unrestricted DNF, various restricted DNF classes have attracted considerable attention [4, 2, 8, 3, 1, 5, 14, 6]. We extend these results. In particular, it is known that the class of read-k DNF (DNF in which every variable appears at most k times) is learnable in polynomial time using membership queries for $k \leq 2$ [2, 8] but is as hard to learn in the distribution-free PAC model as unrestricted DNF for $k \geq 3$ [8]. Also, Aizenstein and Pitt [3] have shown that read-k SAT-ℓ DNF (DNF which are both read-k and such that at most ℓ terms are satisfied by any input) can be learned in the distribution-free PAC model with membership queries. We show that, with respect to the uniform distribution and using membership queries, read-k DNF is polynomial-time learnable with an accuracy that is a constant depending on k. We also prove that SAT-k DNF is strongly learnable in time exponential in k (but otherwise polynomial), and that SAT-k $\log(n)$-DNF is learnable exactly with queries in the same time bound.

Actually, they show exact with mem+eq query learning.

We then turn our attention to the recently introduced *statistical query* model of learning [9]. This is a restricted version of the PAC model in which the learning algorithm does not actually receive labeled examples of the unknown target function drawn with respect to a distribution. Instead, in this model the learner may specify any *property* of labeled examples, and obtain accurate estimates of the probability that a random example will possess the property. An important feature of this model is that any class efficiently learnable from statistical queries is efficiently learnable in the PAC model with an arbitrarily high rate of classification noise [9]. Furthermore, it has been demonstrated [9] that practically every class known to be efficiently learnable in the PAC model (either in the distribution-free sense, or with respect to special distributions) is also learnable in the more restricted statistical query model. In other words, PAC algorithms almost always learn by estimating probabilities. A notable exception to this is the class of parity functions, which is known to be efficiently learnable in the PAC model but is not efficiently learnable in the statistical query model [9].

We provide a general characterization of the number of statistical queries required for learning that is applicable to any concept class with respect to any distribution. This characterization proves that if a class contains a superpolynomial number of nearly uncorrelated functions with respect to a distribution, then a superpolynomial number of statistical queries are required for learning. An immediate application of this result is that DNF and decision trees are not even weakly learnable in polynomial time in this model. This result does not rely on any unproven assumptions, and demonstrates that no straightforward modification of the existing algorithms for learning various restricted forms of DNF and decision trees will solve the general problem.

All of our results rely heavily on the Fourier representation of functions [15, 13, 17], demonstrating once again the utility of these tools in computational learning theory.

2 Definitions and Notation

2.1 Learning Models

A *concept* is a boolean function on an *instance space* X, and for convenience we define boolean functions to have outputs in $\{+1, -1\}$. A *concept class* $\mathcal{F}$ is a set of concepts. An *instance* $\vec{x}$ is an element of the instance space (here $\{0,1\}^n$), and we use x_i to denote the ith bit of $\vec{x}$. We generally use f to denote the target concept.

We say that a (possibly randomized) function g is an ϵ-*approximator* of f if $\mathbf{Pr}[g = f] \geq 1 - \epsilon$, where the probability is taken over the uniform distribution on the instance space and over the random choices of g.

A *membership query* is a query to an oracle for f for the value of f on a desired instance. If there is an algorithm $\mathcal{A}$ using membership queries such that for any positive ϵ and δ and any target $f \in \mathcal{F}$, with probability at least $1 - \delta$ algorithm $\mathcal{A}$ produces an ϵ-approximation for f in time polynomial in n, the size s of f, $1/\epsilon$, and $1/\delta$, then $\mathcal{F}$ is *(strongly) learnable using queries with respect to the uniform distribution*. The *size* of a concept f is a measure of the number of bits in the smallest representation of f; throughout this paper we will use the number of terms in the smallest DNF representation of f as the size s of f. The parameters ϵ and δ above are called the *accuracy* and *confidence* of the approximation, respectively.

If there is a polynomial $p(n, s)$ and an algorithm $\mathcal{A}$ using membership queries such that for any positive δ and any target $f \in \mathcal{F}$, with probability at least $1 - \delta$ algorithm $\mathcal{A}$ produces a $1/2 - 1/p(n, s)$-approximation for f in time polynomial in n, s, and $1/\delta$, then $\mathcal{F}$ is *weakly learnable using queries with respect to the uniform distribution*.

Unlike the models of learning we have defined so far, in the statistical query learning model [9] the learner is not allowed to explicitly see labeled examples $(\vec{x}, f(\vec{x}))$ of the target function, but instead may only *estimate probabilities* involving labeled examples. We formalize this as follows: the learning algorithm is given access to a *statistics oracle*. A query to this oracle is an arbitrary function $g : \{0,1\}^n \times \{+1, -1\} \rightarrow \{+1, -1\}$ along with a *tolerance* τ. The oracle may respond with any value $\hat{\tau}$ satisfying $\mathbf{E}[g(\vec{x}, f(\vec{x}))] - \tau \leq \hat{\tau} \leq \mathbf{E}[g(\vec{x}, f(\vec{x}))] + \tau$. In general, we will examine statistical query learnability not just with respect to the uniform distribution but with respect to any distribution, in which case it is understood that the expectations above are taken with respect to the distribution in question, as is the quality of an approximator.

We say that the concept class $\mathcal{F}$ is *learnable from statistical queries* if there is a learning algorithm $\mathcal{A}$ such that for any positive ϵ and any target $f \in \mathcal{F}$, algorithm $\mathcal{A}$ produces an ϵ-approximation for f in time polynomial in n, the size of f, and $1/\epsilon$, and algorithm $\mathcal{A}$ only makes queries (g, τ) in which g can be evaluated in time polynomial in these same parameters and τ is inverse polynomial in these same parameters. The motivation for this notion of efficiency and for the statistical query model in general can be found in the paper of Kearns [9]; here it suffices to reiterate that almost every learning algorithm in the PAC model is already a statistical query algorithm, and that learnability in the statistical query model implies PAC learning with noise.

2.2 DNF Expressions

A DNF formula is a disjunction of terms, where each term is a conjunction of literals and a literal is either a variable or its negation. For a given DNF formula f we use s to denote the number of terms in f, T_i to represent the ith term in f (the ordering is arbitrary), and V_i to denote the set of variables in T_i. A DNF formula f is *k-DNF* if it has at most k literals in each term, is *read-k* if each variable appears at most k times, and is *SAT-k* if no instance satisfies more than k terms of f. We assume for convenience that the `true` output value of a DNF f is represented by $+1$ and the `false` value by -1.

2.3 The Fourier Transform

For each bit vector $\vec{z} \in \{0,1\}^n$ we define the function $\chi_{\vec{z}} : \{0,1\}^n \rightarrow \{+1, -1\}$ as $\chi_{\vec{z}}(\vec{x}) = 1 - 2\left(\sum_{i=1}^{n} z_i x_i \bmod 2\right)$. That is, $\chi_{\vec{z}}(\vec{x})$ represents the parity of the set of bits in $\vec{x}$ selected by $\vec{z}$, with a parity of 0 represented by $+1$ and a parity of 1 represented by -1. Defined this way, the 2^n parity functions $\chi_{\vec{z}}$ have a number of useful properties which we will exploit repeatedly.

First, with inner product defined by[1] $\langle f, g\rangle = \mathbf{E}[fg]$ and norm by $\|f\| = \sqrt{\mathbf{E}[f^2]}$, $\{\chi_{\vec{z}}\}$ is an orthonormal basis for the vector space of real-valued functions on the Boolean cube $\mathbf{Z}_2^n$. That is, every function $f : \{0,1\}^n \to \mathbf{R}$ can be uniquely expressed as a linear combination of the parity functions:

$$f = \sum_{\vec{a}\in\{0,1\}^n} \hat{f}(\vec{a})\chi_{\vec{a}}.$$

We call the vector of coefficients $\hat{f}$ the *Fourier transform* of f. Because of the orthonormality of the parity functions, $\hat{f}(\vec{a}) = \mathbf{E}[f\chi_{\vec{a}}]$. Thus for boolean f, $\hat{f}(\vec{a})$ represents the correlation of f and $\chi_{\vec{a}}$. Also note that $\hat{f}(\vec{0}) = \mathbf{E}[f\chi_{\vec{0}}] = \mathbf{E}[f]$. We call $\hat{f}(\vec{0})$ the *constant Fourier coefficient* since $\chi_{\vec{0}}$ is the constant function $+1$. Finally, the Fourier transform is a linear operator. That is, if $h = cf + g$ for functions f, g and scalar c, then $\hat{h} = c\hat{f} + \hat{g}$.

Parseval's identity states that for every function f, $\mathbf{E}[f^2] = \sum_{\vec{a}} \hat{f}^2(\vec{a})$. For boolean f it follows that $\sum_{\vec{a}} \hat{f}^2(\vec{a}) = 1$, a fact we use frequently.

We at times use a subset A of the n variables of a function as the index of a parity or Fourier coefficient, with the following meaning: χ_A denotes the function $\chi_{\vec{a}}$ where $\vec{a}$ is the characteristic vector corresponding to A, and $\hat{f}(A)$ has a similar interpretation.

A *t-sparse function* is a function that has at most t non-zero Fourier coefficients. The *support* of a function f is the set $\{A \mid \hat{f}(A) \neq 0\}$.

3 Preliminaries

Our positive learnability results rely heavily on an algorithm of Kushilevitz and Mansour [13] (the *KM algorithm*) which finds, with high probability, close approximations to all of the large Fourier coefficients of a function f. The KM algorithm is allowed to make membership queries for f, but f is treated as a black box. Kushilevitz and Mansour have shown that given such approximate coefficients one can learn some important concept classes such as decision trees [13].

The main approach of our positive results is to show that DNF formulas have sufficiently large Fourier coefficients so that the KM algorithm can be usefully applied. We then use a general transformation given below that shows how to take a deterministic approximator g which is noticeably (that is, inverse polynomially) closer to f than the origin (regarding the functions as vectors), and produce a randomized approximator h such that $\mathbf{Pr}[f \neq h]$ is similarly better than 1/2.

We begin by stating as a lemma the known results about the KM algorithm which we will need. These and the other results of this section hold for any class of boolean functions, not just DNF.

Lemma 1 (Kushilevitz & Mansour) *For any boolean target function f, threshold θ, and $\epsilon, \delta > 0$, the KM algorithm, with probability at least $1 - \delta$, returns the nonzero Fourier coefficients of a function g with support set S with the following properties:*

1. *S contains every A such that $|\hat{f}(A)| > \theta$.*
2. $\displaystyle\sum_{A\in S} (\hat{f}(A) - \hat{g}(A))^2 \leq \epsilon$.
3. *$|S|$ is polynomial in $1/\theta$.*

The algorithm uses membership queries, and runs in time polynomial in n, $1/\theta$, $1/\epsilon$, and $\log(1/\delta)$.

[1] Expectations here and elsewhere are with respect to the uniform distribution over the instance space unless otherwise noted.

We use $KM(\theta, \epsilon, \delta)$ to represent an execution of the KM algorithm with the respective threshold, accuracy, and confidence parameters. The fact that it is possible for the algorithm to return only a number of coefficients polynomial in $1/\theta$ follows from the fact that $\sum_A \hat{f}^2(A) = 1$, so there are at most $1/\theta^2$ coefficients with magnitude at least θ.

We now turn to bounding the difference between the target f and the function g returned by the KM algorithm. The following lemma, whose proof is straightforward and omitted, gives us a bound on $\mathbf{E}[(f-g)^2]$ in terms of a lower bound on $\sum_{A \in S} \hat{f}^2(A)$.

Lemma 2 *Given a $\{+1,-1\}$-valued function f, let S be a set such that $\sum_{A\in S} \hat{f}^2(A) \geq \alpha$, and let g be the output of $KM(\sqrt{\alpha/(4|S|)}, \alpha/4, \delta)$. Then with probability at least $1-\delta$, $\mathbf{E}[(f-g)^2] \leq 1-\alpha/2$.*

Now we are ready to link the squared error measure above with the notion of ϵ-approximation.

Lemma 3 *Given a $\{+1,-1\}$-valued function f and a deterministic approximator g, define the randomized function h as follows: let $h(\vec{x}) = -1$ with probability $p = (1-g(\vec{x}))^2/2(1+g^2(\vec{x}))$ and $h(\vec{x}) = 1$ with probability $1-p$. Then $\mathbf{Pr}[f \neq h] \leq \frac{1}{2}\mathbf{E}[(f-g)^2]$. So, if $\mathbf{E}[(f-g)^2] \leq 1-\alpha$ then h is a $1/2-\alpha/2$-approximator for f.*

Proof: First, the algorithm is well-defined since $0 \leq p \leq 1$ for any value of $g(x)$. Noting that $1-p = \mathbf{Pr}[h(x)=1]$ can be written as $(-1-g(x))^2/2(1+g^2(x))$, it follows that for any fixed x, $\mathbf{Pr}[h(x) \neq f(x)] = (f(x)-g(x))^2/2(1+g^2(x))$, where the probability is taken over the random choices made by h. Now considering the distribution over all instances x as well as h's random choices, we get

$$\mathbf{Pr}[h(x) \neq f(x)] \leq \frac{1}{2}\mathbf{E}[(f-g)^2].$$

□(Lemma 3)

A similar but slightly weaker randomized approximation method was given by Kearns, Schapire, and Sellie [10]. Putting the results of this section together, we have the following.

Theorem 4 *A concept class $\mathcal{F}$ is weakly learnable with membership queries with respect to the uniform distribution if there are polynomials p and q such that for every $f \in \mathcal{F}$ there is a set S with $|S| \leq p(n,s)$ such that $\sum_{A\in S} \hat{f}^2(A) \geq 1/q(n,s)$, where s represents the size of f. In particular, for every f in such a class the algorithm*

$$KM\left(\frac{1}{2\sqrt{p(n,s)q(n,s)}}, \frac{1}{4q(n,s)}, \delta\right)$$

plus the approximation scheme of Lemma 3 will with probability at least $1-\delta$ produce a randomized $1/2 - 1/4q(n,s)$-approximation of f. The algorithm runs in time polynomial in n, s, and $\log(1/\delta)$.

4 Positive Results

4.1 Weakly Learning DNF

Linial, Mansour, and Nisan [15] showed that AC^0, the class of constant-depth circuits, is learnable in superpolynomial but subexponential time with respect to the uniform distribution by proving that for every AC^0 function f almost all of the "large" Fourier coefficients of f are coefficients of parities of "few" variables. We show that an even stronger property holds for the Fourier transform of any DNF function, a property which will be key to several of our positive results about DNF learnability. The following definition will simplify the statement and proof of this property.

Definition 1 *Let f be a DNF formula and let T_i (with variables V_i) be a term in f. Then for every $A \subseteq V_i$, define $\chi_A(T_i)$ to be $\chi_A(\vec{x})$, where $\vec{x}$ is any instance which satisfies T_i.*

Lemma 5 *Let f be a DNF formula. Then for every term T_i (with variables V_i),*

$$\sum_{A \subseteq V_i} \hat{f}(A)\chi_A(T_i) = +1.$$

Proof: Consider a particular term T_i of f. Let f_i represent the restriction of f obtained by fixing the variables in V_i so that T_i is satisfied. Then $f_i \equiv +1$. Since $\chi_{\vec{0}} \equiv +1$, $\hat{f}_i(\vec{0}) = \mathbf{E}[f_i \chi_{\vec{0}}] = 1$. Now since $f = \sum \hat{f}(A)\chi_A$, the restriction f_i is also a linear combination of the restrictions $\chi_{A,i}$ of the χ_A's obtained by fixing the variables in V_i as above, that is,

$$f_i = \sum_{A \subseteq \{x_1,\ldots,x_n\}} \hat{f}(A)\chi_{A,i}.$$

For all $A \subseteq V_i$, $\chi_{A,i} = \chi_A(T_i)$ is a constant function. On the other hand, for all $A \not\subseteq V_i$, the restriction $\chi_{A,i}$ is not a constant since some variables in χ_A survive the restriction. Thus $\hat{f}_i(\vec{0}) = \sum_{A \subseteq V_i} \hat{f}(A)\chi_A(T_i)$, and as established above, $\hat{f}_i(\vec{0}) = 1$. □(Lemma 5)

A particularly useful implication of the lemma for our purposes is that for every term T_i in f, there is some $A \subseteq V_i$ such that $|\hat{f}(A)| \geq 2^{-|V_i|}$ Thus if even one term in a DNF f has $O(\log s)$ variables then there is at least one Fourier coefficient of f which is inverse polynomially large. This allows us to use the KM algorithm to weakly learn DNF with membership queries with respect to the uniform distribution.

Theorem 6 *The class of DNF formulas can be $(\frac{1}{2} - \frac{1}{6s})$-approximated by a randomized learning algorithm which uses membership queries, succeeds with probability $1-\delta$, and runs in time polynomial in n, s, and $\log(1/\delta)$, where s is the number of terms in the target formula.*

Proof: We assume that there is at least one term in f with at most $\log(3s)$ literals; otherwise, f is sufficiently well-approximated by the constant -1 function. Thus by Lemma 5 there is at least one Fourier coefficient (call if $\hat{f}(A)$) of magnitude $1/3s$. The parity χ_A corresponding to $\hat{f}(A)$ can be found with probability $1-\delta$ in time polynomial in n, s, and $\log(1/\delta)$ by $KM(1/3s, 1, \delta)$. As $\hat{f}(A)$ represents the correlation of χ_A and f, $g = \text{sign}(\hat{f}(A))\chi_A$ is an adequate approximator. □(Theorem 6)

A related but more complicated algorithm yields improved accuracy (proved in the appendix):

Theorem 7 *The class of DNF formulas can be $(\frac{1}{2} - \Omega(\frac{\log(s)}{s}))$-approximated by a randomized learning algorithm which uses membership queries, succeeds with probability $1-\delta$, and runs in time polynomial in n, s, and $\log(1/\delta)$.*

4.2 Learning Read-k DNF

Lemma 5 gives us that every term has at least one "large" Fourier coefficient associated with it. However, conceivably a small set of large coefficients are shared by many of the terms, so there may be very few large coefficients in the DNF formula. On the other hand, each coefficient (except the constant coefficient) of a read-k formula may be shared by at most k terms. We use this fact to obtain an accuracy bound of $1/2 - \Omega(1/k)$ for the class of read-k DNF.

Theorem 8 *For every k, the class of read-k DNF can be $(\frac{1}{2} - \frac{1}{16k})$-approximated by a randomized learning algorithm which uses membership queries, succeeds with probability $1-\delta$, and runs in time polynomial in n, k, and $\log(1/\delta)$.*

Proof: For any read-k DNF f we will show that there is a set S with $|S| \leq 24n^2k^2$ such that $\sum_{A \in S} \hat{f}^2(A) \geq \frac{1}{4k}$. The result then follows from Theorem 4.

To derive this bound, first consider the case $k = 1$. Lemma 5 implies that for each term T_i: $\sum_{A \in 2^{V_i}} |\hat{f}(A)| \geq 1$ where 2^{V_i} represents the power set of V_i. Define $S = \cup_i 2^{V_i}$. Because $k = 1$, for any $i \neq j$, $2^{V_i} \cap 2^{V_j} = \{\emptyset\}$. Thus, letting S_i denote the set $2^{V_i} - \emptyset$,

$$\sum_{A \in S} \hat{f}^2(A) \geq \sum_i \sum_{A \in S_i} \hat{f}^2(A).$$

We will assume that $|\hat{f}(\vec{0})| \leq \frac{1}{6}$, since otherwise f is adequately approximated by a constant function. Thus for each T_i, $\sum_{A \in S_i} |\hat{f}(A)| \geq \frac{5}{6}$, which implies that $\sum_{A \in S_i} \hat{f}^2(A) \geq (5/6)^2/|S_i|$. So,

$$\sum_{A \in S} \hat{f}^2(A) \geq \left(\tfrac{5}{6}\right)^2 \sum_i \frac{1}{2^{|V_i|}}.$$

By the restriction on $\hat{f}(\vec{0})$ we know that at least $5/12$ of the instances satisfy f. Since the fraction of instances which satisfy a term T_i is $2^{-|V_i|}$, $\sum_i 2^{-|V_i|} \geq 5/12$ and so $\sum_{A \in S} \hat{f}^2(A) \geq (\frac{5}{6})^2(\frac{5}{12}) > 1/4$.

Now consider larger k. In this case, for any given set A we can have $A \in S_i$ for up to (but no more than) k distinct values of i. Thus

$$\sum_{A \in S} \hat{f}^2(A) \geq \frac{1}{k} \sum_i \sum_{A \in S_i} \hat{f}^2(A)$$

and therefore $\sum_{A \in S} \hat{f}^2(A) > \frac{1}{4k}$.

Finally, we need to bound $|S|$. In general, the set S above can be exponentially large even for a read-once DNF. We get around this by considering only "small" terms when constructing S. Specifically, we now let

$$S = \bigcup_{|V_i| \leq \log(24kn)} 2^{V_i}.$$

Because there are at most kn terms in a read-k DNF, the terms which are excluded from S are satisfied by at most $1/24$ of the instances. The included terms are therefore satisfied by at least $\frac{5}{12} - \frac{1}{24} = \frac{9}{24}$ of the instances, and using this value rather than $5/12$ in the earlier analysis still gives the desired bound. □(Theorem 8)

4.3 Learning SAT-k DNF

We demonstrate the (strong) learnability of SAT-k DNF for constant k by showing that every SAT-k DNF is well-approximated by a function with small support.

Theorem 9 [2] *For any k, the class of SAT-k DNF formulas can be ϵ-approximated by a randomized learning algorithm which uses membership queries, succeeds with probability $1 - \delta$, and runs in time polynomial in n, s^k, $1/\epsilon^k$, and $\log(1/\delta)$.*

Proof: We will show that there is some polynomially sparse deterministic function g such that $\mathbf{E}[(f - g)^2] \leq \epsilon/2$. The result then follows from standard arguments.

Let $r = 8s/\epsilon$ and let g be what remains of f after removing any terms having more than $\log(r)$ variables. Then $\mathbf{E}[(f - g)^2] = 4\mathbf{Pr}[f \neq g] \leq \epsilon/2$. The inequality holds because each term removed from f covers at most an $\epsilon/8s$ fraction of the instance space. To see that g has small support, let s' represent the number of terms in g and define $P_i(\vec{x})$, $1 \leq i \leq s'$, to be $+1$ if $\vec{x}$ satisfies the ith term of g and 0 if $\vec{x}$ does not. At most $\log(r)$ variables are relevant for P_i and thus the Fourier representation of P_i has no more than r non-zero coefficients. Using the principle of

[2] A similar result has also been shown by Lipton using a somewhat different analysis [16].

inclusion-exclusion we can create a function P' from the P_i's which is $(rs')^k$-sparse and which is 1 when g is satisfied and 0 otherwise. Specifically, let

$$P' = \sum_{i_1} P_{i_1} - \sum_{i_1<i_2} P_{i_1}P_{i_2} + \sum_{i_1<i_2<i_3} P_{i_1}P_{i_2}P_{i_3} - \cdots - (-1)^k \sum_{i_1<\cdots<i_k} P_{i_1}\cdots P_{i_k}.$$

It can be verified inductively that this polynomial has the claimed properties. Noting that $g = 2P' - 1$ completes the proof. □(Theorem 9)

By restricting the size of terms in the SAT-k DNF's considered we can extend the above to a distribution-free learning result (this generalizes a similar result for SAT-1 (disjoint) DNF by Khardon [11]).

Theorem 10 *For any k, the class of SAT-k $O(\log s)$-DNF formulas of s terms can be learned exactly by a deterministic learning algorithm which uses membership queries and runs in time polynomial in n, s^k, $1/\epsilon^k$, and $\log(1/\delta)$.*

5 Characterizing Learnability in the Statistical Query Model

In this section we present results that characterize when a given class of functions is weakly learnable under any given distribution in the statistical query model. An important corollary of this characterization is that the class of parity functions on $\log(n)$ variables (that is, the class of functions χ_A where $|A| = O(\log n)$) over $\{0,1\}^n$ cannot be weakly learned with a polynomial number of queries with inverse polynomial tolerance in this model. This immediately implies that DNF and decision trees, both of which contain the $\log(n)$-bit parities as a subclass, are not efficiently weakly learnable in the statistical query model.

Our lower bounds are particularly strong in that they are information-theoretic (and thus do not rely on any unproven assumptions), and for our matching upper bounds we actually give (non-uniform) *polynomial time* weak learning algorithms. Thus, the situation in the statistical query model is quite different from that in the PAC model, where information-theoretic learnability provably does not imply polynomial time learnability given certain cryptographic assumptions.

In order to present our characterization, we need the following definition.

Definition 2 *For $\mathcal{F}$ a class of boolean functions over $\{0,1\}^n$ and D a distribution over $\{0,1\}^n$, we define SQ-DIM$(\mathcal{F}, D)$, the* statistical query dimension of $\mathcal{F}$ with respect to D, *to be the largest natural number d such that $\mathcal{F}$ contains d functions $f_1, \ldots, f_d$ with the property that for all $i \neq j$ we have:*

$$|\mathbf{Pr}_D[f_i = f_j] - \mathbf{Pr}_D[f_i \neq f_j]| \leq \frac{1}{d^3}.$$

The main theorems of this section are the following.

Theorem 11 *Let $\mathcal{F}$ be a class of boolean functions over $\{0,1\}^n$ and D a distribution such that SQ-DIM$(\mathcal{F}, D) \geq d \geq 16$. Then if all statistical queries are made with tolerance at least $1/d^{1/3}$, at least $d^{1/3}/2$ statistical queries are needed to learn $\mathcal{F}$ to error less than $1/2 - 1/d^3$.*

Theorem 12 *If $\mathcal{F}$ is a class of boolean functions over $\{0,1\}^n$ and D is a distribution such that SQ-DIM$(\mathcal{F}, D) = d$, then there is a non-uniform polynomial-time (in d) algorithm to weakly learn $\mathcal{F}$ with respect to D in the statistical query model that makes d queries of tolerance $\frac{1}{3d^3}$ and finds a hypothesis with error at most $\frac{1}{2} - \frac{1}{3d^3}$.*

If we think of $\mathcal{F}$ and D as function and distribution *ensembles* (one for each n), then the above theorems imply the following. If for all polynomials $p(\cdot)$ and infinitely many n we have SQ-DIM$(\mathcal{F}, D) \geq p(n)$, then $\mathcal{F}$ is not weakly learnable in the statistical query model with respect to distribution D. On the other hand, if there exists a polynomial $p(\cdot)$ such that for all sufficiently large n, SQ-DIM$(\mathcal{F}, D) \leq p(n)$, then there is a non-uniform polynomial time weak learning algorithm for $\mathcal{F}$ with respect to D in the statistical query model.

As promised, we have the following corollary.

Corollary 13 *There exists a constant $c > 0$ such that $\Omega(n^{c \log n})$ statistical queries of tolerance $O(1/n^{c \log n})$ are required to weakly learn the classes of polynomial size DNF formulae and polynomial size decision trees with respect to the uniform distribution. Thus, these classes are not efficiently learnable in the statistical query model.*

Because the proof of Theorem 12 is significantly easier than the proof of Theorem 11, we give it first.

Proof of Theorem 12: The nonuniform algorithm has "hardwired" a maximal set of functions $f_1, \ldots, f_d$ such that for all $i \neq j$, $|\langle f_i, f_j \rangle| \leq \frac{1}{d^3}$. The algorithm makes d queries, each with tolerance $\frac{1}{3d^3}$. The ith query g_i is simply a request for the correlation of the target function with f_i, that is, $g_i(\vec{x}, \ell) = \ell f_i(\vec{x})$. By assumption, the set $\{f_1, \ldots, f_d\}$ is maximal (with the desired property) so at least one query g_i will return a value at least $\frac{1}{d^3} - \tau \geq \frac{2}{3d^3}$. Thus we have *found* an f_i such that $\langle f_i, f \rangle \geq \frac{2}{3d^3} - \tau \geq \frac{1}{3d^3}$, where f is the target function, and we can use f_i as our weak hypothesis. □(Theorem 12)

In the following proof, it will be helpful to keep in mind that our eventual approach will be to perform a Fourier analysis not only of the functions in the target class $\mathcal{F}$, but also of the *query* function $g : \{0,1\}^n \times \{+1, -1\} \rightarrow \{+1, -1\}$. Recall that such a query is a request from the learner for an approximation to $\mathbf{E}_D[g(\vec{x}, f(\vec{x}))]$, where D is the target distribution and f is the target function.

Proof of Theorem 11: In order to prove this theorem, we will need to use an extension of the Fourier theory to an arbitrary distribution; this extension has been examined in the learning theory literature before by Furst, Jackson and Smith [7]. Thus let D be an arbitrary probability distribution over $\{0,1\}^n$. Then for any two real-valued functions f and g over $\{0,1\}^n$, we can define the *inner product with respect to D* by

$$\langle f, g \rangle_D = \mathbf{E}_D[fg] = \sum_{\vec{x} \in \{0,1\}^n} D[\vec{x}] f(\vec{x}) g(\vec{x}).$$

It is easy to verify that $\langle , \rangle_D$ is in fact an inner product for the vector space of all real-valued functions over $\{0,1\}^n$, and we shall use this in our analysis. If, as usual, we regard the boolean functions $f_1, \ldots, f_d$ as being $\{+1, -1\}$-valued, then the assumption of the theorem gives that $|\langle f_i, f_j \rangle_D| \leq 1/d^3$ for all $i \neq j$. It is also easy to see that for any $\{+1, -1\}$-valued function f, $\langle f, f \rangle_D = 1$.

In the analysis to follow, we wish to use the given functions $f_1, \ldots, f_d$ as the beginnings of a basis for the vector space of all functions. To do this, we will need the following lemma.

Lemma 14 *The functions $f_1, \ldots, f_d$ are linearly independent.*

Proof: Without loss of generality, assume for contradiction that we could write $f_1 = \sum_{i \geq 2} \alpha_i f_i$ for some real coefficients $\alpha_2, \ldots, \alpha_d$. Then we have

$$\begin{aligned} 0 &= \mathbf{E}_D\left[(f_1 - \sum_{i \geq 2} \alpha_i f_i)^2\right] \\ &= \mathbf{E}_D[f_1^2] - 2\sum_{i \geq 2} \alpha_i \mathbf{E}_D[f_1 f_i] + \sum_{i,j \geq 2} \alpha_i \alpha_j \mathbf{E}_D[f_i f_j] \\ &= 1 - 2\sum_{i \geq 2} \alpha_i \mathbf{E}_D[f_1 f_i] + \sum_{i \geq 2} \alpha_i^2 + \sum_{i,j \geq 2, i \neq j} \alpha_i \alpha_j \mathbf{E}_D[f_i f_j] \end{aligned}$$

where we have used that $\mathbf{E}_D[f_i^2] = 1$ for all i. Our goal is to reach a contradiction by showing that this final expression is strictly larger than 0. Let us define $\alpha_{\max} = \max\{|\alpha_i| : i \geq 2\} \geq 0$, and use $\alpha_{\max}$ to simplify the expression above. Then $1 + \sum_{i \geq 2} \alpha_i^2 \geq 1 + \alpha_{\max}^2$. Also, $|2\sum_{i \geq 2} \alpha_i \mathbf{E}_D[f_1 f_i]| \leq 2\alpha_{\max}/d^2$ since $\mathbf{E}_D[f_1 f_i] = \langle f_1, f_i \rangle_D \leq 1/d^3$ for all $i \neq 1$. Finally, $|\sum_{i,j \geq 2, i \neq j} \alpha_i \alpha_j \mathbf{E}_D[f_i f_j]| \leq \alpha_{\max}^2/d$. So the above sum is at least: $1 + \alpha_{\max}^2 - 2\alpha_{\max}/d - \alpha_{\max}^2/d$, which is always positive. □(Lemma 14)

Before extending $f_1, \ldots, f_d$ to a complete basis, we argue that without loss of generality we can assume that the support of the distribution D is all of $\{0,1\}^n$. If we regard D as a linear transformation of the vector space of all real functions over $\{0,1\}^n$, this is simply saying that D has full rank. The reason we may assume this is that if D does *not* have support $\{0,1\}^n$, we can instead carry out the ensuing analysis using a distribution D' that *does* have support $\{0,1\}^n$, and is obtained from D by taking an infinitesimally small amount of weight away from the support of D, and spreading this weight uniformly among the vectors not in the support of D. Then the functions $f_1, \ldots, f_m$ will still be approximately orthogonal, and it is not hard to prove that any statistical query lower bound we can prove for D' must also hold for D, since the learning algorithm cannot distinguish D and D' (details are omitted).

Now using the Gram-Schmidt process, which applies to any inner product space, we may extend the functions $f_1, \ldots, f_d$ to obtain a basis $f_1, \ldots, f_d, f_{d+1}, \ldots, f_{2^n}$ for the vector space of all real functions over $\{0,1\}^n$ with the property that for any $i \geq d+1$ and any j, $\langle f_i, f_j \rangle_D = 0$, and for any i, $\langle f_i, f_i \rangle_D = 1$. Note that our basis may *not* be orthonormal due to the fact that for $i, j \leq d$, $\langle f_i, f_j \rangle_D$ may be as large as $1/d^3$. Also, note that we may assume there are 2^n basis functions: since D has full rank, the 2^n delta functions on $\{0,1\}^n$ are orthogonal and non-zero with respect to D.

We now wish to extend $f_1, \ldots, f_{2^n}$ to a basis for the space of all real functions on $\{0,1\}^n \times \{+1,-1\}$. To accomplish this, it will be most convenient to use an inner product defined by a distribution $\tilde{D}$ on $\{0,1\}^n \times \{+1,-1\}$ that extends the distribution D, where $\tilde{D}$ is simply the product of D and the uniform distribution on $\{+1,-1\}$. To extend $f_1, \ldots, f_{2^n}$, we define for each $1 \leq i \leq 2^n$ the function $h_i(\vec{x}, y) = y f_i(\vec{x})$, where $\vec{x} \in \{0,1\}^n$ and $y \in \{+1,-1\}$. We also regard each of the original basis functions f_i as a function over $\{0,1\}^n \times \{+1,-1\}$, where $f_i(\vec{x}, y) = f_i(\vec{x})$. We now verify that $f_1, \ldots, f_{2^n}$ together with $h_1, \ldots, h_{2^n}$ is in fact a basis for all functions on $\{0,1\}^n \times \{+1,-1\}$ under the inner product $\langle , \rangle_{\tilde{D}}$. We have

$$\langle h_i, f_j \rangle_{\tilde{D}} = \frac{1}{2}\mathbf{E}_D[h_i(\vec{x},1) f_j(\vec{x})] + \frac{1}{2}\mathbf{E}_D[h_i(\vec{x},-1) f_j(\vec{x})] = \frac{1}{2}\mathbf{E}_D[f_i f_j(\vec{x})] + \frac{1}{2}\mathbf{E}_D[-f_i f_j(\vec{x})] = 0$$

and

$$\langle h_i, h_j \rangle_{\tilde{D}} = \frac{1}{2}\mathbf{E}_D[h_i(\vec{x},1) h_j(\vec{x},1)] + \frac{1}{2}\mathbf{E}_D[h_i(\vec{x},-1) h_j(\vec{x},-1)] = \mathbf{E}_D[f_i f_j]$$

and $\mathbf{E}_D[f_i f_j] = 0$ unless $i, j \leq d$, in which case it is bounded by $1/d^3$, or unless $i = j$, in which case it equals 1. So by the same argument as in Lemma 14 we have $2 \cdot 2^n$ independent functions, forming a basis for functions over $\{0,1\}^n \times \{+1,-1\}$.

Now let $g : \{0,1\}^n \times \{+1,-1\} \to \{+1,-1\}$ be any statistical query. We will soon perform a Fourier analysis of the expectation $\mathbf{E}_D[g(\vec{x}, f(\vec{x}))]$, which is the quantity that is approximated by the response of the query. Because we have a basis, we can write $g = \sum_{i \geq 1} \alpha_i f_i + \sum_{i \geq 1} \beta_i h_i$ for some real coefficients α_i and β_i. Note that it is not true that $\alpha_i = \langle g, f_i \rangle_{\tilde{D}}$ and $\beta_i = \langle g, h_i \rangle_{\tilde{D}}$ because we do not have an orthonormal basis. However, the following bound on the coefficients will serve our purposes.

Lemma 15 *If $g = \sum_{i \geq 1} \alpha_i f_i + \sum_{i \geq 1} \beta_i h_i$, where the f_i and h_i are as defined above, then $|\alpha_i|, |\beta_i| \leq 2$ for all i.*

Proof: Without loss of generality, let $\alpha_1 > 0$ be the largest coefficient. Since we have an inner product space, we can define the f_1-component of g by

$$\langle f_1, g \rangle_{\tilde{D}} f_1 = \left(\alpha_1 + \sum_{i \geq 2} \alpha_i \langle f_1, f_i \rangle_D + \sum_{i \geq 2} \beta_i \langle f_1, h_i \rangle_{\tilde{D}} \right) f_1.$$

Again due to the properties of an inner product, we must have

$$||g|| = \sqrt{\mathbf{E}_{\tilde{D}}[g^2]} \geq \left| \alpha_1 + \sum_{i \geq 2} \alpha_i \langle f_1, f_i \rangle_D + \sum_{i \geq 2} \beta_i \langle f_1, h_i \rangle_{\tilde{D}} \right|.$$

But each summation inside the absolute value is at most α_1/d^2, so the absolute value is at least $\alpha_1 - 2\alpha_1/d^2 > \alpha_1/2$ for $d \geq 2$. Since $||g|| = 1$, the lemma follows. □(Lemma 15)

We are now finally in position to analyze the quantity of interest, the expected value of the query g. Let the target function be f_j for some $1 \leq j \leq d$; thus, we choose as the target one of the original nearly orthogonal functions in the target class $\mathcal{F}$. We may write:

$$\begin{aligned}
\mathbf{E}_D[g(\vec{x}, f_j(\vec{x}))] &= \mathbf{E}_D\left[\sum_{i\geq 1} \alpha_i f_i(\vec{x}) + \sum_{i\geq 1} \beta_i h_i(\vec{x}, f_j(\vec{x}))\right] \\
&= \sum_{i\geq 1} \alpha_i \mathbf{E}_D[f_i] + \sum_{i\geq 1} \beta_i \mathbf{E}_D[f_i f_j] \\
&= C + \sum_{i\leq d} \beta_i \langle f_i, f_j\rangle_D
\end{aligned}$$

where $C = \sum_{i\geq 1} \alpha_i \mathbf{E}_D[f_i]$ is a constant independent of the target function f_j and we have used the fact that $\langle f_i, f_j\rangle_D = 0$ unless $i \leq d$. Now

$$\beta_j - \frac{2}{d^2} \leq \sum_{i\leq d} \beta_i \langle f_i, f_j\rangle_D \leq \beta_j + \frac{2}{d^2}$$

since $\langle f_j, f_j\rangle_D = 1$ and $\langle f_i, f_j\rangle_D \leq 1/d^3$ for $i \neq j$, and $|\beta_i| \leq 2$ for all i by Lemma 15. Thus, we see that the only contribution the target function makes to the expected value of the query is in determining the coefficient β_j, plus an $O(1/d^2)$ contribution. For the lower bound, the statistical query g will always be answered with the value C. We now analyze how many functions in $f_1, \ldots, f_d$ can be eliminated by this answer. For this, we need the following final lemma.

Lemma 16

$$\sum_{i\leq d} \beta_i^2 \leq 2.$$

Proof: Using Lemma 15 and the $1/d^3$ bounds on the inner products, it is easy to verify that $\mathbf{E}_{\tilde{D}}[g^2] = 1$ is bounded above and below by $\sum_{i\leq d} \alpha_i^2 + \sum_{i\leq d} \beta_i^2 \pm 16/d$. This implies $\sum_{i\leq d} \beta_i^2 \leq 1 - \sum_{i\leq d} \alpha_i^2 + 16/d \leq 2$ provided $d \geq 16$. □(Lemma 16)

Now if the query g is made with tolerance as large as $1/d^{1/3}$, then by the preceding arguments the function f_j is eliminated by the query response C only if β_j exceeds $1/d^{1/3}$. But by Lemma 16, if r is the number of functions f_j in $f_1, \ldots, f_d$ such that β_j exceeds $1/d^{1/3}$, then we must have $r(1/d^{1/3})^2 \leq 2$, or $r \leq 2d^{2/3}$. This shows that at least $d^{1/3}/2$ queries of allowed approximation error bounded above by $1/d^{1/3}$ are required in order to eliminate all the functions in $f_1, \ldots, f_d$ that are not the target function. If there are even two functions remaining, by choosing adversarially between the remaining functions we may force the error of the learning algorithm's hypothesis to be $1/2 - 1/d^3$ with significant probability. □(Theorem 11)

Note that in the above proof, even if the learning algorithm is randomized, if it makes only $d^{1/3-\epsilon}$ queries for some constant $\epsilon > 0$, it will eliminate only a small fraction of $f_1, \ldots, f_d$. So if the adversary picks f_j at random from $f_1, \ldots, f_d$, with high probability it can answer as above for each query and so again with high probability the algorithm's error will be close to 1/2.

It is also instructive to note that if the tolerance $\tau = 0$, then over the uniform distribution the statistical query model allows one to make membership queries (one can ask whether the probability of a specific labeled example is non-zero). So the algorithmic results in the previous sections prove that such a lower bound cannot hold when 0-tolerance queries may be made.

References

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

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

[3] Howard Aizenstein and Leonard Pitt. Exact learning of read-k disjoint DNF and not-so-disjoint DNF. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 71–76, 1992.

[4] Dana Angluin, Michael Frazier, and Leonard Pitt. Learning conjunctions of Horn clauses. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 186–192, 1990.

[5] Avrim Blum and Steven Rudich. Fast learning of k-term DNF formulas with queries. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*, pages 382–389, 1992.

[6] Nader H. Bshouty. Exact learning via the monotone theory. In *Proceedings of the 34th Annual Symposium on Foundations of Computer Science*, pages 302–311, 1993.

[7] Merrick L. Furst, Jeffrey C. Jackson, and Sean W. Smith. Improved learning of AC^0 functions. In *Fourth Annual Workshop on Computational Learning Theory*, pages 317–325, 1991.

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

[9] Michael J. Kearns. Efficient noise-tolerant learning from statistical queries. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing*, pages 392–401, 1993.

[10] Michael J. Kearns, Robert E. Schapire, and Linda M. Sellie. Toward efficient agnostic learning. In *Fifth Annual Workshop on Computational Learning Theory*, pages 341–352, 1992.

[11] Roni Khardon. On using the Fourier transform to learn disjoint DNF. Unpublished Manuscript, 9 1993.

[12] Michael Kharitonov. Cryptographic hardness of distribution-specific learning. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, pages 372–381, 1993.

[13] Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the Fourier spectrum. In *Proceedings of the Twenty Third Annual ACM Symposium on Theory of Computing*, pages 455–464, 1991.

[14] Eyal Kushilevitz and Dan Roth. On learning visual concepts and DNF formulae. In *Proceedings of the Sixth Annual Workshop on Computational Learning Theory*, pages 317–326, 1993.

[15] Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. In *30th Annual Symposium on Foundations of Computer Science*, pages 574–579, 1989.

[16] Richard Lipton. Personal communication.

[17] Yishay Mansour. An $O(n^{\log\log n})$ learning algorithm for DNF under the uniform distribution. In *Fifth Annual Workshop on Computational Learning Theory*, pages 53–61, 1992.

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

6 Appendix

Proof of Theorem 7:

The proof utilizes the following relationship between a function and its restrictions.

Lemma 17 *For any variable x_i in function f let f_0 (f_1) be the restriction of f obtained by fixing $x_i = 0$ ($x_i = 1$). Also, let S_0 and S_1 be sets of subsets of variables such that for each $A \in S_0 \cup S_1$, $x_i \notin A$. Then for all S such that $S \supseteq S_0 \cup S_1 \cup \{A \cup \{x_i\} \mid A \in S_0 \cup S_1\}$,*

$$\sum_{A \in S} \hat{f}^2(A) \geq \frac{1}{2}\left(\sum_{A \in S_0} \hat{f}_0^2(A) + \sum_{A \in S_1} \hat{f}_1^2(A)\right).$$

Proof: First, observe that $\sum_{A \in S_0} \hat{f}_0^2(A) + \sum_{A \in S_1} \hat{f}_1^2(A) \leq \sum_{A \in S_0 \cup S_1} \hat{f}_0^2(A) + \hat{f}_1^2(A)$. It follows from the definition of the Fourier transform that for $A \in S_0 \cup S_1$, $\hat{f}_0(A) = \hat{f}(A) + \hat{f}(A \cup \{x_i\})$ and $\hat{f}_1(A) = \hat{f}(A) - \hat{f}(A \cup \{x_i\})$. Thus for any such A, $\hat{f}_0^2(A) + \hat{f}_1^2(A) = 2(\hat{f}^2(A) + \hat{f}^2(A \cup \{x_i\}))$. Summing over all $A \in S_0 \cup S_1$ gives the result. □(Lemma 17)

For the proof of Theorem 7, we will show that for every DNF formula f there is a set S with $|S| \leq 64s^3$ such that $\sum_{A \in S} \hat{f}^2(A) \geq \log(s)/8s$; the result then follows from Theorem 4.

For the moment we restrict our attention to $\log(s)$-DNF. For the set of variables V_i in a term T_i of f, let $\{f_c\}$ represent the $2^{|V_i|}$ restrictions of f obtained by fixing the variables in V_i in all possible ways. Notice that if S is a set such that $2^{V_i} \subseteq S$ then, by recursive application of Lemma 17 on the variables in V_i, $\sum_{A \in S} \hat{f}^2(A) \geq 1/2^{|V_i|} \sum_c \hat{f}_c^2(\vec{0})$. One of these restrictions satisfies T_i, and thus for the associated f_c we have $\hat{f}_c(\vec{0}) = +1$. Therefore

$$\sum_{A \in S} \hat{f}^2(A) \geq 1/2^{|V_i|}. \tag{1}$$

While this inequality can be derived directly from Lemma 5, the derivation above gives the flavor of our approach to improving the accuracy bound. Conceptually, we can imagine building a decision tree to approximate a given DNF f. When we reach a point in the tree at which the value of the DNF is determined ($+1$ or -1) then we create a leaf with the appropriate value. At some depth we terminate the tree with leaves having value 0. Lemma 17 then implies that the expected value of the squared leaf values represents the sum (over the sets of variables on each path in our tree) of the squares of Fourier coefficients of f. Thus the above bound follows from the fact that for any term T_i in a DNF we can construct a depth $\log(|V_i|)$ tree $\mathcal{T}$ as described such that at least one leaf of $\mathcal{T}$ has value 1. We improve on this bound by building a somewhat deeper tree.

First, we formalize the tree-building notion above. Recursively define a *restriction tree* for a DNF f as follows. At any point in the construction of this tree we are at a node R and are building the restriction tree for some function f_c a restriction of f. If f_c is a constant function then we label R with the triple $(1, P, \emptyset, f_c)$ and stop. Here P is the set of variables labeling edges on the path from R to the root; these labels will be defined shortly. If f_c is not constant then we label R with $(2^{-|V_i|}, P, V_i, f_c)$, where T_i is the smallest term in f_c (ties are broken arbitrarily). Next we select a variable x_i in T_i and label one of the edges leaving R with $x_i = 0$ and the other with $x_i = 1$. For each child C_j of R we repeat this process, building the restriction tree for the function formed by restricting f_c according to the label on the edge from R to C_j. We begin the overall process of building the restriction tree for f by creating a root node and setting $f_c = f$.

We claim that every label (α, P, V, f_c) of a node R of the restriction tree satisfies the following properties:

- f_c is a restriction of f on the variables in P, in particular, the restriction of these variables according to the labels on the path from R to the root.
- $\sum_{A \in 2^V} \hat{f}_c^2(A) \geq \alpha$.

The first property is an obvious consequence of the construction. The second follows from the reasoning used to establish the bound in (1) for nonconstant f_c. For constant f_c, since $f_c \in \{+1, -1\}$ then $\hat{f}_c^2(\vec{0}) = 1$.

Now consider the nodes at depth $\log(s)$ of the restriction tree. Assume for now that all s possible nodes are present in the tree. Then the numerical value of at least one of these node labels is 1 since on at least one of the paths we are satisfying the smallest term in a restriction of f with every assignment and there are initially at most $\log(s)$ variables in any term. Similarly, at least one other node has value at least 1/2, in particular the node which corresponds to choosing the "wrong" assignment at the first step and choosing satisfying assignments thereafter. Furthermore, there are at least two nodes with value 1/4, four with value 1/8, and in general 2^{i-1} with value 2^{-i} for $1 \leq i \leq \log(s)$. Therefore the expected value of a node at depth $\log(s)$ is at least $\log(s)/2s$.

Finally, let the label of node R_i be $(\alpha_i, P_i, V_i, f_c)$, and let $S = \cup_{R_i} 2^{P_i \cup V_i}$, where the union is over all nodes R_i at level $\log(s)$ in the restriction tree for f. Clearly $|S| \leq s^3$. Now applying Lemma 17 bottom-up from level $\log(s) - 1$ of the tree, we find that

$$\sum_{A \in S} \hat{f}^2(A) \geq \frac{\log(s)}{2s}.$$

To complete the proof we remove the assumptions which have been made. First, note that if there are fewer than s nodes at depth $\log(s)$ of the restriction tree it is because one or more paths to that level was cut off when f_c became a constant function at some higher-level node. This can only improve our bound, since this is equivalent to giving all the missing nodes value 1.

We remove the assumption that each term has at most $\log(s)$ variables in two steps. First, we relax the assumption slightly and allow up to $\log(4s)$ variables in each term of f. We can build a restriction tree for such an f exactly as before, but now we will consider the nodes at level $\log(4s)$. This changes only constant factors in the preceding analysis. To handle arbitrary DNF, we modify our definition of restriction trees slightly: if at any node R the smallest term T_i of the restricted function f_c has $|V_i| \geq \log(4t)$ then we label R with $(1/4, P, \emptyset, f_c)$ and stop. This labeling maintains the label properties identified above since $\hat{f}_c(\vec{0}) \leq -1/2$. Furthermore, by an argument similar to that above, cutting off the tree in this way cannot hurt our lower bound. Finally, note that the set S created from this tree has at most $64s^3$ elements since each of the sets in the labels has at most $\log(4s)$ elements. □(Theorem 7)

As a corollary of this proof we can show that for every DNF f with s terms there is a decision tree of depth $O(\log(s))$ which $(\frac{1}{2} - \Omega(\frac{\log(s)}{s}))$-approximates f.