On the Fourier spectrum of monotone functions

Nader H. Bshouty Christino Tamon
Department of Computer Science
University of Calgary
2500 University Drive NW
Calgary, AB Canada T2N 1N4
e-mail:{bshouty,tamon}@cpsc.ucalgary.ca

Abstract

We prove that *any* monotone boolean function has most of its power spectrum on its Fourier coefficients of "degree" at most $O(\sqrt{n})$ under *any* product distribution. This is similar to a result of Linial, Mansour and Nisan [LMN93] which showed that AC^0 functions have almost all of its power spectrum on the coefficients of degree at most $\log^{O(1)} n$ under the uniform distribution. As a consequence of our result we obtain the following two corollaries:

- For any constant ϵ, monotone boolean functions are PAC learnable with error ϵ under product distributions in time $2^{\tilde{O}(\frac{1}{\epsilon}\sqrt{n})}$.
- Any monotone boolean function can be approximated within ϵ under any product distribution by a boolean circuit of size $2^{\tilde{O}(\frac{1}{\epsilon}\sqrt{n})}$ and depth $\tilde{O}(\frac{1}{\epsilon}\sqrt{n})$.

The learning algorithm that we present is subexponential as long as the required error is greater than $\tilde{\Omega}(1/\sqrt{n})$. This bound is tight in the sense that for any algorithm that runs in subexponential time, there is a monotone function that this algorithm cannot approximate with error better than $\tilde{O}(1/\sqrt{n})$. We also obtain similar learning results under any distribution D that has subexponential "convex dimension" which is defined as the minimal number of product distributions which contains D in their convex space.

We apply the above results for other problems in learning and complexity theory. In learning theory we provide polynomial-time algorithms for learning some classes of monotone boolean functions such as boolean functions with $\tilde{O}(\log^2 n)$ relevant variables. In complexity theory we apply the results to some monotone NP-complete problems.

1 Introduction

In recent years, the discrete Fourier transform has emerged as one of the most versatile tools in theoretical computer science. This technique has found various applications in areas such as circuit complexity [KKL88, BS92], computational learning theory [LMN93, FJS91, KM91, AM91, B92, M92, BFJ+, J94], cryptography [CKM93] and others. The first paper which introduced the transform to the learning community was the paper of Linial, Mansour and Nisan [LMN93]. Among other results, they proved that AC^0 functions are PAC learnable in time $n^{poly(\log n)}$ under the uniform distribution and they showed that any AC^0 functions has almost all of its power spectrum on the Fourier coefficients of polylogarithmic degree.

We prove that *any* monotone boolean function has most of its power spectrum on its Fourier coefficients of *degree* at most $O(\sqrt{n})$ under *any* product distribution. Our spectral result on monotone functions implies the following two corollaries, one in learning theory and another in circuit complexity.

- For any constant ϵ and any product distribution, the class of monotone boolean functions is PAC learnable with error ϵ in time $2^{\tilde{O}(\frac{1}{\epsilon}\sqrt{n})}$.

- For any constant ϵ, any product distribution, and any monotone function, there exists a boolean circuit of size $2^{\tilde{O}(\frac{1}{\epsilon}\sqrt{n})}$ and depth $\tilde{O}(\sqrt{n})$ that ϵ-approximates the function.

Note that the learning complexity of our algorithm is subexponential as long as $\epsilon = \tilde{O}(n^{-1/2})$. Moreover we also show that this is the best possible error rate for any subexponential time algorithm. The learning result above is the first subexponential PAC learning algorithm for monotone boolean functions (even for monotone functions which have exponential circuit size). The second result is the the first approximation result for monotone boolean functions using non-monotone boolean circuit of subexponential size and sublinear depth.

We also introduce a new measure of complexity for distributions called the *convex dimension* of a distribution. We prove that if a concept class is learnable under a collection of distributions then it is also learnable under any distribution that lies in the convex space of that collection. The convex dimension $cdim(D)$ of a distribution D is the minimal number of product distributions such that D is in their convex linear combination. We note that our previous results hold also for any distribution D modulo a complexity factor of $cdim(D)$ and in particular, our learning algorithm is subexponential for any distribution D with a subexponential convex dimension.

Furthermore we provide some improvements on two polynomial time monotone learning algorithms. Some of our motivation for this work was Kearns and Valiant's result on the $\frac{1}{2} + \frac{1}{2n}$ weak learnability of any monotone function under the uniform distribution. Using a result of Kahn *et al.* [KKL88], we found a weak learning algorithm under the uniform distribution with error $\frac{1}{2} - \Omega(\frac{\log^2 n}{n})$. Under any product distribution, we found a weak learning algorithm with error $\frac{1}{2} - \frac{c}{n}$, for any constant c.

Sakai and Maruoka [SM94] have recently shown that monotone $O(\log n)$-term DNF is PAC learnable under the uniform distribution. We improve their result in two ways. First we prove that their result extends to any constant-bounded product distribution and then we give an extension to a more general concept class. In particular, let $A(k)$ be the class of functions of the form $f(T_1, \ldots, T_k)$ where f is a monotone function on k inputs and each T_i is a monotone conjunction or disjunction. Our theorem states that there are PAC learning algorithms for $A(\log n)$, for $A(\tilde{O}(\log^2 n))$ and for monotone functions which depend on $\tilde{O}(\log^2 n)$ variables (the last two results require constant ϵ).

Aiello and Mihail [AM91] proved that a convex combination of functions with bounded Fourier spectrum is PAC learnable under the uniform distribution. We prove a similar statement by showing that a convex combination of monotone functions is weakly PAC learnable.

We also apply some of our results to monotone graph properties. We show that there is a boolean circuit of size $2^{\tilde{O}(\frac{n}{\epsilon}\sqrt{r(n)})}$ and depth $\tilde{O}(\frac{n}{\epsilon}\sqrt{r(n)})$ that ϵ-approximates any monotone graph property with a threshold function of $r(n)$. For example, there is a boolean circuit of size $n^{O(\frac{1}{\epsilon}\sqrt{\log n})}$ that ϵ-approximates the Hamiltonian property on random graphs $G(n,p)$, for any p.

The impact of the discrete Fourier transform on learning theory cannot be underestimated judging from the sequence of papers that followed the paper [LMN93]. This technique alone has made some outstanding results possible in recent years culminating in Jackson's result [J94] on the PAC learnability of DNF formulas under the uniform distribution with membership queries. As a final remark, we mention that the previous works of Kahn, Kalai and Linial [KKL88] on coin-flipping protocols and Hancock and Mansour [HM91] on learning read-k formulas have also considered monotone functions in relation with the Fourier transform.

2 The Learning Model

The learning model considered in this paper is the PAC model [V84] along with its weak variant [KV89]. Let C_n be a class of boolean functions over n variables, D_n be a distribution over $\{0,1\}^n$, and $f \in C_n$. The learning algorithm has access to an example oracle $EX_{D,f}()$ which generates labeled examples $(a, f(a))$, where $a \in \{0,1\}^n$ is drawn according the distribution D. Given any positive ϵ and δ, after observing some examples, the learning algorithm must output a hypothesis h that satisfies

$$\Pr[D(h \triangle f) \leq \epsilon] \geq 1 - \delta.$$

The running time of the learner should be polynomial in $n, 1/\epsilon, \ln(1/\delta)$ and the size $s(f)$ of f (under some predetermined representation). If there is such a learning algorithm that succeeds for all $f \in C_n$ then C_n is PAC learnable under distribution D. If $\epsilon = 1/2 - 1/poly(n, s(f))$, then C_n is weakly PAC learnable under distribution D.

3 Preliminaries

In this section we review some notation and some standard facts about the discrete Fourier transform of boolean functions.

The notation $H(x)$ will be reserved for the binary entropy function $H(x) \equiv -x \log x - (1-x)\log(1-x)$. We use the shorthand $[n]$ for the set $\{1, 2, \ldots, n\}$ and the Iversonian $I[\text{statement}]$ to mean 1 if the statement is true and 0 otherwise. For $a \in \{0,1\}^n$, let a_i denote the i-th bit of a. The vector $e_i \in \{0,1\}^n$ denotes the vector with all zeros except for the i-th bit which is 1. The Hamming weight of a, i.e. the number of ones in a, is denoted by $|a|$.

Let $f : \{0,1\}^n \to \{-1,+1\}$ be a boolean function. Let D be a product distribution over $\{0,1\}^n$ with $\Pr[x_i = 1] = \mu_i$. Thus for $a \in \{0,1\}^n$ we have the distribution of a is $D(a) = \prod_{a_i=1} \mu_i \prod_{a_i=0}(1-\mu_i)$. The distribution D is called *constant bounded* if there exists a constant $c \in (0,1)$ independent of n such that for all i we have $\mu_i \in [c, 1-c]$.

The *influence* of variable x_i on f (see [HM91, KKL88]) over a product distribution D is defined as the probability that $f(x)$ differs from $f(x^{(i)})$ when x is chosen according to D. Here $x^{(i)}$ means x with its i-th bit flipped. We will use the notation $I_{D,i}(f)$ to denote the above probability. Often we will use the restriction notation of functions, $f_0 = f|_{x_i \leftarrow 0}$ and $f_1 = f|_{x_i \leftarrow 1}$. With this notation

$$I_{D,i}(f) = \Pr_D[f_0(y) \neq f_1(y)].$$

Notice that for any boolean function

$$I_{D,i}(f) = \frac{1}{4} E_D[(f_1 - f_0)^2],$$

and for any monotone boolean function

$$I_{D,i}(f) = \frac{1}{2} E_D[f_1 - f_0].$$

The Fourier transform of boolean functions over product distribution is defined as follows (see [FJS91]). First we define the inner product:

$$(f,g)_D = \sum_x D(x) f(x) g(x) = E_D[fg].$$

Now let $z_i(x) = (\mu_i - x_i)/\sigma_i$ where $\sigma_i^2 = Var[x_i] = \mu_i(1-\mu_i)$. Note that z_i has mean zero and variance one (i.e. it is standard normal). Next we define the basis function

$$\phi_a(x) = \prod_{a_i=1} z_i(x).$$

These basis functions satisfy the following properties.

1. *decomposable :* $\phi_{ab}(xy) = \phi_a(x)\phi_b(y)$.
2. *orthonormal :*

$$(\phi_a, \phi_b)_D = \begin{cases} 1 & \text{if } a \neq b \\ 0 & \text{otherwise} \end{cases}$$

Given the orthonormality of ϕ_a we get the Fourier representation of any boolean function f as

$$f = \sum_a \tilde{f}(a)\phi_a,$$

where $\tilde{f}(a) = (f, \phi_a)_D = E_D[f\phi_a]$. Also because of orthonormality we have Parseval's equation:

$$1 = E_D[f^2] = \sum_a \tilde{f}^2(a).$$

Finally note $\tilde{f}(0_n) = E_D[f]$, where 0_n is the n-bit all-zero vector.

For the case of D being the uniform distribution, the following notations are used: χ_a in place of ϕ_a and $\hat{f}(a)$ in place of $\tilde{f}(a)$. Note that in this case, $\mu_i = \sigma_i = 1/2$, for all i.

To facilitate stating our main results we introduce the following notion of *influence norm* $I_D(f)$ of f with respect to a product distribution D:

$$I_D(f) = \sqrt{\sum_{i=1}^{n} (2\sigma_i I_{D,i}(f))^2}.$$

We will observe that in some sense the influence norm of a function determines the algorithmic learning complexity of the standard low-degree Fourier algorithm of Linial et al [LMN93].

4 Spectral lemmas

We are now ready to a lemma which relates the influence and the Fourier transform of boolean functions. The next lemma is a folklore result whose proof we include for completeness.

Lemma 1 *For any boolean function f, for any product distribution D and for any $i \in [n]$,*

$$4\sigma_i^2 I_{D,i}(f) = \sum_{a:a_i=1} \tilde{f}^2(a).$$

Proof Without loss of generality, let $i = 1$. First recall that $I_{D,1}(f) = \frac{1}{4}E_D[(f_0 - f_1)^2]$. Applying Parseval to the right hand side gives

$$I_{D,1}(f) = \frac{1}{4} \sum_{b \in \{0,1\}^{n-1}} \widetilde{(f_0 - f_1)}^2(b) = \frac{1}{4} \sum_{b \in \{0,1\}^{n-1}} (\tilde{f}_0(b) - \tilde{f}_1(b))^2.$$

We now find some relation for $\tilde{f}_0$ and $\tilde{f}_1$. Recall that

$$f(x) = \sum_a \tilde{f}(a)\phi_a(x) = \sum_{0b} \tilde{f}(0b)\phi_b(y) + \sum_{1b} \tilde{f}(1b)\phi_b(y)\frac{\mu_1 - x_1}{\sigma_1}.$$

The last step uses the decomposable property of ϕ_a. From this we will get

$$f_0 \equiv f|_{x_1 \leftarrow 0} = \sum_b \left(\tilde{f}(0b) + \frac{\mu_1}{\sigma_1}\tilde{f}(1b)\right)\phi_b(y),$$

and

$$f_1 \equiv f|_{x_1 \leftarrow 1} = \sum_b \left(\tilde{f}(0b) - \frac{(1-\mu_1)}{\sigma_1}\tilde{f}(1b)\right)\phi_b(y).$$

This implies

$$\tilde{f}_0(b) = \tilde{f}(0b) + \frac{\mu_1}{\sigma_1}\tilde{f}(1b), \quad \text{and} \quad \tilde{f}_1(b) = \tilde{f}(0b) - \frac{(1-\mu_1)}{\sigma_1}\tilde{f}(1b).$$

So continuing with $I_{D,1}(f)$.

$$\begin{aligned} I_{D,1}(f) &= \frac{1}{4}\sum_b (\tilde{f}_0(b) - \tilde{f}_1(b))^2 \\ &= \frac{1}{4}\sum_b \left(\tilde{f}(0b) + \frac{\mu_1}{\sigma_1}\tilde{f}(1b) - \tilde{f}(0b) + \frac{(1-\mu_1)}{\sigma_1}\tilde{f}(1b)\right)^2 \\ &= \frac{1}{4}\sum_b \left(\frac{\mu_1 + (1-\mu_1)}{\sigma_1}\right)^2 \tilde{f}^2(1b) = \sum_b \left(\frac{\tilde{f}(1b)}{2\sigma_1}\right)^2. \end{aligned}$$

□

Let D be a product distribution. We define the *weight* of $a \in \{0,1\}^n$ under D to be

$$||a|| = \log \prod_{a_i=1} \frac{1}{\sigma_i}.$$

Note that $||a|| = |a|$ under the uniform distribution and that $||a|| > |a|$ for any (nonuniform) product distribution.

Theorem 4.1 *For any product distribution D, for any boolean function f,*

$$\begin{aligned} \sum_{||a|| \geq k} \tilde{f}^2(a) &\leq \frac{2}{k}\, I_D(f)\, \sqrt{\sum_{i=1}^n \left(\sigma_i \log \frac{1}{\sigma_i}\right)^2} \\ &\leq 1.062\, \frac{\sqrt{n}}{k}\, I_D(f). \end{aligned}$$

Proof Note that from Lemma 1

$$\begin{aligned} \sum_{i=1}^n 4\sigma_i^2 I_{D,i}(f) \log \sigma_i^{-1} &= \sum_{i=1}^n \sum_{a:a_i=1} \log \sigma_i^{-1} \tilde{f}^2(a) \\ &= \sum_{a \in \{0,1\}^n} \tilde{f}^2(a) \sum_{i:a_i=1} \log \sigma_i^{-1} \\ &= \sum_{a \in \{0,1\}^n} ||a|| \tilde{f}^2(a). \end{aligned}$$

Recall that the Cauchy-Schwartz inequality is

$$\left(\sum_{i=1}^{n} a_i b_i\right)^2 \leq \left(\sum_{i=1}^{n} a_i^2\right)\left(\sum_{i=1}^{n} b_i^2\right),$$

and now we let $a_i = 2\sigma_i I_{D,i}(f)$ and $b_i = 2\sigma_i \log \sigma_i^{-1}$ to get

$$\begin{aligned} I_D(f)^2 &= \sum_{i=1}^{n} 4\sigma_i^2 I_{D,i}(f)^2 \\ &\geq \frac{\left(\sum_{i=1}^{n} 4\sigma_i^2 I_{D,i}(f)\log\sigma_i^{-1}\right)^2}{4\sum_{i=1}^{n}\left(\sigma_i \log\sigma_i^{-1}\right)^2}, \quad \text{by Cauchy-Schwartz} \\ &= \frac{1}{4\sum_{i=1}^{n}\left(\sigma_i\log\sigma_i^{-1}\right)^2}\left(\sum_a ||a||\tilde{f}^2(a)\right)^2 \\ &\geq \frac{1}{4\sum_{i=1}^{n}\left(\sigma_i\log\sigma_i^{-1}\right)^2}\left(k\sum_{||a||\geq k}\tilde{f}^2(a)\right)^2 \end{aligned}$$

which proves the first inequality. The second inequality can be seen using simple calculus since $\left(x\log x^{-1}\right)^2 \leq e^{-2}\log^2 e \leq 0.2817$ for all $x \in [0, 1/2]$. □

5 Learning boolean functions

We recall the following connection between Fourier transform and learning (see [M94]).

Fact 1 *Let D be a product distribution and let $f \in \{-1,+1\}$ be a boolean function. Suppose that for some $A \subset \{0,1\}^n$ the function g satisfies $g = \sum_{a\in A}\tilde{f}(a)\phi_a$. Then*

$$\Pr_D[f \neq sgn(g)] \leq E_D[(f-g)^2] = \sum_{a\notin A}\tilde{f}^2(a).$$

Using this fact, boolean functions can be learned by collecting the Fourier coefficients $\tilde{f}(a)$ for all $a \in A$. This can be done by finding an $\sqrt{\epsilon/(2|A|)}$-approximation h_a of $E_D[f\phi_a] = \tilde{f}(a)$ with confidence $1-\delta/|A|$, for every $a \in A$. Then we define the hypothesis $h = \sum_{a\in A} h_a\phi_a$. This hypothesis will be an $(\sum_{a\notin A}\tilde{f}^2(a)+\frac{\epsilon}{2})$-approximation of f with probability at least $1-\delta$ (see [M94]). Now if $\sum_{a\notin A}\tilde{f}^2(a) \leq \epsilon/2$ then the hypothesis h is an ϵ-approximation to f. To approximate the Fourier coefficients we use sampling to find $E[f\phi_a]$ for all $a \in A$. By the Hoeffding bound, if $|f\phi_a| = |\phi_a| \leq B$ then we will need a sample of size at least

$$\frac{4B^2|A|}{\epsilon}\ln\frac{|A|}{\delta}. \tag{1}$$

When A is the set of all assignments of Hamming weight less than or equal to k, the above algorithm is called the k-lowdegree Fourier algorithm. In this case

$$|A| = \sum_{i=0}^{k}\binom{n}{i} \leq \left(\frac{ne}{k}\right)^k. \tag{2}$$

The following theorem shows that the influence norm can be used to prove PAC learnability of boolean functions under product distributions.

Theorem 5.1 *For any $\epsilon, \delta > 0$, any boolean function f is PAC learnable under any product distribution with error ϵ and confidence $1-\delta$ in time*

$$2^{O\left(\frac{1}{\epsilon}\sqrt{n}I_D(f)\ln\frac{\sqrt{n}\epsilon}{I_D(f)}\right)}\log\frac{1}{\delta}.$$

Proof In Theorem 4.1, to get an error of at most $\epsilon/2$ using the hypothesis $h \equiv \sum_{||a||\leq k} h_a\phi_a$ we must have $k = 2 \times 1.062 I_D(f)\sqrt{n}/\epsilon$. Since $||a|| \geq |a|$, we have that $\{a : ||a|| \leq k\} \subseteq \{a : |a| \leq k\}$ and hence we can use the k-lowdegree Fourier algorithm. From the definitions of $||a||$ and ϕ_a we get that

$$\begin{aligned} |\phi_a| &= \left|\prod_{a_i=1}\frac{\mu_i - x_i}{\sigma_i}\right| \\ &= 2^{||a||}\left|\prod_{a_i=1}(\mu_i - x_i)\right| \\ &\leq 2^k = B. \end{aligned}$$

We may assume without loss of generality that all the μ_i's are known (via sampling). Now by (1),(2) and the above, the algorithm outputs a hypothesis that is an ϵ-approximation of f with sample size and time complexity of

$$2^{O\left(\frac{\sqrt{n}I_D(f)}{\epsilon}\ln\frac{\sqrt{n}\epsilon}{I_D(f)}\right)}\log\frac{1}{\delta}. \tag{3}$$

□

6 Learning monotone functions

In this section we apply the previous results to learning monotone boolean functions and prove several lower bounds.

We first establish a connection between the influence and the Fourier coefficients at the unit vectors.

Lemma 2 *For any monotone boolean function f, for any product distribution D,*

$$\tilde{f}(e_i) = -2\sigma_i I_{D,i}(f).$$

Proof Let D_i be the induced distribution over all the variables except x_i. We have the following derivation.

$$\begin{aligned} \tilde{f}(e_i) &= E_D[f\phi_{e_i}] \\ &= E_{D_i}E_{x_i}[fz_i] \\ &= E_{D_i}\left[(1-\mu_i)\frac{\mu_i}{\sigma_i}f_0 + \mu_i\frac{\mu_i - 1}{\sigma_i}f_1\right] \\ &= E_{D_i}\left[\frac{\mu_i(1-\mu_i)}{\sigma_i}\ (f_0 - f_1)\right] \\ &= \sigma_i E_{D_i}[f_0 - f_1]. \end{aligned}$$

Now recall that for monotone boolean functions $2I_{D,i}(f) = E_{D_i}[f_1 - f_0]$. □

Theorem 6.1 *For any $\epsilon, \delta > 0$, any monotone boolean function is PAC learnable under any product distribution with error ϵ and confidence $1 - \delta$ in time*

$$2^{O(\frac{1}{\epsilon}\sqrt{n}\log(\epsilon\sqrt{n}))}.$$

Proof By Lemma 2, we note that $I_D(f) \leq 1$ for any monotone function f, because $\sum_i (2\sigma_i I_{D,i}(f))^2 = \sum_i \tilde{f}(e_i)^2 \leq 1$ (by Parseval's). Now we use Theorem 5.1. Since $I_D(f) \leq 1$ we have $I_D(f) \log \frac{1}{I_D(f)} \leq 1$, and therefore

$$\begin{aligned} I_D(f) \log \frac{\epsilon\sqrt{n}}{I_D(f)} &= I_D(f) \log \frac{1}{I_D(f)} + I_D(f) \log(\epsilon\sqrt{n}) \\ &= O(\log(\epsilon\sqrt{n})). \end{aligned}$$

□

Remark. Note that using our algorithm with subexponential time, the best achievable error rate is $\epsilon = \frac{1}{\sqrt{n}}$. In the next theorem we show that this is the best possible error rate up to a $O(\log^2 n)$ factor.

Theorem 6.2 *Any algorithm which approximates any monotone boolean function under the uniform distribution and which runs in subexponential time (even with time 2^{cn}, for any $c < 1$) will output an approximation with an error of at least $\Omega\left(\frac{1}{\sqrt{n}\log n}\right)$.*

Proof There are at least $m(n) = 2^{\binom{n}{n/2}} \geq 2^{d2^n\sqrt{n}}$, monotone boolean functions over n variables, for some constant $d < 1$ (see [W87]). Suppose A is the ϵ-approximation algorithm for any monotone boolean function. If A outputs a hypothesis h then h can ϵ-approximate at most

$$k(n) = \sum_{i \leq \epsilon 2^n} \binom{2^n}{i} \leq 2^{2^n \epsilon \log(e/\epsilon)}$$

boolean functions. Assuming A runs in time 2^{cn}, for some constant $c < 1$, then A can output at most $2^{2^{cn}}$ possible hypotheses. Therefore we must have $2^{2^{cn}} k(n) \geq m(n)$ which implies

$$2^{cn} + \epsilon 2^n \log \frac{e}{\epsilon} \geq \frac{d2^n}{\sqrt{n}}.$$

This implies $\epsilon = \Omega\left(\frac{1}{\sqrt{n}\log n}\right)$. □

The next theorem shows that there is a specific monotone function for which the low-degree algorithm requires subexponential time to obtain an error that is at most $O(1/\sqrt{n})$.

Theorem 6.3 *The majority function f satisfies $\sum_{|a| \geq \Omega(\sqrt{n})} \hat{f}^2(a) \geq \Omega(1/\sqrt{n})$.*

Proof Let $f = MAJ(x) = I[\sum_{i=1}^n x_i \geq n/2]$. Since f is a symmetric function, the influence of all variables are equal. From the first equality in the proof of Theorem 4.1 we have

$$\sum_a |a| \hat{f}^2(a) = \sum_{i=1}^n I_i(f) = n I_1(f).$$

To get a bound on $I_1(MAJ)$, note that

$$I_1(MAJ) \geq 2^{-n}\binom{n}{n/2} \geq \frac{c}{\sqrt{n}},$$

for some constant c.

$$\begin{aligned} c\sqrt{n} &\leq \sum_a |a|\hat{f}(a)^2 \\ &= \sum_{|a|\geq \frac{c}{2}\sqrt{n}} |a|\hat{f}(a)^2 + \sum_{|a|<\frac{c}{2}\sqrt{n}} |a|\hat{f}(a)^2 \\ &\leq n \sum_{|a|\geq \frac{c}{2}\sqrt{n}} \hat{f}(a)^2 + \frac{c}{2}\sqrt{n}. \end{aligned}$$

Thus we have

$$\sum_{|a|\geq \frac{c}{2}\sqrt{n}} \hat{f}(a)^2 \geq \frac{c}{2\sqrt{n}}.$$

□

The next theorem proves the existence of a monotone function whose power spectrum is highly concentrated in the $\Theta(n)$ region.

Theorem 6.4 *For any constant $c < 1$ there is a monotone boolean function f which satisfies*

$$\sum_{|a|\geq cn} \hat{f}^2(a) \geq \Omega\left(\frac{1}{\sqrt{n}\log n}\right).$$

Proof Assume for contradiction that there is some constant $c < 1$ such that for any monotone function f

$$\sum_{|a|\geq cn} \hat{f}^2(a) \leq O\left(\frac{1}{\sqrt{n}\log n}\right).$$

But this implies that the low-degree algorithm which searches all coefficients of degree at most cn will approximate f within an error of $O(1/(\sqrt{n}\log n))$. This contradicts Theorem 6.2 modulo constant factors. □

In the following we consider some lower bounds in the learning complexity for monotone boolean functions. First, we note an application of the Vapnik-Chernovenkis dimension. Recall that if C is a class of boolean functions then C shatters $A \subseteq \{0,1\}^n$ if for every boolean function $g : A \to \{0,1\}$ there exists a boolean function $f \in C$ such that $f|_A = g$. The *Vapnik-Chernovenkis* dimension of C, $VCdim(C)$, is the cardinality of the largest subset A which is shattered by C. Ehrenfeucht *et al.* [EHKV88] proved a sample complexity lower bound of

$$\Omega\left(\frac{1}{\epsilon}\ln\frac{1}{\delta} + \frac{1}{\epsilon}VCdim(C)\right)$$

for PAC learning any class C with error ϵ and confidence δ. It is easy to see that the VC-dimension of monotone functions is at least $\binom{n}{n/2} \sim 2^n/\sqrt{n}$. Hence we get the following easy corollary.

Corollary 1 *Any PAC learning algorithm for monotone boolean functions under an arbitrary distribution with error ϵ requires at least $\Omega\left(\frac{2^n}{\epsilon\sqrt{n}} + \frac{1}{\epsilon}\ln\frac{1}{\delta}\right)$ examples.*

For learning under the uniform distribution we observe the following. Note that Theorem 6.1 a subexponential PAC learning as long as the error ϵ satisfies $\epsilon > n^{-1/2}$. We wish to show that for $\epsilon \ll n^{-1/2}$ we must allow exponential running time.

Theorem 6.5 *Any PAC learning algorithm for monotone boolean functions under the uniform distribution with error $\epsilon \ll n^{-1/2}$ requires a running time of at least $T(n) = \frac{\epsilon 2^n}{\sqrt{n}}$.*

Proof Omitted. □

Theorem 6.6 *Any learning algorithm for monotone functions with a running time bounded by $2^{\sqrt{n}}$ cannot achieve an error smaller than $\tilde{\Omega}(n^{-1/4})$.*

Proof Let $f(n)$ be the smallest achievable error for learning monotone functions over n variables. We note that with $n^{2-\epsilon}$ variables, the allowable running time is $2^{n^{1-\epsilon/2}}$ (which is subexponential). So by Theorem 6.2, $f(n^{2-\epsilon}) > \tilde{\Omega}(n^{-1/2})$. Hence, by substituting n for $n^{2-\epsilon}$, we get that $f(n) > \tilde{\Omega}(n^{-1/4})$. □

7 Approximating Monotone Boolean Functions with Circuits

In this section we study the circuit complexity of approximating monotone boolean functions using boolean circuits (non-monotone). We show that any monotone boolean function can be approximated by a non-monotone boolean circuit of subexponential size and sublinear depth. This result is a consequence of Theorem 4.1.

Theorem 7.1 *For any monotone boolean function f on n variables and for any constant $\epsilon > 0$, there is a boolean circuit of size $2^{O(\frac{1}{\epsilon}\sqrt{n}\log n)}$ and depth $\tilde{O}(\sqrt{n})$ which approximates f to within ϵ.*

Proof Note that the low-degree algorithm outputs a hypothesis

$$h(x) \equiv \sum_{|a| \leq O(\frac{1}{\epsilon}\sqrt{n})} c_a \chi_a(x),$$

where $c_a \sim E[f\phi_a]$. Note that each c_a can be at most $2^{O(\frac{1}{\epsilon}\sqrt{n})}$ bits. So essentially we need to add m number each being m bits, where $m(n) = 2^{O(\frac{1}{\epsilon}\sqrt{n})}$. This problem is doable in bounded fan-in logarithmic depth and polynomial size in the size of the input (i.e. NC^1, see [BCP83]), and hence the claim. □

8 Convexity results

In this section we describe two results with a convexity flavour. The first one states that learnability over a collection of distributions $\{D_i\}$ implies the learnability over any distribution in the convex space of $\{D_i\}$. The second result claims that the weak learnability of functions which are convex combinations of monotone boolean functions.

In the following we introduce the notion of a *convex dimension* of a distribution in terms of product distributions.

Definition 1 *Let $\mathcal{D} = \{D_i\}_{i \in I}$ be the set of all product distributions over $\{0,1\}^n$. Consider a distribution D of the form*

$$D = \sum_{i=1}^{m} \lambda_i D_i,$$

where each $\lambda_i \in [0,1]$ and $\sum_{i=1}^{m} \lambda_i = 1$. We call D the convex linear combination of distributions $D_1, \ldots, D_m$. The convex dimension $cdim(D)$ of D is the least m such that D can be represented as a convex linear combination of m product distributions.

Fact 2 *Any distribution D is a convex linear combination of at most 2^{n-1} product distributions. The convex dimension of any distribution is at least 1 and at most 2^n.*

Lemma 3 *Let $\tau_1, \ldots, \tau_m \in \{-1, 1\}$ and $d_1, \ldots, d_m \in R^+$. If $\sum_{i=1}^{m} d_i \tau_i < 0$ then*

$$\sum_{\tau_i = -1} d_i \geq \frac{1}{2} \sum_{i=1} d_i.$$

Proof Since $\sum_{i=1}^{m} d_i \tau_i = \sum_{\tau_i=1} d_i - \sum_{\tau_i=-1} d_i \leq 0$, we have $\sum_{\tau_i=-1} d_i \geq \sum_{\tau_i=1} d_i$. If $\sum_{\tau_i=-1} d_i < \frac{1}{2}(d_1 + \cdots + d_m)$ then $\sum_{\tau_i=1} d_i < \frac{1}{2}(d_1 + \cdots + d_m)$ and

$$d_1 + \cdots + d_m = \sum_{\tau_i=1} d_i + \sum_{\tau_i=-1} d_i < d_1 + \cdots + d_m.$$

□

Theorem 8.1 *Let C be a class that is PAC learnable over the set $\{D_i\}_{i=1}^{m}$ of distributions. Then C is learnable over any distribution that is a convex linear combination of the D_i's.*

Proof Suppose $\sum_{i=1}^{m} \lambda_i = 1$ and $D = \sum_{i=1}^{m} \lambda_i D_i$. Let h_i be a hypothesis over distribution D_i satisfying $\Pr_{D_i}[h_i(x) \neq f(x)] \leq \epsilon/2$. Define the hypothesis $H \in \{-1, +1\}$ over D to be

$$H(x) \equiv sign\left(\sum_{i=1}^{m} \lambda_i D_i(x) h_i(x)\right).$$

Define an indicator random variable $A(x) = I[H(x) = +1]$. Then

$$\begin{aligned}
\Pr_D[H(x) \neq f(x)] &= \Pr_D[\sum_{i=1}^{m} \lambda_i D_i(x) h_i(x) f(x) < 0] \\
&= \sum_{x:A(x)} \sum_{i=1}^{m} \lambda_i D_i(x) \\
&\leq \sum_{x:A(x)} \sum_{i:h_i(x) \neq f(x)} 2\lambda_i D_i(x), \text{ by Lemma 3} \\
&\leq 2 \sum_{x} \sum_{i:h_i(x) \neq f(x)} \lambda_i D_i(x) \\
&= 2 \sum_{i=1}^{m} \sum_{x:h_i \neq f(x)} \lambda_i D_i(x)
\end{aligned}$$

Therefore, $\Pr_D[H(x) \neq f(x)] \leq \epsilon$. □

Remark. We note that it is enough for a distribution D to be *close* to another distribution with *small* convex dimension to afford a learnability result.

Theorem 8.2 *Let $\{g_i\}_{i\in I}$ be a collection of monotone boolean functions and suppose $\sum_{i\in I}\lambda_i = 1$ with $\lambda_i \geq 0$ for all $i \in I$. Then the function $g(x) = \sum_{i\in I}\lambda_i g_i(x)$, is weakly learnable under any product distribution.*

Proof We prove the result for uniform distribution since the proof is similar. Note that by linearity of the Fourier transform we have $\hat{g}(a) = \sum_{i\in I}\lambda_i\hat{f}_i(a)$. To get a handle on $\sum_{j=1}^n \hat{g}(e_j)^2$ we consider $\sum_{j=1}^n \hat{g}(e_j)$ since by Cauchy-Schwartz we have $\sum_{j=1}^n \hat{g}(e_j)^2 \geq \frac{1}{n}(\sum_{j=1}^n \hat{g}(e_j))^2$. So we have

$$\begin{aligned}\sum_{j=1}^n \hat{g}(e_j) &= \sum_{j=1}^n\sum_{i\in I}\lambda_i\hat{f}_i(e_j) = \sum_{i\in I}\lambda_i\sum_{j=1}^n \hat{f}_i(e_j)\\ &= -\sum_{i\in I}\lambda_i\sum_{j=1}^n I_j(f_i) = -\sum_{i\in I}\lambda_i\sum_{j=1}^n\sum_{a:a_j=1}\hat{f}_i(a)^2.\end{aligned}$$

Taking absolute values will allow us to lower bound the above by

$$\begin{aligned}\sum_{i\in I}\lambda_i\sum_a |a|\hat{f}_i(a)^2 &= \sum_a |a|\sum_{i\in I}\lambda_i\hat{f}_i(a)^2\\ &\geq \sum_{j=1}^n\sum_{i\in I}\lambda_i\hat{f}_i(e_j)^2 = \sum_{i\in I}\lambda_i\sum_{j=1}^n \hat{f}_i(e_j)^2.\end{aligned}$$

Now since each f_i is monotone, $\sum_{j=1}^n \hat{f}_i(e_j)^2$ is *large* and hence $\sum_{j=1}^n \hat{g}(e_j)$ is large. □

Remark. Previously Aiello and Mihail [AM91] has considered the problem of learning a convex mixture of functions with bounded spectrum. It is interesting to note that while they require an upper bound on the low-order spectrum, we require a lower bound on the spectrum of the unit vectors.

9 Learning in polynomial time

Kearns and Valiant [KV89] proved that monotone boolean functions are weakly learnable under the uniform distribution with error $1/2 - 1/cn$, for some constant $c > 1$. In the following we manage to improve slightly upon their result. In particular, there is a weak learner with error $1/2 - \log^2 n/n$ in the uniform distribution and there are weak learners with error $1/2 - c/n$, for any constant c, under any product distribution. For learning under the uniform distribution we will use the following result of Kahn *et al.*

Lemma 4 *[KKL88] Let $f \in \{0,1\}$ be a boolean function with $p = \Pr[f(x) = 1] \leq 1/2$. Then $\sum_{i=1}^n I_i(f)^2 \geq cp^2(\log n)^2/n$, where c is an absolute constant ($c = 1/5$ works).*

Remark. The restriction that $\Pr[f(x) = 1] \leq 1/2$ is not a problem by a result of Ben-Or and Linial [BL89]. They proved that for any boolean function f there is a monotone boolean function g such that $\Pr[f(x) = 1] = \Pr[g(x) = 1]$ and $I_i(f) \geq I_i(g)$, for all $i \in [n]$. So given a monotone function f with $\Pr[f(x) = 1] \geq 1/2$, we consider its negation $g \equiv \neg f$ and apply Ben-Or and Linial's result to g.

Theorem 9.1 *There is a $(1/2 + \log^2 n/n)$ weak PAC learner for any monotone boolean function under the uniform distribution.*

Proof Since $I_i(f)^2 = \hat{f}^2(e_i)$ and using Lemma 4, $\sum_{i=1}^{n} \hat{f}^2(e_i) \geq cp^2 \log^2 n/n$. Combining this with Fact 1 with $A = \{e_i : i \in [n]\}$, we get a weak learner with the claimed accuracy. □

Theorem 9.2 *For any constant k there is a weak approximator with error $\epsilon = 1/2 - k/32n$ for monotone boolean functions under any product distribution.*

Proof By Theorem 4.1, if $\sum_{||a||\geq k} \tilde{f}^2(a) \leq 1/2$ then we get a 1/4-approximator by the standard low-degree algorithm (using again the fact that $||a|| \geq |a|$) and after appealing to the randomizing hypothesis in [BFJ+]. Otherwise $\sum_i \tilde{f}^2(e_i) \geq \frac{1}{n}k/16$. In this case we get a $(1/2 - k/32n)$-approximator. □

A variable x_i is *relevant* for f if there are $a, b \in \{0,1\}^n$ with $a = b \oplus e_i$ (here $\oplus$ means the bitwise exclusive OR of vectors) and $f(a) \neq f(b)$. Note that x_i is relevant if and only if $I_i(f) > 0$. The next theorem is a generalization of the recent result of Sakai and Maruoka [SM94]. Let $A(k)$ be the class of functions of the form $f(T_1, \ldots, T_k)$, where f is an arbitrary monotone function on k inputs and each T_i is a monotone conjunction or disjunction over n variables.

Theorem 9.3 *The class $A(\log n)$ is PAC learnable under constant bounded product distributions.*

Proof We prove the claim for DNF under the uniform distribution since the proof for the general case is almost identical.

Let f be a $(c \log n)$-term DNF. We build a $(4c \log n)$-depth decision tree T which contains relevant variables. The first relevant variable, say x_i, is found by searching a parity on a single variable with high correlation to f. This will be the root of the decision tree. Its subtrees are constructed recursively by applying the same idea to $f|_{x_i \leftarrow 0}$ and $f|_{x_i \leftarrow 1}$.

Since each variable appearing in T has positive influence, they all appear in some term of f. For a uniformly chosen $a \in \{0,1\}^n$, with probability 1/2, a walk down eliminates a term of f. So after $t = 4c \log n$ steps, by Chernoff bounds

$$\Pr[f \neq \text{ constant}] \leq \exp(-t/64) = n^{-\alpha},$$

for some $\alpha > 0$. This implies that after depth $4c \log n$, f has no more relevant variables since it is constant.

The tree T has size at most $s = n^{4c}$. So given $\epsilon, \delta > 0$, we search for an relevant variable with accuracy ϵ/s and confidence $1 - \delta/s$. In total we suffer only a final error of ϵ with probability at least $1 - \delta$. □

Remark. The proof of the above previous theorem can be compared to Quinlan's induction method of building decision trees [Q86].

Theorem 9.4 *For any constant ϵ, the class of monotone functions which depend on $O\left(\frac{\log^2 n}{\log^2 \log n}\right)$ variables is PAC learnable with error ϵ under constant bounded product distributions.*

Proof By the assumption, there are at most $m(n) = \frac{\log^2 n}{\log^2 \log n}$ variables having nonzero influence. Note that we may ignore all the variables with zero influence since we are dealing with constant bounded product distributions. So we can apply the low-degree algorithm which will run in time $2^{\sqrt{m(n) \log m(n)}} = n^{O(1)}$, modulo the constant ϵ. □

Theorem 9.5 *For any constant ϵ, the class $A\left(\frac{\log^2 n}{\log^3 \log n}\right)$ is PAC learnable with error ϵ under constant bounded product distributions.*

Proof First, we claim that any term with size $\Omega(\log\log n)$ may be ignored without incurring an error of more than $O(1)$. Now observe that with this simplification, there are at most $\log^2 n/(\log\log n)^2$ variables. This problem reduces to Theorem 9.4. □

Remark. The learning algorithms discussed so far fit into the *statistical query* learning model [K93]. Hence by Kearns' results, these algorithms are robust agains classification noise in the example oracle.

Remark. So far we have considered monotone boolean functions. The results actually hold for monotone functions ranging over a linear order. We only sketch the idea in the following. Let $L = \{a_0 < a_1 < \ldots < a_{m-1}\}$ be a linear order of size m and let $f : \{0,1\}^n \to L$ be a monotone function over L. For $i = 1, 2, \ldots, l$, where $l = \lceil \log m \rceil$, let $b_1, b_2, \ldots, b_i \in \{0,1\}$ be bit values. Set $t = (\sum_{j=1}^{i} b_j 2^{-j} + 2^{-(i+1)})m$. We will use the notation $bin(i)$ to denote the binary representation of i. For $i \geq 0$, define inductively the following set of boolean functions

$$f_{b_1 b_2 \ldots b_i 0}(x) = sign\,[f_{b_1 b_2 \ldots b_i}(x) = +1 \text{ and } f(x) \leq a_t]$$

$$f_{b_1 b_2 \ldots b_i 1}(x) = sign\,[f_{b_1 b_2 \ldots b_i}(x) = +1 \text{ and } f(x) > a_t]$$

For the base case, let $f_\lambda(x) \equiv +1$ be the constant function $+1$, where λ is the empty string. Now note the following disjoint sum.

$$f(x) = \sum_{i=1}^{m} a_i I[f_{bin(i/m)}(x) = +1].$$

So to learn f we learn each boolean function f_α, for all $\alpha \in \{0,1\}^{\leq l}$ (for which there are at most $2m$ of them). Since each f_α is boolean we can apply the low-degree algorithm with error $\epsilon/\log m$ and confidence $\delta/2m$ to get a total error of ϵ and a total confidence of δ by the union bound.

10 Monotone graph properties

In this section we consider some monotone graph properties on the random graph $G(n,p)$, where n is the number of vertices of G and p is the edge existence probability. Some well-known graph properties are monotone: the clique function $CLIQUE_k^n$ which is one iff the graph has a clique of size at least k, the hamiltonicity function HAM_n which is one iff the graph has a Hamiltonian cycle, the planarity function $PLANAR_n$ which is one iff the graph is planar etc. We investigate the problem of learning these monotone graph-theoretic functions.

We adopt the *probabilistic* model of the random graph on n vertices (see [S94] for other models). The random graph $G = G(n,p)$ is a probability distribution on the edges of the complete graph K_n on n vertices, where each edge exists independently with probability $p \in [0,1]$. A boolean function f on the edge set $E(G)$ is called a *monotone graph property* if f is a monotone (or antimonotone) boolean function over $E(G)$.

Any monotone graph property exhibits a *threshold phenomena* (see [B85, S94]). Let f be a monotone graph property on $G(n,p)$. A function $r(n)$ is called a *threshold function* for f if it satisfies

1. if $\lim_{n\to\infty} \frac{p(n)}{r(n)} = 0$ then $\lim_{n\to\infty} \Pr[f(G(n,p)) = 1] = 0$.

2. if $\lim_{n\to\infty} \frac{p(n)}{r(n)} = 1$ then $\lim_{n\to\infty} \Pr[f(G(n,p)) = 1] = 1$.

Theorem 10.1 *Let f be a monotone graph property with a threshold function of $r(n)$ (the number of inputs to f is $m = \binom{n}{2}$). Then there is an algorithm which approximates f to within error ϵ and which runs in time $2^{\tilde{O}(\frac{1}{\epsilon}\sqrt{mr(n)})}$. The error probability here is with respect to inputs $x \in \{0,1\}^m$ of f generated according to $G(n,p)$ (p might depend on n).*

Remark. In the statement in the above theorem we have abused the $\tilde{O}$ notation. The exponent actually contains a factor of $O(\log n)$ which should remain even when $r(n) = o(n)$.

Proof Note first that if $\lim_{n\to\infty} p(n)/r(n) = 0$ then the function f is either 0 or 1. So we consider only cases where $\lim_{n\to\infty} p(n)/r(n) = 1$.

The distribution $G(n,r)$ is a product distribution on the inputs of f. In particular, using the terminology from the previous sections, we have $\mu_i = r$ for all edges $i \in [m]$. From Theorem 4.1 we get that

$$\sum_{||a||\geq k} \tilde{f}^2(a) \leq \frac{2\sqrt{m}}{k}\sigma|\log\sigma|,$$

where $\sigma^2 = r(1-r)$. If $\lim r =$ constant then the claim is trivially true. So suppose $\lim_{n\to\infty} r(n) = 0$ or 1. Then we may assume that $\sqrt{r(1-r)} \sim \sqrt{r}$ and ignore the logarithmic factors. Hence to obtain an error of ϵ, we set $k = \frac{2\sqrt{mr}}{\epsilon}$. This implies the claim modulo logarithmic factors. □

In a similar manner as in Theorem 7.1 we know that there exists a circuit of size $2^{\tilde{O}(\frac{1}{\epsilon}\sqrt{mr(n)})}$ and depth $\tilde{O}(\frac{1}{\epsilon}\sqrt{mr(n)})$ which approximates any monotone graph property with threshold $r(n)$ under the random graph distribution $G(n, r(n))$.

Next we consider two monotone graph properties which are *NP*-complete: $CLIQUE(n,k)$ and $HAM(n)$. The clique function has a threshold of $r(n) = n^{-2/(k-1)}$ while the Hamiltonicity function has a threshold of $r(n) = \ln n/n$.

Corollary 2 *For any ϵ, there are algorithms which approximates $CLIQUE_k^n$ and HAM_n to within error ϵ and runs in time $2^{\tilde{O}(\frac{1}{\epsilon}n^{1-1/(k-1)})}$ and $2^{\tilde{O}(\frac{1}{\epsilon}\sqrt{n\ln n})}$, respectively.*

11 Acknowledgments

This paper would not have existed without Jeff Jackson telling us about all the *neat* things related to Fourier transform. His enthusiastic lectures during his visit at the Department of Computer Science, University of Calgary has more impact on this work than anything else. We wish to thank him for his generosity and helpful comments during our work in this paper.

We thank Dan Boneh for suggesting the notion of influence norm and to Yishay Mansour and Dave Wilson for helpful comments.

References

[AM91] William Aiello and Milena Mihail. Learning the Fourier Spectrum of Probabilistic Lists and Trees. In *Workshop on Computational Learning Theory*, 1991.

[AS92] Noga Alon and Joel Spencer. **The Probabilistic Method**. *Wiley-Interscience*, 1992.

[B85] Béla Bollobás. *Random Graphs*. *Academic Press Inc.*, 1985.

[B92] Mihir Bellare. A Technique for Upper Bounding the Spectral Norm with Applications to Learning. In *Workshop on Computational Learning Theory*, 1992.

[BCP83] Allan Borodin, Stephen Cook, and Nicholas Pippenger. Parallel Computation for Well-Endowed Rings and Space-Bounded Probabilistic Machines. In *Information and Control*, 1993.

[BL89] Michael Ben-Or and Nathan Linial. Collective Coin Flipping. In *Advances in Computing Research*, volume 5, pages 91-115, 1989.

[BFJ+] Avrim Blum, Merrick Furst, Jeffrey Jackson, Michael Kearns, Yishay Mansour, and Steven Rudich. Weakly Learning DNF and Characterizing Statistical Query using Fourier Analysis. In *ACM Symposium on Theory of Computing*, 1994.

[BS92] Jehoshua Bruck and Roman Smolensky. Polynomial Threshold Functions, AC^0 Functions, and Spectral Norms. In *SIAM Journal on Computing*, 1992.

[CKM93] Don Coppersmith, Hugo Krawczyk, and Yishay Mansour. The Shrinking Generator. In *Advances in Cryptology CRYPTO*, 1993.

[EHKV88] Andrzej Ehrenfeucht, David Haussler, Michael Kearns, and Leslie Valiant. A General Lower Bound on the Number of Examples Needed for Learning. In *ACM Workshop on Computational Learning Theory*, 1988.

[FJS91] Merrick Furst, Jeffrey Jackson, and Sean Smith. Improved Learning of AC^0 Functions. In *Workshop on Computational Learning Theory*, 1991.

[HM91] Thomas Hancock and Yishay Mansour. Learning Monotone k-μ DNF Formulas on Product Distributions. In *Workshop on Computational Learning Theory*, 1991.

[J94] Jeffrey Jackson. An Efficient Membership-Query Algorithm for Learning DNF with Respect to the Uniform Distribution. In *IEEE Foundations of Computer Science*, 1994.

[K93] Michael Kearns. Efficient Noise Tolerant Learning from Statistical Queries. In *ACM Symposium on the Theory of Computing*, 1993.

[KKL88] Jeff Kahn, Gil Kalai, and Nathan Linial. The Influence of Variables on Boolean Functions. In *IEEE Foundations of Computer Science*, 1988.

[KM91] Eyal Kushilevitz and Yishay Mansour. Learning Decision Trees using the Fourier Spectrum. In *ACM Symposium on the Theory of Computing*, 1991.

[KV89] Michael Kearns and Leslie Valiant. Cryptographic Limitations on Learning Boolean Formulae and Finite Automata. In *ACM Symposium on the Theory of Computing*, 1989.

[LMN93] Nathan Linial, Yishay Mansour, and Noam Nisan. Constant Depth Circuits, Fourier Transform and Learnability. In *Journal of the ACM*, 1993.

[M92] Yishay Mansour. An $O(n^{\log\log n})$ Learning Algorithm for DNF. In *Workshop on Computational Learning Theory*, 1992.

[M94] Yishay Mansour. Learning Boolean Functions via the Fourier Transform. Tutorial Notes for the *Workshop on Computational Learning Theory*, 1994.

[Q86] J. R. Quinlan. Induction of Decision Trees. In *Machine Learning*, **1**:81-106, 1986.

[R90] Prabhakar Raghavan. Lecture Notes on Randomized Algorithms. *Research Report, IBM Research Division*, RC15340 (#68237), 1/9/90.

[S94] Joel Spencer. *Ten Lectures on the Probabilistic Method. SIAM CBMS-NSF Regional Conference Series in Applied Mathematics*, second edition, 1994.

[SM94] Yoshifumi Sakai and Akira Maruoka. Learning $O(\log n)$-term Monotone DNF. In *ACM Workshop in Computational Learning Theory*, 1994.

[V84] Leslie Valiant. A Theory of the Learnable. In *ACM Symposium on the Theory of Computing*, 1984.

[W87] Ingo Wegener. The Complexity of Boolean Functions. Wiley-Teubner, 1987.