Journal of Machine Learning Research 2 (2002) 359-395 Submitted 10/01; Published 02/02

On Using Extended Statistical Queries to Avoid Membership Queries

Nader H. Bshouty BSHOUTY@CS.TECHNION.AC.IL
Vitaly Feldman FELVIT@TX.TECHNION.AC.IL
Department of Computer Science
Technion , Israel Institute of Technology,
Haifa, 32000, Israel

Editor: Dana Ron

Abstract

The Kushilevitz-Mansour (`KM`) algorithm is an algorithm that finds all the "large" Fourier coefficients of a Boolean function. It is the main tool for learning decision trees and DNF expressions in the PAC model with respect to the uniform distribution. The algorithm requires access to the membership query (MQ) oracle. The access is often unavailable in learning applications and thus the `KM` algorithm cannot be used.

We significantly weaken this requirement by producing an analogue of the `KM` algorithm that uses extended statistical queries (SQ) (SQs in which the expectation is taken with respect to a distribution given by a learning algorithm). We restrict a set of distributions that a learning algorithm may use for its statistical queries to be a set of product distributions with each bit being 1 with probability $\rho, 1/2$ or $1-\rho$ for a constant $1/2 > \rho > 0$ (we denote the resulting model by SQ–$\mathcal{D}_\rho$). Our analogue finds all the "large" Fourier coefficients of degree lower than $c \log n$ (we call it the *Bounded Sieve* (`BS`)). We use `BS` to learn decision trees and by adapting Freund's boosting technique we give an algorithm that learns DNF in SQ–$\mathcal{D}_\rho$. An important property of the model is that its algorithms can be simulated by MQs with persistent noise. With some modifications `BS` can also be simulated by MQs with product attribute noise (i.e., for a query x oracle changes every bit of x with some constant probability and calculates the value of the target function at the resulting point) and classification noise. This implies learnability of decision trees and weak learnability of DNF with this non-trivial noise.

In the second part of this paper we develop a characterization for learnability with these extended statistical queries. We show that our characterization when applied to SQ–$\mathcal{D}_\rho$ is tight in terms of learning parity functions. We extend the result given by Blum et al. by proving that there is a class learnable in the PAC model with random classification noise and not learnable in SQ–$\mathcal{D}_\rho$.

1. Introduction and Overview

The problems of learning decision trees and DNF expressions are among the most well studied problems of computational learning theory. In this paper we address learning of these classes in Valiant's popular PAC model with respect to the uniform distribution.[1] The first algorithm that learns decision trees in this setting was given by Kushilevitz and Mansour [KM91]. The main tool that they used is the beautiful technique due to Goldreich and Levin [GL89]. Given a black box that will answer membership queries for a Boolean function f over $\{0,1\}^n$, their technique efficiently locates a subset A of the inputs of f such that the parity χ_A of the bits in A is a weak approximator for f with respect to the uniform (if such A exists). The correlation of the parity function χ_A is called the Fourier coefficient of A. Using this technique Kushilevitz and Mansour give an algorithm that finds all the Fourier coefficients of a Boolean function larger than a given threshold (it is usually referred as the `KM` algorithm). They prove that this algorithm can be used to efficiently learn the class of decision trees. Later Jackson [Ja94] again used the Kushilevitz-Mansour algorithm and Freund's boosting technique to build his famous algorithm for learning DNF expressions. An important property of Freund's boosting in this context is that it requires weak learning only with respect to distributions that are polynomially close to the uniform.

As we have mentioned the core of both algorithms utilizes *membership query* (MQ) oracle to find weakly approximating parity function. That is, in order to use these algorithms we need access to values of the learned function at any given point. Angluin and Kharitonov [AK91] proved, under standard cryptographic assumptions, that if DNF expressions are learnable in the distribution-independent PAC model with MQs then they are also learnable without MQs, in other words membership queries are not helpful in the distribution-independent PAC learning model. On the other hand, there is no known algorithm that learns the above-mentioned classes in the PAC model with respect to the uniform distribution without the use of MQs.

The access to membership query oracle is not available in most of applications and thus the question of whether it is possible to avoid the use of them is of great theoretical interest and practical importance. We investigate the way to reduce the dependence on MQs for the distribution-specific PAC learning that results from the following approach. Learning in the basic PAC model (without MQs) represents learning without any control over the points in which the value of the target function will be known. On the other hand, membership queries represent the total control over these points. Both of these situations can be seen as (totally) different amount of control over the probability that the next sampled point will be some specific x. This observation suggests that we can represent any intermediate situation by allowing the learning algorithm to choose the distribution with respect to which the points will be sampled. The set of distributions $\mathcal{D}$ that a learning algorithm may choose from measures the amount of control that the learning algorithm has. Naturally, such an extension defines learnability with respect to a given set of distributions $\mathcal{D}$ or learnability in the PAC–$\mathcal{D}$ model with respect to D. In this paper we actually discuss SQ–$\mathcal{D}$—the statistical query (SQ) analogue of the above model. The SQ model introduced by Kearns [Kea93] allows the learning algorithms to get "good" estimates of expectation of any (polynomially computable) function involving the value of $f(x)$ with respect to the

1. This and all other models of learning that are mentioned further are fully defined in the following section.

target distribution D. The major benefit of this restriction of the PAC model is that every algorithm in this model can be automatically transformed into classification noise tolerant algorithm. Analogously, the learning algorithm in SQ–$\mathcal{D}$ is allowed to ask SQs with respect to any distribution in $\mathcal{D}$. This restriction will be sufficient for the learning algorithms that we demonstrate and possesses the property similar to that of SQ model: under some limitations on $\mathcal{D}$ every algorithm in SQ–$\mathcal{D}$ can be simulated using MQs with *persistent* classification noise.

Learning with respect to the uniform is applicable in the case when all the input bits (or *attributes*) are independent, i.e., the value of each attribute of the next randomly sampled point is chosen randomly and independently of other attributes. For this case we increase the amount of control by allowing the learning algorithm to influence the probability with which any attribute of the next sample point will be 0 or 1. Particularly, we consider the set $\mathcal{D}_\rho$ containing all the product distributions with each input bit being 1 with probability $\rho, 1/2$ or $1-\rho$ for a constant $0<\rho<1/2$.

Learning algorithm in SQ–$\mathcal{D}_\rho$ can ask for a statistical query with respect to product distribution with some attributes biased towards 0 or 1. Amount of this bias is given by ρ and is limited by restrictions on ρ. It is easy to see that boundary values of ρ represent the regular PAC learning with respect to the uniform distribution ($\rho = 1/2$) and the PAC model with membership queries ($\rho = 0$).

We give a weaker analogue of the KM algorithm for SQ–$\mathcal{D}_\rho$ that efficiently finds all the target function Fourier coefficients for sets with less than $c \log n$ attributes and larger than a given threshold (we call it the *Bounded Sieve*). We show that the Bounded Sieve (BS) by itself is, in fact, sufficient for learning the class of decision trees and weakly learning the class of DNF formulae. Then, by employing Freund's boosting algorithm, we prove that DNF are strongly learnable in SQ–$\mathcal{D}_\rho$. This requires an adaption of Freund's technique to SQ–$\mathcal{D}_\rho$. The adaptation we use is simpler and more efficient than a more general one by Aslam and Decatur [AD93] (for the regular SQ model).

As we have noted our SQ–$\mathcal{D}_\rho$ algorithms can be automatically transformed into persistent classification noise tolerant algorithms. We then show how to modify BS so that it could handle another non-trivial noise: product attribute noise in membership queries. This type of noise was previously considered as appearing in randomly sampled points, i.e., in the regular PAC model [SV88; GS95; DG95; BJT99]. We extend this noise model to learning with membership queries. Particularly, for every sample point x (randomly sampled or asked as a membership query) the oracle flips every bit i of x with probability p_i and returns the value of target function at the resulting point (unless classification noise is also present). This type of noise may reflect the situation when communication with the oracle is done using faulty channel or querying very specific point is difficult (or even impossible) to attain. Specifically, we show that our implementation of BS handles any *known* attribute noise rates bounded away from $1/2$ by a constant. This implies learnability of decision trees and weak learnability of DNF in this setting. These results can also be extended to the case when the noise rate is unknown yet *uniform*, that is, equal for all the attributes.

In the second part of this paper we show some negative results about the newly introduced model. We start by developing a characterization of classes weakly learnable in the regular SQ model and extend it to the SQ–$\mathcal{D}$ model. The characterization will enable us to show that the class of all parity functions having at most $k(n)$ variables is not weakly learnable in SQ–$\mathcal{D}_\rho$ if $k(n)$ is $\omega(\log n)$. This fact complements the Bounded Sieve which obviously learns every such class for $k(n) = O(\log n)$. Another interesting application of this characterization will help us to show that although the model we have introduced is powerful enough to learn DNF expressions, it still cannot learn a class of parity functions learnable in the PAC model with classification noise (to show this we rely on the result by Blum et al. [BKW00]). We also show that the class of all parity functions is not weakly learnable with respect to any "non-biased" distribution (i.e., not containing points with probability greater than $\frac{1}{poly(n)}$) even in SQ–$\mathcal{D}$ for $\mathcal{D}$ containing all "non-biased" distributions. This gives futher evidence that the class of all parity functions is hard for any statistics based learning.

1.1 Relation to Other Works

Shamir and Shwartzman [ShSh95] were first to consider extending the notion of statistical query as a way to produce (persistent) noise tolerant algorithms. Their extension is a statistical query analogue of "second-order" queries, that is, each PAC example consists of a pair $(x, y) \in X^2$ with label $\ell = (f(x), f(y))$ and examples are sampled randomly with respect to distributions over X^2 (not necessarily product). They show that the KM algorithm can be implemented using this type of extended statistical queries and prove that this implementation can be simulated using membership queries with persistent classification noise. Using the same approach it was proven [JSS97] that DNF can be learned by noisy membership queries.

The main distinction between this approach and the SQ–$\mathcal{D}$ is that Jackson, Shamir and Shwartzman investigated strong extensions of the SQ model which can be simulated by noisy membership queries and then showed that DNF is learnable in an extension of this kind. On the other hand, we show relatively weak extension of the SQ model, based on regular ("first-order") statistical queries, in which DNF is learnable. The extension we describe can as well be simulated by noisy membership queries.

Kearns in his original paper on the SQ model [Kea93] showed that the SQ model is weaker than the regular PAC model. Specifically, he proved that the class of all parity functions is not learnable in the SQ model. Subsequently, Blum et al. [BFJ+94] gave a general characterization of the complexity of statistical query learning in terms of the number of uncorrelated functions in the concept class. We give a very similar characterization in terms of number of functions required to "approximate" every function in the concept class. Our characterization is not stronger as a tool for proving negative results and its main advantage is its very simple proof as well as easy extendability to learning in SQ–$\mathcal{D}$.

1.2 Organization

The rest of this paper is organized as follows. In the next section we provide definitions and several well-known facts that are used later in the paper. In Section 3 we define the SQ–$\mathcal{D}$ model and present all the learnability results for the model. Discussion of the attribute noise and the implementation of BS for PAC learning with membership queries corrupted by product attribute noise is given in Section 4. Finally, the characterization of classes weakly learnable in SQ–$\mathcal{D}$ and its applications are described in Section 5.

2. Definitions and Notation

Before we start describing our results we provide all the relevant definitions. This section describes models, tools, and techniques that we are employing in our discussion.

2.1 General Notation

We consider an input space $X = \{0,1\}^n$. A *concept* is a Boolean function on X. For convenience, when applying the Fourier transform technique, we define Boolean functions to have output in $\{-1,+1\}$, where 1 stands for TRUE and -1 for FALSE. A *concept class* $\mathcal{F}$ is a set of concepts. Concept class is defined together with the representation of every concept in it. The representation scheme used (e.g., DNF or decision tree) defines the mapping *size*(f) that measures the size of the smallest representation of f within the given representation scheme. Concept class together with a representation scheme is called representation class. We denote the representation class of DNF expressions by DNF, and the representation class of decision trees by DT.

For any vector y we denote by y_i the i-th element (or bit) of y and by $y_{[i,j]}$ we denote $y_i y_{i+1} \dots y_j$. Bits of a member in our domain are usually called *attributes* and are referred using *variables* $x_1, x_2, \dots, x_n$. We use $[a]^k$ to denote a vector of k elements each equal to a. We use U to denote the uniform distribution over X. For Boolean functions $f(x)$ and $g(x)$ over the domain X we say that $g(x)$ ϵ−approximates $f(x)$ with respect to distribution D (on X) if $Pr_D[f(x) = g(x)] \geq 1 - \epsilon$. It is easy to see that this is equivalent to saying that $\mathbf{E}_D[fg] \geq 1 - 2\epsilon$. For any real-valued function ϕ define $L_\infty(\phi) = \max_{x \in X} |\phi(x)|$.

Denote by $\Delta(\gamma)$ and $\Delta'(\gamma)$ probability distributions over $\{0,1\}$ and $\{-1,1\}$ respectively, such that the probability of 1 is γ (that is, the probability distribution of Bernoulli random variable with parameter γ).

If $p(n) = O(q(n))$ and $q'(n)$ is $q(n)$ with all the logarithmic factors removed we write $p(n) = \tilde{O}(q(n))$. Similarly we define $\tilde{\Omega}$ for lower bounds. This extends to k-ary functions in an obvious way.

2.2 Learning Models

Throughout this paper we refer to several models of learning. Below we describe all the mentioned models and provide the relevant definitions.

2.2.1 The PAC Model

Most of the models we are discussing are based on Valiant's Probably Approximately Correct (PAC) model of learnability. The basic version of this model reflects passive learning from random examples. Formally, let f be a Boolean function and D be a distribution over X. An *example oracle* for f with respect to D — $\mathrm{EX}(f, D)$, is an oracle that on request draws an instance x at random according to the probability distribution D and returns the example $\langle x, f(x) \rangle$. A *membership oracle* for f — $\mathrm{MEM}(f)$, is an oracle that given any point x returns the value $f(x)$. Let ϵ and δ be positive values (called the *accuracy* and the *confidence* of learning, respectively). We say that the representation class $\mathcal{F}$ is (*strongly*) PAC-learnable if there is an algorithm $\mathcal{A}$ such that for any ϵ,δ, $f \in \mathcal{F}$ of size s (the *target function*) and any distribution D, with probability at least $1-\delta$, algorithm $\mathcal{A}(EX(f, D), n, s, \epsilon, \delta)$ produces an

ϵ–approximation for f with respect to D in time polynomial in $n, s, 1/\epsilon$ and $\log(1/\delta)$. We generally drop the "PAC" from "PAC-learnable" when the model is clear from the context.

We will consider a number of variations on the basic PAC model. Let $\mathcal{M}$ be any model of learning (e.g., PAC). If $\mathcal{F}$ is $\mathcal{M}$-learnable by an algorithm $\mathcal{A}$ that requires a membership oracle then $\mathcal{F}$ is *$\mathcal{M}$-learnable using membership queries* (or learnable in $\mathcal{M}$+MQ). If $\mathcal{F}$ is $\mathcal{M}$-learnable for $\epsilon = 1/2 - p(n, s)$, where $p(\cdot, \cdot)$ is a fixed polynomial, then $\mathcal{F}$ is weakly $\mathcal{M}$-learnable. Note that in the above definitions we have placed no restriction on the example distribution D, that is they are *distribution-independent*. If $\mathcal{F}$ is $\mathcal{M}$-learnable only for a specific distribution D (known to $\mathcal{A}$) then $\mathcal{F}$ is $\mathcal{M}$-learnable with respect to D (such learning is called *distribution-specific* and the learning model is denoted by $\mathcal{M}_D$).

Remark 1 *When we are dealing with concept classes in which all the concepts have representations of polynomial (in n) size the parameter s could be omitted from the discussion.*

2.2.2 Noise Models

In the above described models it was always assumed that all the labels (i.e., values of the target function) that are received from oracles are accurate. We are now going to describe the extensions of the above models in which labels provided with the examples (random or received from membership oracle) are corrupted by random noise.[2] This extension was first considered in the learning literature by Angluin and Laird [AL88]. Since then, in addition to being the subject of a number of theoretical studies [AL88; Lai88; Sl88], the *classification noise* model has become a common paradigm for experimental machine learning.

Below we will give the formal definitions that incorporate the notion of random classification noise into the PAC learning model. A new parameter $\eta \leq 1/2$ called the *noise rate* is introduced and the regular example oracle $\mathrm{EX}(f, D)$ is replaced with the faulty oracle $\mathrm{EX}^\eta(f, D)$. On each call, $\mathrm{EX}^\eta(f, D)$ first draws an input x according to D. The oracle then chooses randomly a value $\zeta \in \{-1, 1\}$; 1 with probability η and -1 with probability $1 - \eta$ and returns $\zeta f(x)$. When η approaches $1/2$ the result of the corrupted query approaches the result of the random coin flip, and therefore learning becomes more and more difficult. Thus the running time of algorithms in this model is allowed to polynomially depend on $\frac{1}{1-2\eta}$. This model of noise is not suitable for corrupting labels returned by a membership oracle $\mathrm{MEM}(f)$ since a learning algorithm can, with high probability, find the correct label at point x by asking the label of x polynomial (in $\frac{1}{1-2\eta}$) number of times and then returning the label that appeared in the majority of answers. An appropriate modification of the noise model is the introduction of *persistent classification noise* by Goldman, Kearns and Shapire [GKS94]. In this model, as before, the answer to a query at each point x is corrupted by random value ζ. But if the oracle was already queried about the value of f at some specific point x, it returns the same value it has returned in the first query (i.e, in such a case the noise persists and is not purely random). There is another subtle but important distinction of this model. We cannot require learning algorithms in this model to work for an arbitrarily small confidence parameter δ. Because with some (though negligible) probability target function can be persistently totally corrupted (for example negated) making learning absolutely infeasible. So we must impose a positive lower bound on the confidence δ that

2. Noise in points themselves or attribute noise is defined in Section 4.

can be required from the learning algorithm in the persistent noise model, although this bound will be negligible.

2.2.3 The Statistical Query Learning Model

The *statistical query model* introduced by Kearns [Kea93] is a natural restriction of the PAC learning model in which a learning algorithm cannot see labelled examples of the target concept, but instead may only estimate probabilities involving the target concept. Formally, when learning with respect to distribution D the learning algorithm is given access to $\mathrm{STAT}(f, D)$ – a *statistical query oracle* for target concept f with respect to distribution D. A query to this oracle is a pair (ψ, r), where $\psi : \{0,1\}^n \times \{-1,+1\} \to \{-1,+1\}$ is a *query function* and $r \in [0,1]$ is a real number called the *tolerance* of the query. The oracle may respond to the query with any value v satisfying

$$|\mathbf{E}_D[\psi(x, f(x))] - v| \leq r \ .$$

We denote the result returned by oracle $\mathrm{STAT}(f, D)$ to query (ψ, r) by $\mathrm{STAT}(f, D)[\psi, r]$.

We say that the concept class $\mathcal{F}$ is *(strongly) learnable from statistical queries* (SQ–learnable) if there is a learning algorithm $\mathcal{A}$, and polynomials $p(\cdot,\cdot,\cdot)$, $q(\cdot,\cdot,\cdot)$ and $r(\cdot,\cdot,\cdot)$ such that for every n, $f \in \mathcal{F}$, distribution D and ϵ the following holds. Given n, the size s of f, ϵ and access to $\mathrm{STAT}(f, D)$, algorithm $\mathcal{A}$ produces an ϵ–approximation to f with respect to D in time bounded by $p(n, s, 1/\epsilon)$. Furthermore, algorithm $\mathcal{A}$ makes queries (ψ, r) in which ψ can be evaluated in time bounded[3] by $q(n, s, 1/\epsilon)$, and in which r is lower-bounded by $1/r(n, s, 1/\epsilon)$.

Although the "usual" randomness of PAC algorithms is incorporated in the statistical queries some applications may still need randomization. In this case we, as usual, have confidence parameter δ, i.e., get the required approximation with probability at least $1-\delta$, and the running time of the algorithm is also polynomial in $\log(1/\delta)$.

It is important to note that the statistical query model is indeed a restriction of the basic PAC model. This follows from the fact that a query (ψ, r) to $\mathrm{STAT}(f, D)$ where ψ can be computed in polynomial time and r lower-bounded by inverse of the polynomial can (with high probability) be estimated (within r) by computing ψ on polynomial number of samples and then averaging the result (see Section 2.3 for more details). Moreover, as it was shown by Kearns [Kea93], this estimation can also be made in the presence of random classification noise. This means that any SQ algorithm can be automatically transformed into noise tolerant algorithm. This important property of the model allowed Kearns to obtain noise-tolerant algorithms for practically every concept class for which an efficient learning algorithm in the original noise-free PAC model is known.

2.3 Estimating Expected Values

We will frequently need to estimate the expected value of a random variable. Although the SQ model itself provides us with such estimates, sometimes a random variable will be a result of a statistical query. In such a case we usually cannot use a statistical query to estimate

3. In most of the cases we will use simple and explicit query functions. In such cases we omit the analysis of complexity of these functions.

the expectation of the variable. Thus we will find the expectation by sampling the variable and averaging the results. The justification for this technique is usually referred as *Chernoff bounds.* The Chernoff's lemma which is concerned with 0,1 random variables was generalized by Hoeffding. Hoeffding's lemma, presented below, allows to compute the sample size needed to ensure that with high probability the sample mean closely approximates the true mean of the random real-valued variable.

Lemma 2 (Hoeffding) *Let X_i, $1 \leq i \leq m$, be independent random variables all with mean μ such that for all i, $a \leq X_i \leq b$. Then for any $\lambda > 0$,*

$$\mathbf{Pr}\left[\left|\frac{1}{m}\sum_{i=1}^{m} X_i - \mu\right| \geq \lambda\right] \leq 2e^{-2\lambda m/(b-a)^2} \ .$$

According to the lemma if we sample randomly and independently a random variable Y with values in $[a, b]$ then to reach the accuracy of at least λ with confidence of at least δ it is sufficient to take

$$(b-a)^2 \ln(2/\delta)/(2\lambda^2) = O(\log(1/\delta)(b-a)^2/\lambda^2)$$

samples of Y. That is, Hoeffding's lemma gives the following corollary.

Corollary 3 *Let Y represent a random variable that produces values in the range $[a, b]$. Then there is an algorithm* `AMEAN`$(Y, b-a, \lambda, \delta)$ *that for any such Y and any $\lambda > 0$ produces a value μ such that $|\mathbf{E}[Y] - \mu| \leq \lambda$ with probability at least $1 - \delta$. Furthermore,* `AMEAN` *runs in $O(\log(1/\delta)(b-a)^2/\lambda^2)$ time.*

Remark 4 *Another implication of the fact that a value of random variable is a result of a statistical query is that we get its value within some tolerance r. It is easy to see that in such a case the above execution of procedure* `AMEAN` *will return the expectation within $r + \lambda$.*

In general, we say that a constant value c can be *efficiently approximated* if there exists an algorithm that for every τ and δ, with probability at least $1 - \delta$ produces a value that estimates c within τ and runs in time polynomial in $1/\tau$ and $\log(1/\delta)$.

The SQ model gives the learner ability to estimate expectations of Boolean functions involving the value of the target function $f(x)$. In further algorithms we will need to get expectations of any real-valued functions involving $f(x)$. For this purpose we will rely on the following simple lemma.

Lemma 5 *There exists a procedure* `RV-SQ` *that for every real-valued function $\phi : \{0,1\}^n \times \{-1,1\} \to [-b, b]$, tolerance r and distribution D,* `RV-SQ`$(STAT(f, D), \phi, b, r)$ *returns a value v satisfying*

$$|\mathbf{E}_D \phi(x, f(x)) - v| \leq r$$

Its time complexity is $O(\log(b/r))$. The required tolerance is bounded from below by $\frac{r}{4b\log(b/r)}$ and complexity of the query functions is the complexity of computation of ϕ plus $O(n)$.

Proof: We find the required expectation by decomposing ϕ into linear combination of Boolean functions according to its binary representation in the following way:
Let k be equal to $\lceil \log b \rceil$ and $l = \lceil \log \frac{2}{r} \rceil$

$$\phi(x, f(x)) = -2^k + 2^k \phi_k(x, f(x)) + 2^{k-1}\phi_{k-1}(x, f(x)) + \ldots + \phi_0(x, f(x)) +$$
$$\frac{1}{2}\phi_{-1}(x, f(x)) + \ldots + 2^{-l}\phi_{-l}(x, f(x)) + \psi(x, f(x))$$

Where all the ϕ_i are Boolean 0,1-valued functions representing the bits of $\phi(x, f(x)) + 2^k$ (nonnegative function). Choice of l ensures that $0 \leq \psi(x, f(x)) \leq r/2$. Since we would like the composition to use ± 1-valued functions, we can further write

$$\phi(x, f(x)) = 2^{k-1}(2\phi_k(x, f(x)) - 1) + 2^{k-2}(2\phi_{k-1}(x, f(x)) - 1) + \ldots +$$
$$2^{-l-1}(2\phi_{-l}(x, f(x)) - 1) + (\psi(x, f(x)) - 2^{-l-1})$$

For every $-l \leq j \leq k$, $2\phi_j(x, f(x)) - 1$ is a ± 1-valued Boolean function depending on x and $f(x)$. Thus to get the required approximation, for every $-l \leq j \leq k$ we approximate $\mathbf{E}_D[2\phi_j(x, f(x)) - 1]$ within $\frac{r}{2^{j+1}(k+l+1)}$ using query

$$\left(2\phi_j(x, f(x)) - 1, \frac{r}{2^{j+1}(k+l+1)}, D\right)$$

Let μ_j be the result of the query. The procedure returns the value $\sum_{-l \leq j \leq k} 2^j \mu_j$. This value is a valid result since

$$|\mathbf{E}_D\phi(x, f(x)) - \sum_{-l \leq j \leq k} 2^j \mu_j| \leq \sum_{-l \leq j \leq k} 2^j \frac{r}{2^{j+1}(k+l+1)} + \mathbf{E}_D[|\psi(x, f(x)) - 2^{-l-1}|] \leq$$
$$\sum_{-l \leq j \leq k} \frac{r}{2(k+l+1)} + r/2 = r.$$

The complexity of this algorithm is $O(\log(b/r))$. The required tolerance is at least

$$\frac{r}{2^{k+1}(k+l+1)} \geq \frac{r}{4b \log(b/r)}.$$

Complexity of the query functions is the complexity of computation of ϕ plus $O(n)$. $\square$

2.4 The Fourier Transform

Many parts of our analysis rely heavily on the use of the well-known Fourier transform technique. This beautiful technique was introduced to the learning theory by Linial, Mansour, and Nisan [LMN89]. Below we give a short description of the technique and its notation (an extended survey of the technique and the `KM` algorithm was given by Mansour [Man94]).

Since all the expectations and probabilities we use when applying this technique will be with respect to the uniform distribution, we will sometimes omit U in our formulae. For every vector $a \in \{0,1\}^n$ we define χ_a—the parity function for the vector by

$$\chi_a(x) = (-1)^{\sum_{i \in \{1,\ldots,n\}} a_i x_i}$$

A common alternative is to associate a parity function with a set $A \subseteq \{1, \ldots, n\}$ of attribute indices that influence the parity. This definition implies that χ_a and χ_A are equivalent whenever $A = \{i \mid a_i = 1\}$. Also define the inner product by $\langle f, g \rangle = \mathbf{E}[f(x)g(x)]$ and define the norm as usual by $\|f\| = \sqrt{\mathbf{E}_x[f^2(x)]}$. In this context we define $w(a)$ to be the number of bits a_i equal to 1 in a and call it the *weight* of a. Let $\oplus$ denote the bit-wise XOR operation on binary vectors. Parity functions possess several important and easily verifiable properties:

1. $\chi_a(x)\chi_b(x) = \chi_{a\oplus b}(x)$;
2. $\chi_a(x)\chi_a(y) = \chi_a(x \oplus y)$;
3. $\mathbf{E}_x[\chi_a(x)] = 0$ for every $a \neq [0]^n$;
4. $\chi_{[0]^n} \equiv 1$ and thus $\mathbf{E}_x[\chi_{[0]^n}(x)] = 1$.

By these properties, for every $a, b \in \{0,1\}^n$,

$$\langle \chi_a, \chi_b \rangle = \mathbf{E}_x[\chi_a(x)\chi_b(x)] = \mathbf{E}_x[\chi_{a\oplus b}(x)] = \begin{cases} 0 & a \neq b \\ 1 & a = b \end{cases}$$

Thus the set $\{\chi_a\}_{a\in\{0,1\}^n}$ is orthonormal. Its size is 2^n which is also the dimension of the vector space of real-valued functions on $\{0,1\}^n$, i.e., the set forms the basis for the vector space. That is, every function $g : \{0,1\}^n \to \mathbb{R}$ can be uniquely expressed as a linear combination of parity functions:

$$g(x) = \sum_{a\in\{0,1\}^n} \hat{g}(a)\chi_a(x) \ .$$

The coefficients $\hat{g}(a)$ are called *Fourier coefficients* and vector of coefficients $\hat{g}$ the Fourier transform of g. The *degree* of the coefficient $\hat{g}(a)$ is the weight of a.[4] Because of the orthonormality of the parity functions, $\hat{g}(a) = \mathbf{E}[g(x)\chi_a(x)]$. Thus $\hat{g}(a)$ represents the correlation of g with the parity function χ_a.

Parseval's identity states that for every function g, $\|g\|^2 = \mathbf{E}[g^2] = \sum_a \hat{g}^2(a)$. For Boolean g this implies that $\sum_a \hat{g}^2(a) = 1$.

The Fourier transform technique is usually used in the following way. Given some access to the target function f we try to find its Fourier coefficients or at least the "largest" among them (here and in further discussion we refer to the absolute value of the coefficient). These coefficients define a real-valued function g. We then produce hypothesis by taking $h = sign(g)$. Justification for this method is provided by the following well-known lemma.

Lemma 6 *Let f be a Boolean function and g be any real-valued function then*

$$\mathbf{Pr}[f(x) \neq sign(g)(x)] \leq \mathbf{E}[(f(x) - g(x))^2] = \sum_{a\in\{0,1\}^n} (\hat{f}(a) - \hat{g}(a))^2 \ .$$

4. For a parity function defined using a set of indices the degree of the corresponding coefficient is the size of the set.

2.5 The KM Algorithm

One of the most fundamental application of the Fourier analysis is the Kushilevitz-Mansour algorithm [KM91]. This randomized algorithm uses membership queries to find all the Fourier coefficients of the target function larger than some threshold θ in time polynomial in $n, \log(1/\delta)$ and θ^{-1}. The algorithm is based on an ingenious way to estimate the sum of squares of all the Fourier coefficients for vectors starting with some given prefix. Formally for $0 \le k \le n$ and $\alpha \in \{0,1\}^k$ denote

$$C_\alpha = \sum_{b \in \{0,1\}^{n-k}} \hat{f}^2(\alpha b) \ .$$

As it has been proved by Kushilevitz and Mansour

$$C_\alpha = \mathbf{E}_x \left[\mathbf{E}_y[f(yx)\chi_\alpha(y)]\right]^2 \ ,$$

where the expectation is uniform over $x \in \{0,1\}^{n-k}$, $y \in \{0,1\}^k$. This formula means that C_α can be efficiently approximated by random sampling (see Section 2.3). If there exists at least one Fourier coefficient for a vector starting with α and larger (by absolute value) than θ then $C_\alpha \ge \theta^2$. Using this simple observation Kushilevitz and Mansour wrote a recursive procedure (named `Coef`) that given α finds all the Fourier coefficient for vectors starting with α and larger than θ as follows. For α being a vector of length n `Coef`(α) returns the vector if its Fourier coefficient is larger than θ. For shorter prefixes the procedure checks whether $C_\alpha \ge \theta^2$. If so executes itself for prefixes $\alpha \cdot 0$ and $\alpha \cdot 1$ and returns the union of the results. Otherwise ($C_\alpha < \theta^2$), returns the empty set. When executed on an empty prefix this procedure should return all the Fourier coefficients that are larger than θ. Effectiveness of this algorithm follows from the fact that for any given length of prefix k

$$\sum_{\alpha \in \{0,1\}^k} C_\alpha = \sum_{a \in \{0,1\}^n} \hat{f}(a)^2 = \|f\|^2 = 1 \ .$$

This means that for any given length of prefix `Coef` will be called at most $1/\theta^2$ times, i.e., the resulting algorithm is polynomial in $n, \log(1/\delta)$ and $1/\theta$.

In this description we have neglected the fact that C_α is not known exactly but only estimated. Minor modifications required to solve this problem can be found in the original description of the algorithm [KM91] or in the similar analysis of our analogue to this algorithm described in Section 3.2.

Remark 7 *It can be easily seen that the algorithm can be applied to any real-valued function $\phi(x)$ (and not only to Boolean). In such a case for any given length of prefix the procedure* `Coef` *will be called at most $\|\phi\|^2/\theta^2$ and thus running time of the* KM *algorithm becomes polynomial in $\|\phi\|$. Moreover, we do not need to know the value of $\phi(x)$ exactly. An ability to efficiently estimate the value of $\phi(x)$ with any desired accuracy is sufficient since the value of $\phi(x)$ is used only for estimating C_α's and Fourier coefficients of ϕ.*

3. Learning by Extended Statistical Queries

In this section we introduce a new model of learning. The model (or actually a collection of models) we define is at least as strong as the basic PAC model and is weaker (or equivalent) to the use of membership queries. For all of our applications a restriction of the new model to statistical queries is sufficient. Moreover, as we will demonstrate, the SQ version of the new model has a few important advantages. Thus we almost immediately start discussing the restricted version.

We begin by defining the SQ–$\mathcal{D}$ model and discussing its properties. Then we apply these general ideas to learning of decision trees and DNF expressions with respect to the uniform distribution. In particular, we implement the Bounded Sieve algorithm which has the functionality of the `KM` algorithm with an additional restriction. The Bounded Sieve by itself is sufficient for learning of DT and weak learning of DNF. By adapting Freund's boosting technique to our model we show that DNF expressions are also strongly learnable.

3.1 Extending the SQ Model

We are now going to give a detailed description of the extension of the basic PAC model and the SQ model. In this new model we allow the learning algorithm to supply a distribution with respect to which the next point will be sampled. Formally, let D be a distribution over X and $\mathcal{D}$ be a set of distributions over X containing distribution D. The new example oracle $\mathrm{EX}(f, \mathcal{D})$ is defined as follows. A query to this oracle is a distribution $D' \in \mathcal{D}$. To such a query the oracle $\mathrm{EX}(f, \mathcal{D})$ responds with example $\langle x, f(x)\rangle$, where x is sampled randomly and independently according to distribution D'. In other words, access to $\mathrm{EX}(f, \mathcal{D})$ is equivalent to access to all the oracles of the form $\mathrm{EX}(f, D')$, where $D' \in \mathcal{D}$. Learning with access to the newly-defined oracle is referred as learning in the PAC–$\mathcal{D}$ model. Learnability (with respect to D) in this new model is defined as in the regular PAC model with the regular example oracle $\mathrm{EX}(f, D)$ replaced by $\mathrm{EX}(f, \mathcal{D})$.

The extension of the PAC learning model obviously possesses the following properties:

- PAC–$\mathcal{D}$ is equivalent to the regular PAC model for $\mathcal{D} = \{D\}$;
- We can simulate the PAC–$\mathcal{D}$ model using membership queries;
- If we do not restrict $\mathcal{D}$ we might be able to simulate membership queries (e.g., using distributions with all the weight concentrated in one point).

Our next step is restricting the PAC–$\mathcal{D}$ model to statistical queries, or extending Kearns' SQ model in the similar fashion. That is, instead of the regular $\mathrm{STAT}(f, \mathcal{D})$ oracle a learning algorithm in the SQ–$\mathcal{D}$ model is provided with access to all the oracles $\mathrm{STAT}(f, \mathcal{D}')$ where $D' \in D$. More formally, learning algorithm in the *SQ–$\mathcal{D}$* model is supplied with the $\mathrm{STAT}(f, \mathcal{D})$ oracle, where f is the target concept. A query to this oracle is a triple (ψ, r, D') where $\psi : \{0,1\}^n \times \{-1,+1\} \to \{-1,+1\}$ is a query function, $r \in [0,1]$ is a tolerance (as in the SQ model) and D' is a distribution from $\mathcal{D}$. To such a query the oracle responds with the value v satisfying $|\mathbf{E}_{D'}[\psi(x, f(x))] - v| \leq r$. Respectively, we say that the concept class $\mathcal{F}$ is learnable in the SQ–$\mathcal{D}$ model with respect to D if it is learnable by a polynomial algorithm (as defined in the regular SQ model) with access to the newly-defined oracle.

An important property of the model that results from the fact that it is statistical-query-based is that $\mathrm{STAT}(f, \mathcal{D})$ can be simulated by membership queries in the presence of random classification noise. The simulation is done by offsetting the effect of the noise on a query [Kea93]. Since we are simulating using membership queries we are actually interested in learning with *persistent* classification noise. Offsetting noise in this model can be a more complicated task. However, if all the sample points in the sample that was used for simulating statistical queries are different then the persistent noise in the sample is equivalent to usual random classification noise and therefore we can offset the effect of noise in the same way as before. In the following theorem we show that under certain simple restriction on set $\mathcal{D}$ the probability to get two equal points in the sample will be negligible and therefore we will be able to use Kearns' noise offsetting procedure.[5] Denote by $L_\infty(\mathcal{D}) = \max_{D' \in \mathcal{D}}\{L_\infty(D')\}$.

Theorem 8 *Let $\mathcal{F}$ be a concept class and $\mathcal{D}$ be a set of distributions with sampling oracle available for every $D' \in \mathcal{D}$. If $\mathcal{F}$ is learnable in SQ–$\mathcal{D}$ with respect to D and $L_\infty(\mathcal{D}) < \tau^n$ for a constant $0 < \tau < 1$, then $\mathcal{F}$ is learnable with respect to D using membership queries corrupted by random persistent classification noise.*

Proof: Let $\mathcal{A}$ be an algorithm that learns $\mathcal{F}$ in SQ–$\mathcal{D}$. Let us examine the execution of $\mathcal{A}$ for a particular $f \in \mathcal{F}$. We can assume that the learning parameters s and ϵ are subexponential[6] in n (otherwise a trivial exponential learning algorithm will solve the problem). The number of queries that $\mathcal{A}$ asks during its execution is bounded by a fixed polynomial in n, s, and ϵ (denote it by $p_1(n, s, \epsilon^{-1})$). Let $1/p_2(n, s, \epsilon^{-1})$ denote the inverse-polynomial bound on the required accuracy of every query. Let $\sigma = \frac{\tau^{n/2}}{2p_1(n,s,\epsilon^{-1})}$. According to Corollary 3, in order to estimate all the $\mathcal{A}$'s queries with accuracy of $1/p_2(n, s, \epsilon^{-1})$ and with confidence of σ polynomial number of sample points is required (the polynomial depends on n, s, ϵ^{-1}, and $\frac{1}{1-\eta}$ since the estimation is done in the presence of noise). Denote this polynomial bound by $p_3(n, s, \epsilon^{-1}, \frac{1}{1-\eta})$. Assuming that the noise is random, all the estimations will succeed with probability at least $1 - \sigma p_1(n, s, \epsilon^{-1}) = 1 - \tau^{n/2}/2$. The probability to see a point more than once in the above sample is less than

$$p_3^2(n, s, \epsilon^{-1}, \frac{1}{1-\eta})L_\infty(\mathcal{D}) \leq p_3^2(n, s, \epsilon^{-1}, \frac{1}{1-\eta})\tau^n < \tau^{n/2}/2 \ . \qquad (1)$$

Therefore, with probability at least $1-\tau^{n/2}/2$, the noise in the generated sample is truly random. This means that the simulation with succeed with probability at least $1 - \tau^{n/2}$.

Now, if we impose a lower bound of $\tau^{n/2}$ on confidence δ that can be required from the learning algorithm (as is allowed when the noise is persistent), then the above-described procedure gives the algorithm that learns $\mathcal{F}$ by using membership queries with persistent classification noise. □

3.2 Learning with Respect to the Uniform Distribution Using Product Distributions

In this section we will examine set $\mathcal{D}$ that contains product distributions such that every input bit is 1 with probability ρ , $1/2$ or $1-\rho$, that is, for every input bit x_i, $\mathbf{Pr}[x_i = 1] = p_i$ independently of all the other inputs and $p_i \in \{\rho, 1/2, 1-\rho\}$ (we call $p = p_1 p_2 \ldots p_n$ a

5. Proof of this theorem can be easily extracted from [JSS97].
6. A more accurate restriction on these parameters can be deduced from equation (1).

probability vector). We assume that ρ is a constant satisfying $1/2 > \rho > 0$. For every $p \in \{\rho, 1/2, 1-\rho\}^n$ denote by D_p the product distribution defined by this probability vector. Denote by

$$\mathcal{D}_\rho = \{D_p \mid p \in \{\rho, 1/2, 1-\rho\}^n\} \ .$$

This distribution class will be used to learn with respect to the uniform distribution which itself is contained in $\mathcal{D}_\rho$ for any ρ.

For a probability vector p and $x \in \{0,1\}^n$ denote by $p \oplus x$ probability vector for which $(p \oplus x)_i = p_i$ if $x_i = 0$ and $(p \oplus x)_i = 1 - p_i$ if $x_i = 1$. Since $p \in \{\rho, 1/2, 1-\rho\}^n$ we also have $p \oplus x \in \{\rho, 1/2, 1-\rho\}^n$.

According to the discussion in Section 1, we call this ρ the *degree of control* that a learning algorithm in SQ–$\mathcal{D}_\rho$ has. Intuitively, smaller values of ρ represent larger degree of control (with 0 giving the equivalent of membership queries). This is proven in the next lemma.

Lemma 9 *If $1/2 \geq \rho_1 \geq \rho_2 > 0$ then any algorithm in SQ–$\mathcal{D}_{\rho_1}$ can be simulated by a randomized algorithm in SQ–$\mathcal{D}_{\rho_2}$.*

Proof: We show that a result of any statistical query in SQ–$\mathcal{D}_{\rho_1}$ can be efficiently approximated in SQ–$\mathcal{D}_{\rho_2}$. Let (ϕ, r, D_p) be a query to STAT$(f, \mathcal{D}_{\rho_1})$. For $\pi \in [0,1]$ we define $q = [\pi]^n$ (i.e., vector with every element equal to π). Let x_1 and x_2 be chosen randomly according to $\Delta(\pi)$ and $\Delta(\rho)$. It is easy to verify that $x_1 \oplus y_1$ is distributed according to $\Delta(\pi + \rho - 2\pi\rho)$. When sampling with respect to a product distribution D_p, bit i is sampled randomly and independently according to distribution $\Delta(p_i)$. Therefore,

$$\mathbf{E}_{y \sim D_p}[\phi(y, f(y))] = \mathbf{E}_{x \sim D_q}\left[\mathbf{E}_{y \sim D_{s \oplus x}}[\phi(y, f(y))]\right] \ , \tag{2}$$

where s is a probability vector such that $p_i = \pi + s_i - 2\pi s_i$ or $s_i = \frac{p_i - \pi}{1 - 2\pi}$. For $\pi = \frac{\rho_1 - \rho_2}{1 - 2\rho_2}$ ($\pi \in [0,1]$ since $1/2 \geq \rho_1 \geq \rho_2 > 0$) we get that

$$s_i = \begin{cases} \rho_2 & p_i = \rho_1 \\ 1 - \rho_2 & p_i = 1 - \rho_1 \\ 1/2 & p_i = 1/2 \end{cases}$$

Thus $D_s \in \mathcal{D}_{\rho_2}$ and for every x, $D_{s \oplus x} \in \mathcal{D}_{\rho_2}$, i.e., $\mathbf{E}_{y \sim D_{s \oplus x}}[\phi(y, f(y))]$ can be estimated by a query in SQ–$\mathcal{D}_{\rho_2}$. Equation (2) means that the result of query (ϕ, r, D_p) can be efficiently approximated by sampling in SQ–$\mathcal{D}_{\rho_2}$. $\square$

Remark 10 *For the simplicity of the exposition we have chosen the degree of control to be uniform over all the attributes (and equal to ρ). Alternatively, we can consider classes of distributions for which this degree is different for each attribute. It can be easily shown (by slightly modifying the proof of Lemma 9) that if $\mathcal{D}$ is a class of distributions giving a learning algorithm degrees of control in range $[\rho_1, \rho_2]$ then the SQ–$\mathcal{D}$ model is not stronger than SQ–$\mathcal{D}_{\rho_1}$ and not weaker than SQ–$\mathcal{D}_{\rho_2}$.*

It is important to note that according to Theorem 8 learning in SQ–$\mathcal{D}_\rho$ implies learning with persistent noise since $L_\infty(\mathcal{D}_\rho) = (1-\rho)^n$.

3.3 An Analogue to the KM Algorithm

Our purpose is to give an algorithm in the SQ–$\mathcal{D}_\rho$ model that could find "large" Fourier coefficients of the target function. When membership queries are available this task can be performed using the KM algorithm. As we will show in the characterization section this task cannot be performed in the SQ–$\mathcal{D}_\rho$ model. Instead we will give a weaker analogue of the KM algorithm implemented in the SQ–$\mathcal{D}_\rho$ model. It will find all the "large" Fourier coefficients of degree lower than ℓ in time polynomial in 2^ℓ thus allowing only the Fourier coefficients with degree bounded by $c \log n$ to be found efficiently. Particularly, for the target function f, given parameters n, θ, ℓ and δ it will, with probability at least $1-\delta$, find a set $S \subseteq \{0,1\}^n$ with the following properties:

- for all a, if $|\hat{f}(a)| \geq \theta$ and $w(a) \leq \ell$ then $a \in S$;
- for all $a \in S$, $|\hat{f}(a)| \geq \theta/2$.

We say that such a set possesses the *large Fourier coefficient property for function f, threshold θ and degree ℓ* (this property can be defined for any real-valued function).

The KM algorithm is based on a subroutine that estimates the sum of squares of all the coefficients for vectors starting with given prefix. In the same fashion our algorithm will be based on ability to estimate weighted sum of squares of all the coefficients for vectors starting with a given prefix. In particular, for $0 \leq k \leq n$ and $\alpha \in \{0,1\}^k$ denote

$$S^\rho_\alpha(f) = \sum_{b \in \{0,1\}^{n-k}} \hat{f}^2(\alpha b)(1-2\rho)^{2w(b)}.$$

Lemma 11 *There exists an SQ–$\mathcal{D}_\rho$ randomized algorithm* P1 *that for any target function f, prefix vector α of length k, confidence δ and accuracy τ,* P1(n,k,α,τ,δ) *returns, with probability at least $1-\delta$, a value estimating $S^\rho_\alpha(f)$ within τ. It runs in $O(\tau^{-2}\log(1/\delta))$ time and tolerance of its queries is bounded from below by $\tau/4$.*

Proof: Let $p = [\frac{1}{2}]^k[\rho]^{n-k}$ and $\alpha_0 = \alpha[0]^{n-k}$. The ability to estimate $S^\rho_\alpha(f)$ is based on the following equation

$$S^\rho_\alpha(f) = \mathbf{E}_{x \sim U}\left[\mathbf{E}_{y \sim D_{p \oplus x}}[f(y)\chi_{\alpha_0}(x \oplus y)]\right]^2 .$$

To prove it denote

$$f^\rho_\alpha(x) \triangleq \mathbf{E}_{y \sim D_{p \oplus x}}[f(y)\chi_{\alpha_0}(x \oplus y)] .$$

Since by Parseval's identity for every function f, $\mathbf{E}_{x \sim U}[f^2(x)] = \sum_{a \in \{0,1\}^n} \hat{f}^2(a)$, it is enough to prove that

$$f^\rho_\alpha(x) = \sum_{b \in \{0,1\}^{n-k}} \hat{f}(\alpha b)(1-2\rho)^{w(b)}\chi_{\alpha b}(x) . \tag{3}$$

As follows from the definition of $p \oplus x$ (see Section 3.2), $D_{p\oplus x}(y) = D_p(x \oplus y)$. Thus equation (3) can be proved as follows

$$\begin{aligned} f^\rho_\alpha(x) = \mathbf{E}_{y\sim D_{p\oplus x}}[f(y)\chi_{\alpha_0}(x\oplus y)] &= \sum_{y\in\{0,1\}^n} f(y)\chi_{\alpha_0}(x\oplus y)D_{p\oplus x}(y) \\ &= \sum_{y\in\{0,1\}^n} f(y)\chi_{\alpha_0}(x\oplus y)D_p(x\oplus y) \\ &= \sum_{z\in\{0,1\}^n} f(x\oplus z)\chi_{\alpha_0}(z)D_p(z) \\ &= \mathbf{E}_{z\sim D_p}[f(x\oplus z)\chi_{\alpha_0}(z)] \\ &= \mathbf{E}_{z\sim D_p}\left[\sum_{a\in\{0,1\}^n} \hat{f}(a)\chi_a(x\oplus z)\chi_{\alpha_0}(z)\right] \\ &= \mathbf{E}_{z\sim D_p}\left[\sum_{a\in\{0,1\}^n} \hat{f}(a)\chi_a(x)\chi_{a\oplus\alpha_0}(z)\right] \\ &= \sum_{a\in\{0,1\}^n} \hat{f}(a)\chi_a(x)\mathbf{E}_{z\sim D_p}[\chi_{a\oplus\alpha_0}(z)]. \end{aligned} \tag{4}$$

But since $p = [\frac{1}{2}]^k[\rho]^{n-k}$ and $\alpha_0 = \alpha[0]^{n-k}$,

$$\begin{aligned} \mathbf{E}_{z\sim D_p}[\chi_{a\oplus\alpha_0}(z)] = \prod_{i\in\{1,\dots,n\}} \left[1 - p_i + p_i(-1)^{a_i\oplus\alpha_{0,i}}\right] &= \prod_{i\in\{1,\dots,n\}, a_i\oplus\alpha_{0,i}=1} (1-2p_i) \\ &= \left(\prod_{i\in\{1,\dots,k\}, a_i\neq\alpha_i} 0\right)\left(\prod_{i\in\{k+1,\dots,n\}, a_i=1} (1-2\rho)\right), \end{aligned} \tag{5}$$

that is, if $a = \alpha b$ then $\mathbf{E}_{z\sim D_p}[\chi_{a\oplus\alpha_0}(z)] = (1-2\rho)^{w(b)}$, otherwise $\mathbf{E}_{z\sim D_p}[\chi_{a\oplus\alpha_0}(z)] = 0$. This result substituted to the last term of equation (4) gives precisely the required identity.

Our goal is to estimate $S^\rho_\alpha(f) = \mathbf{E}_{x\sim U}[(f^\rho_\alpha(x))^2]$. By the definition of f^ρ_α, for every z, we can estimate $f^\rho_\alpha(z)$ within $\tau/4$ using statistical query $(f(x)\chi_{\alpha_0}(z\oplus x), \tau/4, D_{p\oplus\alpha_0})$ and we can assume that the estimate has an absolute value of at most 1. It is easy to see that the square of this estimate is an estimate for $(f^\rho_\alpha(z))^2$ within $\tau/2$. Therefore $(f^\rho_\alpha(z))^2$ is a random variable whose value we are able to estimate within $\tau/2$. Denote it by Y. By Corollary 3 with Remark 4 we get that the call to $\mathtt{AMEAN}(Y, \tau/2, 1, \delta)$ will give us the estimate for $S^\rho_\alpha(f)$ within $\tau/2 + \tau/2 = \tau$. Figure 1 summarizes the description of procedure `P1`. Since computing the value of Y takes $O(1)$ time, the complexity of the algorithm is $O(\tau^{-2}\log(1/\delta))$. □

With algorithm `P1` we are now ready to describe the Bounded Sieve. We define $c_0 \triangleq -2\log(1-2\rho)$ (it is a positive constant).

Theorem 12 *There exists an SQ–$\mathcal{D}_\rho$ randomized algorithm $\mathtt{BS}_\rho$ that for any target concept f, threshold $\theta > 0$, confidence $\delta > 0$ and degree bound $0 \leq \ell \leq n$, $\mathtt{BS}_\rho(n, \theta, \ell, \delta)$ returns,*

Input: Access to $STAT(f, \mathcal{D}_\rho)$; $0 \leq k \leq n$; $\alpha \in \{0,1\}^k$; $\ell \leq n$; $\theta > 0$; $\tau > 0$; $\delta > 0$
Output: Value v such that, with probability at least $1-\delta$, $|S^\rho_\alpha(f) - v| \leq \tau$.

1. $p \leftarrow [\frac{1}{2}]^k[\rho]^{n-k}$
2. $\alpha_0 \leftarrow \alpha[0]^{n-k}$
3. $Y \equiv$ draw z w.r.t. U and compute $(\mathrm{STAT}(f, D_\rho)[f(x)\chi_{\alpha_0}(z \oplus x), \tau/6, D_{p\oplus\alpha_0}])^2$
4. $v \leftarrow$ `AMEAN`$(Y, b-a=1, \tau/2, \delta)$
5. **return** v

Figure 1: Procedure `P1`

with probability at least $1-\delta$*, a set with the large Fourier coefficient property for function* f*, threshold* θ *and degree* ℓ*. Its running time is polynomial in* $n, 2^\ell, \log(1/\delta)$ *and* θ^{-1}*, particularly* $\tilde{O}(n2^{3c_0\ell}\theta^{-6}\log(1/\delta))$*. Tolerance of queries is bounded from below by an inverse of a polynomial in* 2^ℓ *and* θ^{-1}*, specifically* $2^{-c_0\ell}\theta^2/16$*.*

Proof: If there is at least one coefficient greater than θ with degree lower than ℓ for the parity function of vector starting with α then $S^\rho_\alpha(f)$ is at least $(1-2\rho)^{2\ell}\theta^2 = 2^{2\log(1-2\rho)\ell}\theta^2 = 2^{-c_0\ell}\theta^2$. All the coefficients for the vectors starting with α can be separated into two disjoint sets—those for the vectors that start with $\alpha0$ and those for the vectors that start with $\alpha1$. With these observations and the procedure `P1` for estimating $S^\rho_\alpha(f)$ we can write a recursive subroutine `P2` that for every α outputs the set W_α that satisfies the following conditions:

- for all $a \in W_\alpha$,α is a prefix of a;
- for all $a \in \{0,1\}^n$, if $|\hat{f}(a)| \geq \theta$ and $w(a) \leq \ell$ then $a \in W_\alpha$;
- for all $a \in W_\alpha$, $|\hat{f}(a)| \geq \theta/2$.

That is, W_α has the large Fourier coefficient property for function f, threshold θ, and degree ℓ; and is restricted to prefix α.

To find all the required coefficients (i.e., to implement the $\mathtt{BS}_\rho$) we invoke `P2` on the empty prefix. The implementation of the procedure `P2` is given in Figure 2.

If `P1` succeeds then `P2` is called recursively only when

$$S^\rho_\alpha(f) \geq (3/4 - 1/4)2^{-c_0\ell}\theta^2 = 2^{-c_0\ell}\theta^2/2 \ .$$

Since $\sum_{\alpha\in\{0,1\}^k} S^\rho_\alpha(f) \leq \sum_a \hat{f}^2(a) = 1$, `P2` will be invoked at most $2 \cdot 2^{c_0\ell}\theta^{-2}$ times for each length of α, and therefore there will be at most $2n2^{c_0\ell}\theta^{-2}$ calls to `P1`. This means that total probability of failure of the algorithm is at most $2n2^{c_0\ell}\theta^{-2} \cdot \delta 2^{-c_0\ell}\theta^2/(2n) = \delta$. According to Lemma 11, each call to `P1` in `P2` takes

$$O(2^{2c_0\ell}\theta^{-4}\log(2n2^{c_0\ell}\theta^{-2}\delta^{-1})) = \tilde{O}(2^{2c_0\ell}\theta^{-4}\log(1/\delta)) \ .$$

This gives a bound on total running time of $\tilde{O}(n2^{3c_0\ell}\theta^{-6}\log(1/\delta))$. By the same lemma, tolerance of all the queries is bounded from below by $2^{-c_0\ell}\theta^2/16$. It is important to note

Input: Access to $\mathrm{STAT}(f, \mathcal{D}_\rho)$; $0 \le k \le n$; $\alpha \in \{0,1\}^k$; $\ell \le n$; $\theta > 0$; $\delta > 0$
Output: The set W_α that, with probability at least $1-\delta$, has the large Fourier coefficient property for function f, threshold θ and degree ℓ; and is restricted to prefix α.

1. **if** $k = n$ **then**
2. $\quad v \leftarrow \mathrm{STAT}(f, \mathcal{D}_\rho)[f\chi_\alpha, \theta/4, U]$
3. $\quad$ **if** $v \ge \frac{3}{4}\theta$ **then output**(α)
4. **else**
5. $\quad b \leftarrow 2^{-c_0\ell}\theta^2$
6. $\quad s \leftarrow$ `P1`$(n, \alpha, b/4, \delta b/(2n))$
7. $\quad$ **if** $s \ge 3b/4$ **then**
8. $\quad\quad$ `P2`$(k+1, \alpha 0, \ell, \theta, \delta)$
9. $\quad\quad$ `P2`$(k+1, \alpha 1, \ell, \theta, \delta)$
10. $\quad$ **endif**
11. **endif**

Figure 2: Procedure `P2`

that if the estimations fail the running time of the algorithm may exceed this bound. This can be easily prevented by adding a time counter that terminates the execution of the algorithm whenever it exceeds this time bound. □

In our future applications we will need to find the Fourier coefficients not only of a Boolean target f but also of any real-valued function involving $f(x)$. Thus we extend the `BS` algorithm as follows.

Theorem 13 *Let $\phi : \{0,1\}^n \times \{-1,1\} \to [-\beta, \beta]$ be any real-valued function ($\beta > 0$). There exists an SQ–$\mathcal{D}_\rho$ randomized algorithm `RV-BS`$_\rho$ that for every target concept f, positive threshold θ, confidence $\delta > 0$ and degree bound $0 \le \ell \le n$ `RV-BS`$_\rho(n, \phi, \beta, \theta, \ell, \delta)$ returns, with probability at least $1-\delta$, a set with the large Fourier coefficient property for function ϕ, threshold θ and degree ℓ. It runs in $\tilde{O}(n 2^{3c_0\ell}\theta^{-6}\beta^2 \log(1/\delta))$ time and tolerance of queries is bounded from below by $\tilde{\Omega}(\beta^{-2}2^{-c_0\ell}\theta^2)$. Complexity of query functions is bounded by $O(\tau) + O(n)$, where τ is the complexity of computation of ϕ.*

Proof: Let us look back into the proof of Theorem 12. Its idea is obviously applicable in the new setting. All we have to check is the details of the implementation. Let us substitute $\pi(x) \triangleq \phi(x, f(x))$ for $f(x)$ in all the places in the proof. In the second line of `P2` we need an estimate of $\hat{\phi_f}(\alpha)$. This cannot be done using one statistical query but instead we can use the procedure `RV-SQ`$(\mathrm{STAT}(f, U), \phi\chi_\alpha(x), \theta/4)$. Now, given that `P1` will estimate correctly $S^\rho_\alpha(\pi)$, the algorithm will work correctly. The only thing that will change is the bound on the number of calls to `P2` since it is based on the fact that: $\mathbf{E}[f^2] = 1$. Now we have that: $\mathbf{E}[\pi^2] \le L^2_\infty(\pi) \le \beta^2$ and hence `P2` will be invoked at most $\beta^2 2n 2^{c_0\ell}\theta^{-2}$ times. Due to the increase in the number of calls to `P1` the confidence of each call has to be increased (it has to be divided by β^2). Now we should get a modified version of `P1` that will produce estimations of $S_\alpha^\rho(\pi)$. If we substitute π for f in Lemma 11 we see that all the properties will hold but now the estimation of $\pi_\alpha^\rho(z) = \mathbf{E}_{x\sim D_{p\oplus z}}[\pi(x)\chi_{\alpha_0}(z\oplus x)]$ (line 3 in `P1` cannot be done using one query. Moreover, since we want $(\pi_\alpha^\rho(z))^2$ to be approximated within $\tau/2$, $\pi_\alpha^\rho(z)$ has to be approximated within $\frac{\tau}{4|\pi_\alpha^\rho(z)|}$. Since

$$|\pi_\alpha^\rho(z)| \le \mathbf{E}_{x\sim D_{p\oplus z}}[|\pi(x)\chi_{\alpha_0}(z\oplus y)|] \le L_\infty(\phi) \le \beta$$

it will suffice to approximate $\pi_\alpha^\rho(z)$ within $\tau' = \frac{\tau}{4\beta}$. To make this approximation we call

$$\texttt{RV-SQ}(\mathrm{STAT}(f, D_{p\oplus z}), \phi(x, f(x))\chi_{\alpha_0}(z\oplus x), \beta, \tau')\ .$$

By Lemma 5, we get the required value. The time complexity of the call to `RV-SQ` is $O(\log(\beta/\tau')) = O(\log(\beta/\tau))$. The required tolerance is at least $\frac{\tau'}{4\beta\log(\beta/\tau')} = \tilde{\Omega}(\frac{\tau}{\beta^2})$. Complexity of the query functions is the complexity of computation of ϕ plus $O(n)$. These new bounds on the complexity of `P1` render the total running time of the algorithm to be $\tilde{O}(n2^{3c_0\ell}\theta^{-6}\beta^2\log(1/\delta))$. The required tolerance is $\tilde{\Omega}(\beta^{-2}2^{-c_0\ell}\theta^2)$. Thus all the bounds satisfy the required conditions. $\square$

3.4 Learning Decision Trees

Below we give the first application of the $\mathtt{BS}_\rho$ algorithm. We prove that the representation class of decision trees is learnable in the SQ–$\mathcal{D}_\rho$ model. For this and further applications we take $\rho = \frac{1}{4}$. This makes $c_0 = 2\log(1-2\rho)$ to be 2. Our idea is to show that in order to learn `DT` it is sufficient to consider only "large" Fourier coefficients of "low" degree (we add a restriction on degree over the original proof by Kushilevitz and Mansour). To achieve this we rely on several well-known properties of Fourier transform of decision trees. The properties are described in the following lemmas (we provide their proofs for completeness).

Lemma 14 *Let T be a decision tree of depth d. All the Fourier coefficients of T of degree larger than d are 0 and all the non-zero coefficients of T are larger than 2^{-d}.*

Proof: We start by examining the Fourier transform of a single term. Let t be a term of length l and let $i_1,\ldots,i_l$ be the indices of variables appearing as literals in t. For every $1\le j\le l$ let b_j be 1 if x_{i_j} appears negated in t and 0 otherwise. Let $t^+ = \frac{t+1}{2}$ (the $\{0,1\}$ version of term t). It is easy to verify that

$$t^+ = \prod_{j=1}^{l}\left(\frac{1+(-1)^{b_j}\chi_{\{i_j\}}}{2}\right) \tag{6}$$

For every two disjoint sets of indices A and B, $\chi_A\chi_B = \chi_{A\cup B}$. Thus by developing the equation (6) we obtain that all the non-zero Fourier coefficients of t^+ are for sets not larger than l and are by their absolute value equal to 2^{-l}.

Let m be the number of leaves labelled by 1 in T and $t_1,\ldots,t_m$ be the terms corresponding to each of these leaves. Let $T^+, t_1^+,\ldots,t_m^+$ be $\{0,1\}$ versions of the decision tree and the terms. Every input value satisfies at most one term of T and thus $T^+ = \sum_{i=1}^m t_i^+$, that is, every Fourier coefficient of T^+ is a sum of corresponding Fourier coefficients of terms $t_1^+,\ldots,t_m^+$. Length of every term t_i is at most d and thus all the non-zero Fourier coefficients of T^+ are for sets not larger than d and their absolute value is greater or equal to 2^{-d}. By the definition of T^+, $T = 2T^+ - 1$ and therefore the above property of the Fourier transform of T^+ holds for T as well. □

The size of decision tree is the number of leaves in it. Let *DT-size*(f) denote the size of a minimal DT representation of f.

Lemma 15 *Let T be a decision tree of size m. There exists a function h such that all the Fourier coefficients of h of degree larger than t are 0, where $t = \log(4m/\tau)$, all non-zero coefficients of h are larger than $\tau/(4m)$ and $\mathbf{E}[(T-h)^2] \leq \tau$.*

Proof: Let h be the decision tree that results from truncating T at depth t. At the truncated places we add a leaf with value 1. The probability that uniformly chosen assignment will reach some specific leaf at depth greater or equal to t is at most 2^{-t}. Therefore the probability of reaching one of the leaves we added (instead of the truncated parts) is at most $m2^{-t} = \tau/4$, therefore $\mathbf{Pr}_U[T \neq h] \leq \tau/4$. For Boolean functions h and T this implies that

$$\mathbf{E}_U[(T-h)^2] = 4\ \mathbf{Pr}_U[T \neq h] \leq \tau\ .$$

By Lemma 14, we get that the Fourier transform of h has the required properties.

We are now ready to prove that `BS` can be used to learn `DT`.

Theorem 16 *There exists an SQ–$\mathcal{D}_{\frac{1}{4}}$ randomized algorithm* `DT-learn` *such that for any target concept f, $s =$ DT-size(f), $\epsilon > 0$ and $\delta > 0$,* `DT-learn`(n, s, ϵ, δ) *returns, with probability at least $1-\delta$, an ϵ–approximation of f. The algorithm runs in $\tilde{O}(ns^{12}\epsilon^{-12}\log(1/\delta))$ time. Tolerance of its queries is $\Omega(\epsilon^4 s^{-4})$.*

Proof: Given its input `DT-learn` gets $H = \mathtt{BS}_{\frac{1}{4}}(n, \frac{\epsilon}{8s}, \log\frac{8s}{\epsilon}, \delta)$. Denote $g = \sum_{a\in H}\tilde{f}(a)$, where $\tilde{f}(a)$ is an estimate of $\hat{f}(a)$ satisfying $|\tilde{f}(a) - \hat{f}(a)| \leq \sqrt{\frac{\epsilon}{2}}\frac{\epsilon}{16s}$ (such an estimate can be obtained using query $(f\chi_a, \sqrt{\frac{\epsilon}{2}}\frac{\epsilon}{2s}, U)$). Then the algorithm returns $h = sign(g)$. We need to prove that (given that `BS` succeeds) h calculated in this fashion is an $\epsilon-$approximation of f. By Fact 6, $\mathbf{Pr}[h \neq f] \leq \mathbf{E}[(f-g)^2]$, thus it is sufficient to prove that

$$\mathbf{E}[(f-g)^2] = \sum_{a\in\{0,1\}^n}(\hat{f}(a) - \hat{g}(a))^2 \leq \epsilon\ .$$

Denote

$$V = \{a \mid \hat{f}(a) \leq \frac{\epsilon}{8s} \vee w(a) \geq \frac{8s}{\epsilon}\}\ .$$

Clearly, $H \cup V = \{0,1\}^n$. Thus

$$\mathbf{Pr}[h \neq f] \leq \mathbf{E}[(f-g)^2] = \sum_{a\in\{0,1\}^n}(\hat{f}(a) - \hat{g}(a))^2 =$$

$$\sum_{a\in H}(\hat{f}(a) - \tilde{f}(a))^2 + \sum_{a\notin H}\hat{f}^2(a) \leq \sum_{a\in H}(\hat{f}(a) - \tilde{f}(a))^2 + \sum_{a\in V}\hat{f}^2(a) \qquad (7)$$

Since every coefficient in H is at least $\frac{\epsilon}{16s}$, $|H|$ is at most $(\frac{16s}{\epsilon})^2$ and therefore the defined g satisfies :

$$\sum_{a\in H}(\hat{f}(a)-\hat{g}(a))^2 = \sum_{a\in H}(\hat{f}(a)-\tilde{f}(a))^2 \leq \epsilon/2 \ ,$$

that is, the first term of equation (7) is at most $\epsilon/2$. As is proved in Lemma 15 (for $m=s$ and $\tau=\epsilon/2$), there exists a function f' such that all the Fourier coefficients of f' for vectors in V are 0 and $\mathbf{E}[(f-f')^2] \leq \epsilon/2$. That is,

$$\epsilon/2 \geq \mathbf{E}[(f-f')^2] = \sum_{a\in\{0,1\}^n}(\hat{f}(a)-\hat{f'}(a))^2 \geq \sum_{a\in V}\hat{f}^2(a) \ .$$

Altogether, $\mathbf{Pr}[h \neq f] \leq \epsilon$. By properties of $\mathtt{BS}_{\frac{1}{4}}$ (see Theorem 12), the running time of the algorithm is $\tilde{O}(ns^{12}\epsilon^{-12}\log(1/\delta))$ and tolerance of its queries is $\Omega(\epsilon^4 s^{-4})$. □

Remark 17 *It can be easily seen that the size parameter is used only for "bounding" purposes and therefore if the parameter is not supplied to* `DT-learn` *we can use the standard guess-and-double technique instead. This remark is applicable to the rest of our algorithms as well.*

3.5 Weak DNF Learning

Another simple application of the Bounded Sieve algorithm is weak DNF learning in SQ–$\mathcal{D}_{\frac{1}{4}}$. It is based on the fact that every DNF expression has a "large" Fourier coefficient of "low" degree [BFJ+94; Ja97]. Below we prove a generalization of this fact that will also be required for our future application. Let *DNF-size*(f) denote the size (number of terms) of a minimal DNF representation of f.

Lemma 18 *Let f be a Boolean function, $s =$ DNF-size(f) and D be a distribution over X. There exists a parity function χ_a such that $E_D[f\chi_a] \geq \frac{1}{2s+1}$ and $w(a) \leq \log((2s+1)L_\infty(2^n D))$.*

Proof: As it was proved by Jackson [Ja97, Fact 8], there exists a parity function χ_a such that $E_D[f\chi_a] \geq \frac{1}{2s+1}$. Let $A = \{i \mid a_i = 1\}$. As it can be seen from the proof of the fact, $A \subseteq T$, where T is a term of f (a set of literals and a Boolean function it represents) and $\mathbf{Pr}_D[T=1] \geq \frac{1}{2s+1}$. On the other hand,

$$\mathbf{Pr}_D[T=1] = \sum_{x, T(x)=1} D(x) < 2^{-|T|}2^n L_\infty(D) \ .$$

Thus

$$w(a) = |A| \leq |T| \leq \log((2s+1)L_\infty(2^n D)) \ .$$

□

Theorem 19 *There exists an SQ–$\mathcal{D}_{\frac{1}{4}}$ randomized algorithm* `UWDNF` *such that for any target concept f, $s =$ DNF-size(f), $\epsilon > 0$ and $\delta \geq 0$,* `UWDNF`(n, s, δ) *returns, with probability at* *least* $1-\delta$, *a* $(\frac{1}{2}-\frac{1}{4s+4})$*–approximation to* f*. The algorithm runs in* $\tilde{O}(ns^12\log(1/\delta))$ *time. Tolerance of its queries is* $\Omega(s^{-4})$*. The weak hypothesis is a parity function (possibly negated).*

Proof: By Lemma 18, a call to $\mathtt{BS}_{\frac{1}{4}}(n,\theta=\frac{1}{2s+1},\ell=\log(2s+1),\delta)$ returns, with probability at least $1-\delta$, a set containing at least one vector. We estimate every coefficient in the returned set with tolerance $r=\frac{1}{(2s+1)(2s+2)}$ and choose the vector with the maximal Fourier coefficient (it will be larger than $\frac{1}{2s+1}-r=\frac{1}{2s+2}$). Parity function for this vector (or its negation) will give a $(\frac{1}{2}-\frac{1}{4s+4})$–approximation to the target. By Theorem 12 the running time of this invocation will be $\tilde{O}(ns^{12}\log(1/\delta))$ and tolerance of queries is $\Omega(s^{-4})$.

3.6 Boosting the Weak DNF Learner

Certainly, the next interesting question is whether we can strongly learn DNF in SQ–$\mathcal{D}_{\frac{1}{4}}$. We answer this question positively by following the way used by Jackson in his proof of DNF learnability. The proof consists of two components. The first one is weak DNF learning with respect to any given distribution (although efficient only on distributions that are polynomially close to uniform). The second one is Freund's boosting technique that boosts the weak learner to a strong learner.

First we present the required weak learner.

Theorem 20 *Let* f *be a Boolean function,* $s=$ *DNF-size*(f) *and let* D_f *be a computable probability distribution over the sample space. The computation of* $D_f(x)$ *may involve the value of* $f(x)$*. There exists a randomized SQ–*$\mathcal{D}_{\frac{1}{4}}$ *algorithm* $\mathtt{WDNF}_{\frac{1}{4}}$ *such that for the target function* f*, confidence* $\delta>0$ *and* β *that bounds* $L_\infty(2^nD_f)$*,* $\mathtt{WDNF}_{\frac{1}{4}}(n,s,D_f,\beta,\delta)$ *finds, with probability at least* $1-\delta$*, a Boolean function* h *that* $(\frac{1}{2}-\frac{1}{4s+4})$*–approximates* f*. The algorithm runs in* $\tilde{O}(n\beta^8s^{12}\log(1/\delta))$ *time. The tolerance of its queries is lower bounded by* $\tilde{\Omega}(\beta^{-4}s^{-4})$*. Complexity of its query functions is the time complexity of* D_f *plus* $O(n)$*. The weak hypothesis is a parity function(possibly negated).*

Proof: As is proved in Lemma, 18 for every DNF expression f, there exists a parity function χ_b such that $|\mathbf{E}_{D_f}[f\chi_b]|\geq\frac{1}{2s+1}$ and

$$w(b)\leq\log\left((2s+1)L_\infty(2^nD_f)\right)\leq\log\left((2s+1)\beta\right)\ .$$

$\mathbf{E}_{D_f}[f\chi_b]=\mathbf{E}_U[2^nD_ff\chi_b]$ thus in order to find a parity function that weakly approximates f with respect to D_f we can find the "large" Fourier coefficients of function 2^nD_ff. This means that by invoking

$$\mathtt{RV\text{-}BS}_{\frac{1}{4}}(n,\phi=2^nD_ff,\beta,\theta=\frac{1}{2s+1},\ell=\log\left((2s+1)\beta\right),\delta)$$

and then proceeding as in UWDNF we can find the required weak approximator. By Theorem 13, we get that all the complexities are as stated. □

The next step is adapting Freund's boosting technique to the SQ–$\mathcal{D}_\rho$ model. The main advantage of Freund's booster F1 utilized by Jackson is that it requires only weak learning with respect to polynomially computable distributions that are polynomially close to the uniform distribution (i.e., $L_\infty(2^n D) \le p(n, s, \epsilon)$ for some polynomial p). Freund's boosting algorithm is not the only boosting algorithm possessing this advantage and is not the most efficient one (see discussion below). But its presentation is relatively simple and is sufficient for demonstrating the idea of adaptation.

Below we are going to give a brief account of Freund's boosting technique [Fr90] and in particular his `F1` algorithm for the PAC learning model with respect to distribution D.

As input, `F1` is given positive ϵ, δ, and γ, a weak learner WL that produces $(\frac{1}{2} - \gamma)$–approximate hypotheses for functions in a function class $\mathcal{F}$, and an example oracle $\mathrm{EX}(f, D)$ for some $f \in \mathcal{F}$. The WL is assumed to take the example oracle for some distribution D' and confidence parameter δ' as inputs and produce $(\frac{1}{2} - \gamma)$–approximate hypothesis with probability at least $1-\delta'$. Given these inputs, `F1` steps sequentially through k stages (k is given in Figure 3). At each stage i, $0 \le i \le k-1$ `F1` generates distribution function D_i, runs WL on simulated oracle $\mathrm{EX}(f, D_i)$ and (with high probability) gets a $(\frac{1}{2} - \gamma)$–approximate hypothesis w_i. Simulation of the example oracle is done by filtering. Probability of acceptance of the next example is estimated before the simulation (to E_α). If the probability is "too low" the algorithm does not produce any more weak hypotheses. Finally, `F1` generates its hypothesis using majority function MAJ of all the w_i it has received.

In Figure 3 we give the original implementation of Freund's boosting algorithm `F1`.

Theorem 21 *DNF is strongly learnable in the SQ–$\mathcal{D}_\rho$ model.*

Proof: The first and simple step towards the adaptation of Freund's algorithm to SQ–$\mathcal{D}_{\frac{1}{4}}$ is estimation of E_α using the procedure `RV-SQ` instead `AMEAN` (since we do not have access to $\mathrm{EX}(f, D)$). This is possible since the value of $\alpha^i_{r_i(x)}$ can be found using $f(x)$ and previous hypotheses. Next, we show how to provide our weak learner with distribution D_i. By the definition, $D_i(x) \equiv \frac{D(x)\alpha^i_{r_i(x)}}{\sum_y D(y)\alpha^i_{r_i(y)}}$. It is easy to see that given $f(x)$ the value of $D(x)\alpha^i_{r_i(x)}$ can be evaluated straightforwardly. On the other hand, the value of $\sum_y D(y)\alpha^i_{r_i(y)}$ (independent of x), which is the probability of accepting a sample from D, cannot be computed exactly. Instead, we can estimate this value. In fact, we already have the estimate: the value E_α used for termination condition (lines 12–14). The condition ensures that $E_\alpha \ge \frac{2}{3}\theta$ (otherwise we terminate) and E_α is an estimate of $\sum_y D(y)\alpha^i_{r_i(y)}$ within $1/3\theta$ (line 11). Thus

$$1/2E_\alpha \le \sum_y D(y)\alpha^i_{r_i(y)} \le 3/2E_\alpha \ .$$

This means that if we define $D'_i(x) \equiv U(x)\alpha^i_{r_i(x)}/E_\alpha$ then we get that there is a constant $c_i \in [1/2, 3/2]$ such that for all x, $D'_i(x) = c_i D_i(x)$. Now consider the functional impact of supplying this approximate distribution rather than true distribution to $\mathtt{WDNF}_{\frac{1}{4}}$. $\mathtt{WDNF}_{\frac{1}{4}}$ uses its given distribution for exactly one purpose: to find the "large" Fourier coefficients of function $2^n D_f f$. Since the Fourier transform is a linear operator (i.e., $\widehat{cg}(a) = c\hat{g}(a)$) we can be certain that $2^n D'_i f$ has a Fourier coefficient with absolute value of at least $c_i \frac{1}{2s+1} \ge \frac{1}{4s+2}$. Moreover, since the largest Fourier coefficient of $2^n D'_i f$ corresponds to the largest Fourier coefficient of $2^n D_i f$, the parity function corresponding to that coefficient (or its negation)

Input: Example oracle $EX(f, D)$; $0 < \gamma \leq \frac{1}{2}$; $(\frac{1}{2} - \gamma)$–approximate weak learner WL(EX, δ) which succeeds with probability at least $1 - \delta$; $\epsilon > 0$; $\delta > 0$
Output: h such that, with probability at least $1 - \delta$, $\mathbf{Pr}_D[f = h] \geq 1 - \epsilon$

$\hat{\gamma} \equiv \min(\gamma, 0.39)$
$k \leftarrow \frac{1}{2}\gamma^{-2} \ln(4/\epsilon)$
$w_0 \leftarrow WL(EX(f, D), \delta/2k)$
for $i \leftarrow 1, \ldots, k-1$ **do**
  $r_i(x) \equiv |\{0 \leq j < i | w_j(x) = f(x)\}|$
  $B(j; n, p) \equiv \binom{n}{j} p^j (1-p)^{n-j}$
  $\tilde{\alpha}^i_r \equiv B(\lfloor k/2 \rfloor - r; k - i - 1, 1/2 + \hat{\gamma})$ if $i - k/2 < r \leq k/2$, $\tilde{\alpha}^i_r \equiv 0$ otherwise
  $\alpha^i_r \equiv \tilde{\alpha}^i_r / \max_{r'=0,\ldots,i-1}\{\tilde{\alpha}^i_{r'}\}$
  $\theta \equiv \epsilon^3/57$
  $Y \equiv$draw example $\langle x, f(x) \rangle$ from EX(f, D) and compute $\alpha^i_{r_i(x)}$
  $E_\alpha \leftarrow$ `AMEAN`$(Y, b - a = 1, \frac{1}{3}\theta, \frac{\delta}{2k})$
  if $E_\alpha \leq \frac{2}{3}\theta$ **then**
    $k \leftarrow i$
    break do
  endif
  $D_i(x) \equiv \frac{D(x)\alpha^i_{r_i(x)}}{\sum_y D(y)\alpha^i_{r_i(y)}}$
  $w_i \leftarrow WL(EX(f, D_i), \frac{\delta}{2k})$
enddo
$h(x) \equiv MAJ(w_0(x), w_1(x), ..., w_{k-1}(x))$
return h

Figure 3: Freund's algorithm for boosting a weak learner in the PAC model. EX(D_i, f) is a simulated example oracle.

$(\frac{1}{2} - \frac{1}{4s+2})$–approximates target function with respect to D_i. Thus by slightly modifying [7] $\texttt{WDNF}_{\frac{1}{4}}$ we can handle this problem. These modifications may increase the running time of weak learner only by a small constant and thus do not affect our analysis.

Input: Access to $STAT(f, \mathcal{D}_{\frac{1}{4}})$; $s = DNF\text{-}size(f)$; $\epsilon > 0$; $\delta > 0$
Output: h such that, with probability at least $1 - \delta$, $\mathbf{Pr}_U[f = h] \geq 1 - \epsilon$

$\gamma \leftarrow \frac{1}{4s+4}$;
$k \leftarrow \frac{1}{2}\gamma^{-2} \ln(4/\epsilon)$
$w_0 \leftarrow \texttt{UWDNF}_{\frac{1}{4}}(n, s, \delta/k)$
for $i \leftarrow 1, \ldots, k-1$ **do**
  $r_i(x) \equiv |\{0 \leq j < i | w_j(x) = f(x)\}|$
  $B(j; n, p) \equiv \binom{n}{j} p^j (1-p)^{n-j}$
  $\tilde{\alpha}^i_r \equiv B(\lfloor k/2 \rfloor - r; k - i - 1, 1/2 + \hat{\gamma})$ if $i - k/2 < r \leq k/2$, $\tilde{\alpha}^i_r \equiv 0$ otherwise
  $\alpha^i_r \equiv \tilde{\alpha}^i_r / \max_{r'=0,\ldots,i-1}\{\tilde{\alpha}^i_{r'}\}$
  $\theta \leftarrow \epsilon^3/57$
  $E_\alpha \leftarrow \texttt{RV-SQ}(\mathrm{STAT}(f, U), \alpha^i_{r_i(x)}, 1, \theta/3)$
  if $E_\alpha \leq \frac{2}{3}\theta$ **then**
    $k \leftarrow i$
    break do
  endif
  $D'_i(x) \equiv U(x)\alpha^i_{r_i(x)}/E_\alpha$
  $w_i \leftarrow \texttt{WDNF}_{\frac{1}{4}}(n, s, D'_i, 3/(2\theta), \delta/k)$
enddo
$h(x) \equiv MAJ(w_0(x), w_1(x), \ldots, w_{k-1}(x))$
return h

Figure 4: Learning DNF in the SQ–$\mathcal{D}_{\frac{1}{4}}$ model

In Figure 4 we give a straightforward implementation of the discussed adaptation.

Our last concern is the complexity of this algorithm. The total number of phases executed will be $O(s^2 \log \epsilon^{-1})$. Clearly the "heaviest" part of each phase is the execution of the weak learner. By Theorem 20, the running time of each call to $\texttt{WDNF}_{\frac{1}{4}}$ is $\tilde{O}(ns^{12}\epsilon^{-24}\log(1/\delta))$. Thus the total running time of the algorithm is $\tilde{O}(ns^{14}\epsilon^{-24}\log(1/\delta))$. The tolerance of queries is $\tilde{\Omega}(s^{-4}\epsilon^{12})$. The complexity of query functions is as complexity of evaluation of $\alpha^i_{r_i(x)}$ plus $O(n)$. All the $O(k^2) = \tilde{O}(s^4)$ possible values of $\alpha^i_{r_i(x)}$ can be evaluated in advance, that is complexity of query functions will be $O(n)$. $\square$

The adaptation of boosting algorithm that we obtained is not suitable for boosting weak learners which will fail when supplied with distributions D'_i as above instead of D_i. Except

7. Lower bound θ has to be modified to $\frac{1}{4s+2}$.

for this limitation we, in fact, described the general way to boost weak learners in the SQ–$\mathcal{D}$ model.

Recently, two new boosting algorithms have appeared that use only distributions that are polynomially close to the uniform. The first one is by Klivans and Servedio [KS99] and the second one is by Bshouty and Gavinsky [BG01]. Distributions produced by these boosting algorithms are optimally "flat", that is, the L_∞ norm of distributions they produce is $\tilde{O}(\epsilon)2^{-n}$. They also require the same order of phases as `F1`. Both algorithms are more complicated than `F1`, but nevertheless can be adapted to SQ–$\mathcal{D}$ in a very similar way. These algorithms were used to improve the running time of Jackson's original algorithm and can be substituted for `F1` in our algorithm as well. This would significantly decrease the dependence on ϵ. In particular, $L_\infty(D_i) = \tilde{O}(\epsilon^{-1})$ and $k = O(\gamma^{-2}\log(1/\epsilon))$ gives the running time of $\tilde{O}(ns^{14}\epsilon^{-8}\log(1/\delta))$ and tolerance becomes lower-bounded by $\tilde{\Omega}(s^{-4}\epsilon^4)$.

4. Learning with Attribute Noise in Membership Queries

We now turn our attention to another type of noise in membership queries: the noise that corrupts the attributes of a sample point. That is, as an answer to a membership query at point x the learner gets the value of $f(x')$, where x' might differ from x. This type of noise was previously considered as appearing in the example oracle $\mathrm{EX}(f, D)$, that is the learner gets random labelled samples $(x, f(x'))$ [SV88; GS95; BJT99]. We define the noise model formally and then focus our attention on the product attribute noise with known noise rate (which is the most commonly referred type of attribute noise). We prove that membership queries corrupted by such a noise are weaker than extended statistical queries discussed in the previous chapter. Nevertheless, the Bounded Sieve can be implemented in this model. This implies that DT is learnable and DNF is weakly learnable in this "weak" model. We prove that these learning results can also be achieved for the uniform product attribute noise of unknown rate.

4.1 The Noise Model and Its Properties

The attribute noise is usually described as returning the value $f(x \oplus b)$ where b is sampled randomly with respect to some noise distribution D'. The simplest and the most well studied noise distribution is the product distribution, i.e., independence of all the attributes is assumed.[8] If every attribute is changed with the same probability the noise is called *uniform*. The resulting model represents the situation in which the learner cannot ask the membership query in the desired point since the attributes it supplies change their state randomly and independently in the process of the query. Examples of situations of this kind are:

- sending the point x to the oracle via a faulty communication channel[9]
- "material" representing the attributes is unstable.

8. This assumption certainly agrees with assumptions of learning with respect to the uniform.
9. For this case the attribute noise is uniform and random classification noise has to be added if the answer is also sent via a faulty channel.

Formally, the noise model is defined in the following way. For a constant probability vector $p \in (0,1)^n$ and D_p the product distribution defined by p, we define noisy membership oracle $\mathrm{MEM}^p(f)$ as follows. For a query x, $\mathrm{MEM}^p(f)$ chooses randomly (and independently of any previous queries) b according to the distribution D_p and returns $f(x \oplus b)$. We can assume that for all i, $p_i \leq 1/2$ since inverting the i-th attribute of the query is equivalent to changing p_i to $1 - p_i$.

We may also add classification noise to the oracle and define $\mathrm{MEM}^{p,\eta}(f)$. This classification noise does not have to be persistent as the learner cannot get a label of the same point twice (with any non-negligible probability).

The use of membership queries with product attribute noise can be related to the SQ–$\mathcal{D}_\rho$ model in the following way.

Theorem 22 *For $p \in [\rho, 1/2]^n$ MEM^p can be efficiently simulated in SQ–$\mathcal{D}_\rho$.*

Proof: The result of query at point x to oracle MEM^p is a Bernoulli random variable (or a coin flip) such that $e \in \{1, -1\}$ is returned with probability $\mathbf{Pr}_{y \sim D_p}[f(x \oplus y) = e]$. Given these probabilities for every x, MEM^p can be replaced by an appropriate coin flip. Moreover, the exact values of probabilities are not required, since the ability to approximate them efficiently will suffice to simulate the MEM^p oracle. But

$$\mathbf{Pr}_{y \sim D_p}[f(x \oplus y) = e] = \mathbf{Pr}_{y \sim D_{p \oplus x}}[f(y) = e] \ ,$$

i.e., this probability can be estimated in SQ–$\mathcal{D}$ whenever $D_{p \oplus x} \in \mathcal{D}$. Thus for $\mathcal{D} = \{D_{p \oplus x} \mid x \in \{0,1\}^n\}$ MEM^p can be simulated in SQ–$\mathcal{D}$. Such $\mathcal{D}$ is included in the class of probability distributions which for every i, gives a learning algorithm a degree of control p_i over attribute i (see Remark 10 for details about the definition). For every i, $p_i \in [\rho, 1/2]$ and thus by Lemma 9 and Remark 10, MEM^p can be simulated in SQ–D_ρ. $\square$

4.2 Offsetting Attribute Noise in the Bounded Sieve

Let $\tau > 0$ be a constant and $p \in [0, 1/2 - \tau]^n$ be a probability vector. We are now going to show that we can implement the Bounded Sieve using access to MEM^p oracle (we call the resulting algorithm `AN-BS` where AN stands for attribute noise).

Theorem 23 *There exists a randomized algorithm* `AN-BS` *that for any target concept f, threshold $\theta > 0$, degree bound $0 \leq \ell \leq n$, confidence $\delta > 0$ and probability vector p (with restrictions as above),* `AN-BS`$(MEM^p, n, \theta, \ell, \delta, p)$ *returns, with probability at least $1 - \delta$, a set with the large Fourier coefficient property for function f, threshold θ and degree ℓ. Its running time is polynomial in $n, \theta^{-1}, 2^\ell$ and $\log(1/\delta)$.*

Proof: A query to the oracle $\mathrm{MEM}^p(f)$ at point y returns $f(y \oplus b)$ where b is chosen randomly according to the distribution D_p. Thus by Hoeffding's formula we can efficiently approximate $f^p(y) \triangleq \mathbf{E}_{b \sim D_p} f(y \oplus b)$. Obviously, for every x, $|f^p(x)| \leq 1$ and thus $\|f^p\| \leq 1$. Now, by using the real-valued version of the `KM` algorithm (see Remark 7 in Section 2.5) we can find a set with the large Fourier coefficients property for function f^p and threshold ϑ

in time polynomial in n, ϑ^{-1} and $\log(1/\delta)$. But

$$f^p(y) = \mathbf{E}_{b\sim D_p}[f(y\oplus b)] = \sum_{a\in\{0,1\}^n} \hat{f}(a)\mathbf{E}_{b\sim D_p}[\chi_a(y\oplus b)] = \sum_{a\in\{0,1\}^n} \left(\hat{f}(a)\mathbf{E}_{b\sim D_p}[\chi_a(b)]\right)\chi_a(y),$$

i.e., we have that for all a, $\hat{f^p}(a) = \hat{f}(a)\mathbf{E}_{b\sim D_p}\chi_a(b)$. We can easily calculate the value $c_a \triangleq \mathbf{E}_{b\sim D_p}\chi_a(b) = \prod_{a_i=1}(1-2p_i)$. By the properties of p, $|c_a| \geq (2\tau)^{w(a)}$. Thus if we run the above algorithm for the threshold $\vartheta = (2\tau)^\ell\theta$ we will get the set S' that includes a set with the large Fourier coefficient property for the function f and threshold θ. Size of S' is bounded by $(2\tau)^{-2\ell}\theta^{-2}$. In order to estimate the values of the coefficient $\hat{f}(a)$ with accuracy σ we estimate the coefficient $\hat{f^p}(a)$ with accuracy $\sigma c_a \geq (2\tau)^\ell\sigma$ and return $c_a^{-1}\hat{f^p}(a)$. Thus we can refine the set S' and return a set with the large Fourier coefficient property for the function f, threshold θ and degree ℓ. All the parts of the described algorithm run in time polynomial in $n, (2\tau)^{-\ell}\theta^{-1}$ and $\log(1/\delta)$, that is, for a constant τ the time is polynomial in $n, \theta^{-1}, 2^\ell$, and $\log(1/\delta)$ □

Remark 24 *It is easy to note that knowing the noise rates exactly is not necessary. That is, the result will hold when the learning algorithm is supplied with "good" estimates of the noise rates. Particularly, estimates within inverse of a polynomial (in learning parameters) will suffice.*

Remark 25 *When using MEM$^{p,\eta}(f)$ oracle instead of MEM$^p(f)$ in the above algorithm we can immediately offset the classification noise by using the fact: $f^{p,\eta}(y) = (1-2\eta)f^p(y)$, where $f^{p,\eta}(y)$ is the expectation of the query in point y to the oracle MEM$^{p,\eta}(f)$. This will require increasing the accuracy of the estimation by $\frac{1}{1-2\eta}$ and make the running time of the algorithm polynomially dependent on $\frac{1}{1-2\eta}$.*

By sole use of `AN-BS` we can efficiently learn DT and weakly learn DNF as described in Theorems 16 and 19.

4.3 Coping with Attribute Noise of Unknown Rate

As was shown by Goldman and Sloan [GS95], coping with attribute noise becomes much more difficult when the noise rates are unknown.[10] The reason for this is that the usual way of handling the unknown noise rate—choosing the hypotheses with the minimum disagreement on samples—does not work for this type of noise. Nevertheless, several simple classes (e.g., $k-$DNF and monomials) are learnable when the attribute noise is uniform [GS95; DG95]. The following two theorems show that our learnability results hold in this setting as well.

Theorem 26 *DT is learnable by membership queries with uniform attribute noise of fixed unknown rate $\nu < 1/2$.*

10. In fact, no PAC learning algorithm exists that can tolerate a noise rate higher than 2ϵ, where ϵ is the required accuracy parameter.

Sketch of proof: Learning algorithm has access to MEM^p oracle for $p = [\nu]^n$. For the ease of presentation let us assume that all estimations we do by sampling produce the exact result. In particular, our `AN-BS` algorithm returns all the vectors with Fourier coefficients larger than the bound that was given to it(and only them) and we can estimate exactly coefficients of the function f^p defined in the proof of Theorem 23. For positive $\nu' < 1/2$ we look at the execution of `AN-BS` for learning DT supplied with noise rate ν' (equal for all the attributes) instead of the actual noise rate, i.e.,

$$\texttt{AN-BS}(\delta, \text{MEM}^p, n, \frac{\epsilon}{8s}, \log\frac{8s}{\epsilon}, [\nu']^n)$$

We then find all the Fourier coefficients for returned vectors and offset the effect of noise by multiplying every coefficient $\hat{f^p}(a)$ by $(1-2\nu')^{-w(a)}$ (as if ν' is a correct noise rate). We denote the function defined by coefficients we have found by $f^p_{\nu'}$ (with the assumptions we made this function is well-defined). If $\nu' = \nu$ then the algorithm we have just described will find all the required vectors and calculate their Fourier coefficients correctly. Thus, as follows from Lemma 15, $1-\epsilon < \|f^p_\nu\|^2 \leq 1$. It is easy to note that the larger the supplied noise rate ν' is, the more vectors the execution of `AN-BS` returns (since the threshold gets lower) and the larger is the value by which we multiply every Fourier coefficient of f^p that was found by `AN-BS`. That is, for every $a \in \{0,1\}^n$, $|\hat{f^p_{\nu'}}(a)|$ and $\|f^p_{\nu'}\|^2$ are monotonous in ν'. Thus by using a binary search we can efficiently find a value ν' for which $1-\epsilon \leq \|f^p_{\nu'}\|^2 \leq 1+\epsilon$ $(*)$ (the search will be efficient since values which are "close" to ν will satisfy this criterion). Denote the value we have found by ν_1. As we have noted, for every vector a, $|\hat{f^p_{\nu'}}(a)|$ is monotonous in ν' thus the criterion $(*)$ that ν_1 satisfies ensures that

$$\left|\|f^p_\nu\|^2 - \|f^p_{\nu'}\|^2\right| = \left|\sum_a \left(\hat{f^p_\nu}^2(a) - \hat{f^p_{\nu'}}^2(a)\right)\right| \leq 2\epsilon \tag{8}$$

On the other hand, as it was shown in the proof of Theorem 16, $sign(f^p_\nu)$ ϵ–approximates f and therefore

$$\sum_a \left(\hat{f^p_\nu}(a) - \hat{f}(a)\right)^2 \leq \epsilon \tag{9}$$

By combining equations 8 and 9 we can easily obtain that

$$\sum_a \left(\hat{f^p_{\nu_1}}(a) - \hat{f}(a)\right)^2 \leq 3\epsilon \tag{10}$$

That is, $sign(f^p_{\nu_1})$ 3ϵ–approximates f. □

Remark 27 *Without the "exactness" assumption we made in this proof we will need to improve the tolerance of all the queries so that the overall error in computation of $f^p_{\nu'}$ be smaller than $\epsilon/8$ (and substitute $\epsilon/4$ for ϵ in other bounds).*

Theorem 28 *DNF is weakly learnable by membership queries with uniform attribute noise of fixed unknown rate $\nu < 1/2$.*

Sketch of proof: We define $f^p_{\nu'}$ as in the previous proof. We denote by $\lambda(\nu')$ the absolute value of the maximal Fourier coefficient of $f^p_{\nu'}$ and by $\alpha(\nu')$ its corresponding vector. It is easy to see that $\lambda(\nu')$ is monotonous in ν' (by the same argument as in the previous proof). It is also given by the properties of DNF expressions that $\lambda(\nu) \geq \frac{1}{2s+1}$, where s is number of terms in the target function. Thus we can find the minimal ν' for which $\lambda(\nu') \geq \frac{1}{2s+1}$ (as usual, an estimate for this value is sufficient). We denote this error rate by ν_m. By the definition of ν_m, $\nu \geq \nu_m$ and therefore

$$\lambda(\nu) = |\hat{f}(\alpha(\nu_m))| \geq \lambda(\nu_m) \geq \frac{1}{2s+1} \ ,$$

i.e., $\chi_{\alpha(\nu_m)}$ (or its negation) $(\frac{1}{2} - \frac{1}{4s+2})$–approximates the target. □

Remark 29 *Both algorithms provided above can also be modified to tolerate the classification noise of the same (unknown) rate ν.*

5. Limitations of the SQ-$\mathcal{D}$ Model

So far we have been showing, so called, positive results about the models that we have introduced. The other side of understanding a model is proving that certain classes are not learnable in the model (or so called negative results). Non-trivial results of this kind are usually very hard to achieve. Fortunately for us, the statistical query model is the most tractable (non-trivial) learning model. In this chapter we extend this "tractability" to the SQ–$\mathcal{D}$ model. This will be done by showing a new and simple way to characterize weakly SQ–learnable concept classes. This characterization will then be extended to SQ–$\mathcal{D}$ and applied to classes of parity functions.

5.1 Properties of Statistical Queries

We start by showing a simple property of statistical queries which simplifies the analysis of the SQ model. According to the definition of statistical query, the query function ψ is a Boolean function of two parameters. The first one is denoted by x and the second one is denoted by $f(x)$ according to the values substituted when estimating the expectation of the query function. We distinguish two types of statistical queries (either regular or extended) based on their query function. We say that a query is *independent of the target* if it is based on ψ which is a function of x alone (and does not depend on the value of the second parameter – $f(x)$) and we say that a query is *correlational* if $\phi(x, f(x)) \equiv g(x)f(x)$ for a Boolean function $g(x)$.

Lemma 30 *Any statistical query (regular or extended) can be substituted by two statistical queries that are independent of the target and two correlational queries.*[11]

Proof: We denote the distribution of the query by D. The following equation proves the statement:

$$\mathbf{E}_D[\psi(x, f(x))] = \mathbf{E}_D\left[\psi(x, -1)\frac{1 - f(x)}{2} + \psi(x, 1)\frac{1 + f(x)}{2}\right] =$$

11. This decomposition appears implicitly in [BFJ+94].

$$\frac{1}{2}\mathbf{E}_D[\psi(x,1)f(x)] - \frac{1}{2}\mathbf{E}_D[\psi(x,-1)f(x)] + \frac{1}{2}\mathbf{E}_D[\psi(x,1)] + \frac{1}{2}\mathbf{E}_D[\psi(x,-1)] \ .$$

□

A query that is independent of the target is estimation of the expectation of random variable which is independent of f and therefore such a query can be substituted by a call to procedure `AMEAN` (we assume that the learning algorithm is able to sample randomly with respect to the distribution of the query). Thus we may assume that learning algorithms in the statistical query based models use only the correlational statistical queries.

5.2 The Characterization

We are now going to present the characterization of classes weakly learnable in statistical query based models. Let $\mathcal{F}$ be the concept class and $\mathcal{A}$ be an algorithm that weakly learns $\mathcal{F}$ in the SQ model (and uses only correlational queries). The characterization is based on the following observation: for every (but, probably, one) function in $\mathcal{F}$ there exists a correlational query for which $\mathcal{A}$ gets a reply which differs "noticeably" from 0. This is true because there could not exist two different functions in $\mathcal{F}$ for which the results of all the queries are "almost" equal. For every $f \in \mathcal{F}$ denote by g_f the function for which $|\mathbf{E}_D[fg_f]|$ differs "noticeably" from 0, that is, g_f (or its negation) weakly approximates f. This means that the set of queries of $\mathcal{A}$ weakly "approximates" all (but, probably, one) functions in $\mathcal{F}$. This argument is formalized in the following way. Let V and W be sets of Boolean functions. We say that W ϵ–approximates V with respect to distribution D if for every $f \in V$ there exists $g \in W$ such that g ϵ–approximates f with respect to distribution D. For a concept class $\mathcal{F}$ denote by $\mathcal{F}^s$ the set of all the functions in $\mathcal{F}$ of size s.

Theorem 31 *Let $\mathcal{F}$ be a concept class weakly SQ–learnable with respect to D, particularly, there exists an algorithm $\mathcal{A}$ that for every $f \in \mathcal{F}^s$, uses at most $p(n,s)$ queries of tolerance lower-bounded by $\frac{1}{r(n,s)}$ to produce $(\frac{1}{2} - \frac{1}{q(n,s)})$–approximation to f. Let $r'(n,s) \triangleq max\{2r(n,s), q(n,s)\}$. There exists a collection of sets $\{W_i\}_{i=1}^{\infty}$ such that $|W_s| \leq p(n,s)+1$ and W_s $(\frac{1}{2} - \frac{1}{r'(n,s)})$–approximates $\mathcal{F}^s$ with respect to D.*

Proof: According to the above discussion, we may assume that $\mathcal{A}$ uses only correlational queries. We build the set W_s as follows. We simulate algorithm $\mathcal{A}(n,s)$ and for every query (fg_i, ρ) we add g_i to W_s and return the value 0 as a result of the query. We continue the simulation till it violates any of the complexity bounds of $\mathcal{A}$ or $\mathcal{A}$ stops and returns final hypothesis h. In the later case we also add h to W_s. First, by the definition of W_s, $|W_s| \leq p(n,s)+1$. Now, assume that W_s does not $(\frac{1}{2} - \frac{1}{r'(n,s)})$–approximate $\mathcal{F}^s$, i.e., there exists $f \in \mathcal{F}^s$ such that for every $g \in W_s$, $|\mathbf{E}_D[fg]| < \frac{2}{r'(n,s)}$. This means that in our simulation for building W_s, zeros that we gave as answers are valid answers to the queries. Therefore, by the definition of $\mathcal{A}$, it returns hypothesis h that $(\frac{1}{2} - \frac{1}{q(n,s)})$–approximates f, that is, $h \in W_s$. This contradicts the assumption that $|\mathbf{E}_D[fh]| < \frac{2}{r'(n,s)} \leq \frac{2}{q(n,s)}$. □

In the above proof we assumed that $\mathcal{A}$ is a deterministic algorithm. On the other hand, the proof does not rely on the uniformity of the algorithm $\mathcal{A}$ (our only restriction is the polynomial number of queries). As is well known, any randomized algorithm can be transformed into non-uniform (deterministic) algorithm (in the context of languages this result is referred as BPP $\subset$ P/poly). Therefore, Theorem 31 is also true for classes weakly learnable by randomized algorithms.

This characterization can be easily extended to SQ–$\mathcal{D}$ in the following way. Every query (fg, ρ, D') to STAT$(f, \mathcal{D})$ has its own distribution $D' \in \mathcal{D}$. Thus the fact that results of a query differ "noticeably" from 0 means that f is weakly approximated with respect to some distribution from $\mathcal{D}$. Distinct queries may use distinct distributions from $\mathcal{D}$ and therefore along with every function in the approximating collection of sets we need to record the distribution with respect to which function in $\mathcal{F}$ is approximated. Formally, we say that a set $W \subseteq \{-1,1\}^X \times \mathcal{D}$ ϵ–approximates the set V of Boolean functions with respect to $\mathcal{D}$ if for every $f \in V$ there exists a pair $(g, D') \in W$ such that g ϵ–approximates f with respect to D'. Theorem 31 is generalized to SQ–$\mathcal{D}$ as follows.

Theorem 32 *Let $\mathcal{F}$ be a class learnable in SQ–$\mathcal{D}$. For p and r' defined as in Theorem 31 there exists a collection of sets $\{W_i\}_{i=1}^{\infty}$ such that $W_s \subseteq \{-1,1\}^X \times \mathcal{D}$, $|W_s| \leq p(n,s)+1$ and W_s $(\frac{1}{2} - \frac{1}{r'(n,s)})$–approximates $\mathcal{F}^s$ with respect to $\mathcal{D}$.*

5.3 Applications

Although the characterization we gave for the SQ model is not equivalent to characterization by SQ–DIM [BFJ+94] (see Section 1.1 for more details) it is applied in a very similar way. To demonstrate this we give a new proof for the main result obtained using the SQ–DIM characterization.

Theorem 33 *Any concept class containing superpolynomial number of parity functions is not weakly SQ–learnable with respect to the uniform distribution.*

Proof: Since the size of representation of any parity function is bounded by a fixed polynomial (e.g., $c_0 n$) by Theorem 31 (with Remark 1) we have that there exist polynomials $r(n)$ and $p(n)$ such that $\mathcal{F}$ has a $(\frac{1}{2} - \frac{1}{r(n)})$–approximating set W of size bounded by $p(n)$. For any $g \in W$ approximating a parity function χ_a means that $|\mathbf{E}[g(x)\chi_a]| = |\hat{g}(a)| \geq 2/r(n)$. But from Parseval's identity we know that $\sum_{a \in \{0,1\}^n} \hat{g}^2(a) = 1$. This means that at most $r^2(n)/4$ Fourier coefficients of g could be greater than $2/r(n)$, i.e., every $g \in W$ could be correlated with at most $r^2(n)/4$ different parity functions. On the other hand, $|W| \leq p(n)$, that is, W can approximate at most $p(n)r^2(n)/4$ parity functions. Thus, any $\mathcal{F}$ containing a superpolynomial number of parity functions is not weakly SQ–learnable with respect to the uniform distribution. $\square$

Our characterization of SQ–$\mathcal{D}$ learnable classes can be used to show the limitations of the SQ–$\mathcal{D}_\rho$ model. $\mathtt{BS}_\rho$ can be straightforwardly used to learn the class of all parity function containing at most $O(\log n)$ variables (with respect to U). On the other hand, we will show that the class of all the parity functions containing at most $c(n)\log n$ variables for unbounded $c(n)$ cannot be learned in SQ–$\mathcal{D}_\rho$. For this purpose we denote by $A^k = \{0,1\}^k[0]^{n-k}$, and let PAR^k denote the class of all parity functions for vectors in A^k, i.e., all parity functions based on the first k variables. In fact, we give somewhat stronger result.

Theorem 34 *PAR^k is not weakly learnable in SQ–$\mathcal{D}_\rho$ for $k(n) = \omega(\log n)$.*

Proof: Assume that the theorem is not true. By Theorem 32 (with Remark 1) there exist polynomials $p(n)$ and $r(n)$ and set of pairs $W = \{(g_1, D_1), (g_2, D_2), \ldots, (g_m, D_m)\}$ such that $|W| = m \leq p(n)$ and for every $\chi_a \in \mathsf{PAR}^k$ there exists $(g_i, D_i) \in W$, where $D_i \in \mathcal{D}_\rho$, such that $\mathbf{E}_{D_i}[g_i\chi_a] \geq 1/r(n)$. This means that $\sum_{1\leq i\leq m} \mathbf{E}^2_{D_i}[g_i\chi_a] \geq 1/r^2(n)$. Thus

$$\sum_{a\in A^k} \sum_{1\leq i\leq m} \mathbf{E}^2_{D_i}[g_i\chi_a] \geq |A^k|/r^2(n)$$

This means that there exists i such that

$$\sum_{a\in A^k} \mathbf{E}^2_{D_i}[g_i\chi_a] \geq |A^k|/(|W|r^2(n)) \geq |A^k|/(p(n)r^2(n)).$$

Or, if we denote by U^k a uniform distribution over A^k, then we get that

$$\mathcal{I} \triangleq \mathbf{E}_{a\sim U^k}\left[\mathbf{E}^2_{D_i}[g_i\chi_a]\right] \geq 1/(p(n)r^2(n)). \tag{11}$$

On the other hand,

$$\begin{aligned}\mathcal{I} = \mathbf{E}_{a\sim U^k}\left[\mathbf{E}^2_{D_i}[g_i\chi_a]\right] &= \mathbf{E}_{a\sim U^k}\left[\mathbf{E}_{x\sim D_i}\left[\mathbf{E}_{y\sim D_i}[g_i(x)\chi_a(x)g_i(y)\chi_a(y)]\right]\right] \\ &= \mathbf{E}_{x\sim D_i}[\mathbf{E}_{y\sim D_i}[g_i(x)g_i(y)\mathbf{E}_{a\sim U^k}[\chi_a(x\oplus y)]]] \ .\end{aligned} \tag{12}$$

But

$$\lambda(z) \triangleq \mathbf{E}_{a\sim U^k}[\chi_a(z)] = \mathbf{E}_{a\sim U^k}[\chi_{z_{[1,k]}}(a_{[1,k]})] = \begin{cases} 0 & z_{[1,k]} \neq [0]^k \\ 1 & z_{[1,k]} = [0]^k \end{cases}$$

Since $\lambda(x\oplus y)$ is non-negative and $|g_i(x)g_i(y)| = 1$, equation (12) yields that

$$\begin{aligned}\mathcal{I} \leq \mathbf{E}_{x\sim D_i}\left[\mathbf{E}_{y\sim D_i}[\lambda(x\oplus y)]\right] &= \mathbf{E}_{x\sim D_i}\left[\mathbf{Pr}_{y\sim D_i}[\lambda(x\oplus y) = 1]\right] \\ &= \mathbf{E}_{x\sim D_i}[\mathbf{Pr}_{y\sim D_i}[y_{[1,k]} = x_{[1,k]}]] \leq \mathbf{E}_{x\sim D_i}[(1-\rho)^k] \leq (1-\rho)^k \ .\end{aligned} \tag{13}$$

For $k = \omega(\log n)$, $(1-\rho)^k$ cannot be lower-bounded by inverse of a polynomial and thus equation (11) contradicts equation (13). $\square$

The last theorem may also be used to prove that the set of concept classes learnable in the PAC model with classification noise of constant rate is not contained in the set of concept classes weakly learnable in SQ–$\mathcal{D}_{\frac{1}{4}}$. This result can be obtained by combining our result with the following theorem for $k = \log n \log\log n$ [BKW00, Theorem 2].

Theorem 35 *PAR^k for noise rate η equal to any constant less than $\frac{1}{2}$, can be learned with respect to the uniform distribution using number of samples and total computation time $2^{O(k/\log k)}$.*

Another interesting application of our characterization is to demonstrate that PAR, the class of all parity functions, possesses some inherent hardness for any statistics based learning. For this purpose we give the following definition. We say that a distribution class $\mathcal{D}$ is *biased* if there exist a polynomial $p(n)$ such that $L_\infty(\mathcal{D}) \geq \frac{1}{p(n)}$. We may notice that if a distribution class $\mathcal{D}$ is biased then a non-uniform learning algorithm may get the value of the target function at the "biased" point. Use of attributes of specific point together with its label contradicts the idea of statistical learning and, on the other hand, is the main element of all the known algorithms for learning of parity functions.

Theorem 36 *Let $\mathcal{D}$ be a class of distributions. If $\mathcal{D}$ is not biased then for every $D \in \mathcal{D}$, PAR is not weakly learnable in the SQ–$\mathcal{D}$ model with respect to D.*

Proof: By assuming that PAR is weakly learnable in SQ–$\mathcal{D}$ with respect to some distribution $D \in \mathcal{D}$ and then proceeding as in the proof of the previous theorem we get that there exists set W and $(g_i, D_i) \in W$ ($D_i \in \mathcal{D}$) such that

$$\mathbf{E}_{a\sim U}\left[\mathbf{E}^2_{D_i}[g_i\chi_a]\right] \geq 1/(p(n)r^2(n)).$$

On the other hand,

$$\begin{aligned}\mathbf{E}_{a\sim U}\left[\mathbf{E}^2_{D_i}[g_i\chi_a]\right] &= \mathbf{E}_{a\sim U}\left[\mathbf{E}_{x\sim D_i}[\mathbf{E}_{y\sim D_i}[g_i(x)\chi_a(x)g_i(y)\chi_a(y)]]\right] \\ &= \mathbf{E}_{x\sim D_i}[\mathbf{E}_{y\sim D_i}[g_i(x)g_i(y)\mathbf{E}_{a\sim U}[\chi_a(x\oplus y)]]] \quad (14)\end{aligned}$$

By the properties of parity functions, if $x \neq y$ then $\mathbf{E}_{a\sim U}[\chi_a(x\oplus y)] = \mathbf{E}_{a\sim U}[\chi_{x\oplus y}(a)] = 0$. Otherwise, $\mathbf{E}_{a\sim U}[\chi_a(0^n)] = 1$. So the expression we are evaluating is equal to

$$\mathbf{E}_{x\sim D_i}[D_i(x)g_i^2(x)] = \mathbf{E}_{x\sim D_i}[D_i(x)]$$

Thus we got that $\mathbf{E}_{x\sim D_i}[D_i(x)] \geq 1/(p(n)r^2(n))$, in particular there exists x such that $D_i(x) \geq 1/(p(n)r^2(n))$, that is, $\mathcal{D}$ is biased. This contradiction finishes the proof. □

6. Conclusions and Open Problems

In this work we have shown that DNF is learnable in an extension of the PAC model (PAC-$\mathcal{D}_\rho$) that seems to be weaker than the use of membership queries. While it seems to be very difficult to say anything about the exact strength of the PAC-$\mathcal{D}_\rho$ model, we can show that the statistical query analogue of PAC-$\mathcal{D}_\rho$ (SQ–$\mathcal{D}_\rho$) is sufficient for our DNF learning algorithm. This fact combined with characterization of SQ–$\mathcal{D}_\rho$ learnable classes was used to show that, in terms of learning parity functions, SQ–$\mathcal{D}_\rho$ has just enough power to be able to learn parity functions that can be expressed as polynomial-size DNF expressions. Another useful property of the SQ–$\mathcal{D}_\rho$ model used is that it can be simulated using membership queries with persistent classification noise.

We have introduced an extension of attribute noise to membership queries. It turns out that the use of membership queries with product attribute noise is interestingly related to the SQ–$\mathcal{D}_\rho$ model. We have shown that decision trees are learnable and DNF expressions are weakly learnable by membership queries with product attribute noise.

The introduction of new model between PAC$_U$ and PAC$_U$+MQ poses interesting and apparently difficult questions about the relation between the strengths of these models. Another interesting question of similar nature is whether there exists a class learnable in SQ–$\mathcal{D}_\rho$ and not learnable by membership queries with product attribute noise, and particularly (in case the answer is positive) whether DNF is such a class. As we have proved, the power of SQ–D_ρ does not decrease when we decrease ρ. But it remains unclear to us whether the power is strictly increasing or the models are equally strong (for a constant $0 < \rho < 1/2$).

Acknowledgments

This research was supported by the fund for the promotion of research at the Technion. Part of research was done at the University of Calgary, Calgary, Alberta, Canada.

We would like to thank Dmitry Gavinsky, Jeff Jackson and Eyal Kushilevitz for useful discussions about this research, and to anonymous referees of COLT/EuroCOLT 2001 conference and Journal of Machine Learning Research for their careful reading, corrections, and insightful comments on this work.

References

[AD93] Javed Aslam, Scott Decatur. General Bounds on Statistical Query Learning and PAC Learning with Noise via Hypothesis Boosting. In *Proceedings of 34-th Annual Symposium on Foundations of Computer Science*, pp. 282–291, 1993.

[AL88] Dana Angluin, Philip Laird. Learning From Noisy Examples. In *Machine Learning*, 2(4) pp.343–370, 1988.

[AK91] Dana Angluin, Michael Kharitonov. When Won't Membership Queries Help? In *Proceedings of the 23-rd Annual ACM Symposium on Theory of Computing*, 1991, pp. 454–454.

[Bs93] Nader Bshouty. Exact Learning via the Monotone Theory. In *Proceedings of 34-th Annual Symposium on Foundations of Computer Science*, pp. 302–311, 1993.

[BFJ+94] Avrim Blum, Merrick Furst, Jeffrey Jackson, Michael Kearns, Yishay Mansour, Steven Rudich. Weakly Learning DNF and Characterizing Statistical Query Learning Using Fourier Analysis. In *Proceedings of the 26-th Annual ACM Symposium on the Theory of Computing*, pp. 253–262, 1994.

[BF01] Nader Bshouty, Vitaly Feldman. On Using Extended Statistical Queries to Avoid Membership Queries. In *Proceedings of the 14-th Annual Conference on COLT/EuroCOLT*, pp. 529–545, 2001.

[BG01] Nader Bshouty, Dmitry Gavinsky. On Boosting With Optimal Poly-Bounded Distributions. In *Proceedings of the 14-th Annual Conference on COLT/EuroCOLT*, pp. 490–506, 2001.

[BJT99] Nader Bshouty, Jeffrey Jackson, Christino Tamon. Uniform-Distribution Attribute Noise Learnability. In *Proceedings of the 12-th Annual Conference on COLT*, pp. 75–80, 1999.

[BKW00] Avrim Blum, Adam Kalai, Hal Wasserman. Noise-Tolerant Learning, the Parity Problem and the Statistical Query Model. In *Proceedings of the 32-th Annual ACM Symposium on Theory of Computing*, pp. 435–440, 2000.

[DG95] Scott Decatur, Rosario Gennaro. In *Proceedings of the Eighth Annual ACM Workshop on Computational Learning Theory*, pp.353–360, 1995.

[Fr90] Yoav Freund. Boosting a Weak Learning Algorithm by Majority. In *Proceedings of the Third Annual Workshop on COLT*, pp. 202–216, 1990.

[FS94] Yoav Freund, Robert Schapire. A Decision-theoretic Generalization of On-line Learning and an Application to Boosting. In *Proceedings of the Second Annual European Conference on Computational Learning Theory*, 1994.

[GKS94] Sally Goldman, Michael Kearns, Robert Shapire. Exact Identification of Circuits Using Fixed Points of Amplification Functions, In *SIAM Journal on Computing, 22 (1993)*, pp. 705–726.

[GL89] Oded Goldreich, Leonid A. Levin. A Hard-Core Predicate for all One-Way Functions. In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, pp. 25–32, 1989.

[GS95] Sally Goldman, Robert Sloan. Can PAC Learning Algorithms Tolerate Random Attribute Noise? In *Algorithmica*, 14(1) pp. 70–84, 1995.

[Ja94] Jeffrey Jackson. An Efficient Membership-Query Algorithm for Learning DNF with Respect to the Uniform Distribution. In *Proceedings of the 35th Annual Symposion on Foundations of Computer Science*, pp. 42–53, 1994.

[Ja97] Jeffrey Jackson. An Efficient Membership-Query Algorithm for Learning DNF with Respect to the Uniform Distribution. In *Journal of Computer and System Sciences*, 55(3), 1997

[JSS97] Jeffrey Jackson, Eli Shamir, Clara Shwartzman. Learning with Queries Corrupted by Classification Noise. In *Proceedings of Fifth Israel Symposium on Theory of Computing and Systems*, pp. 45–53, 1997.

[Kea93] Michael Kearns. Efficient Noise-Tolerant Learning from Statistical Queries. In *Proceedings of the Forth Annual Workshop on COLT,* pp. 392–401, 1993.

[KM91] Eyal Kushilevitz, Yishay Mansour. Learning Decision Trees Using the Fourier Spectrum. In *Proceedings of the 23-rd Annual Symposium on Theory of Computing,* pages 455–464.

[KS99] Adam Klivans, Rocco Servedio. Boosting and Hard-Core Sets. In *Proceedings of 40-th Annual Symposium on Foundations of Computer Science*, pp. 624–633, 1999.

[KSS92] Michael Kearns, Robert Shapire, Linda Sellie. Toward Efficient Agnostic Learning. In *Proceedings of the Fifth Annual Workshop on COLT,* pp. 341–352, 1992.

[Lai88] Phillip D. Laird. Learning From Good and Bad Data. *Kluwer Academic Publishers*, Boston, 1988.

[LMN89] Nathan Linial, Yishay Mansour, Noam Nisan. Constant Depth Circuits, Fourier Transform, and Learnability. *In Proceedings of the 31-st Symposium on the Foundations of Computer Science,* pp. 574–579, 1989.

[Man94] Yishay Mansour. Learning Boolean Functions via the Fourier Transform. In *Theoretical Advances in Neural Computation and Learning*, (V.P. Roychodhury and K-Y. Siu and A. Orlitsky, ed.), 391–424, 1994.

[ShSh95] Eli Shamir, Clara Shwartzman. Learning by Extended Statistical Queries and Its Relation to PAC Learning. In *Proceedings of Second European Conference, EuroCOLT '95*, pp. 357–366, 1995.

[Sch90] Robert Shapire. The Strength of Weak Learnability. In *Machine Learning*, 5, pp. 197–227, 1990.

[Sl88] Robert Sloan. Types of Noise in Data for Concept Learning. In *Proceedings of the 1988 Workshop on Computational Learning Theory*, pp. 91–96, MIT, ACM Press, 1988.

[SV88] George Shakelford, Dennis Volper. Learning $k-$DNF with Noise in the Attributes. In *Proceedings of the 1988 Workshop on COLT,* pp. 97–103, 1988.

[V84] Leslie Valiant. A Theory of the Learnable. In *Commmunications of the ACM*, 27(11) pp. 1134–1142, 1984.