# Learning random monotone DNF


Jeffrey C. Jackson *
Duquesne University
Pittsburgh, PA 15282
jacksonj@duq.edu

Homin K. Lee
Columbia University
New York, NY 10027
homin@cs.columbia.edu

Rocco A. Servedio†
Columbia University
New York, NY 10027
rocco@cs.columbia.edu

Andrew Wan
Columbia University
New York, NY 10027
atw12@cs.columbia.edu


May 22, 2008


**Abstract**

We give an algorithm that with high probability properly learns random monotone $t(n)$-term DNF under the uniform distribution on the Boolean cube $\{0,1\}^n$. For any polynomially bounded function $t(n) \leq \text{poly}(n)$ the algorithm runs in time $\text{poly}(n, 1/\epsilon)$ and with high probability outputs an $\epsilon$-accurate monotone DNF hypothesis. This is the first algorithm that can learn monotone DNF of arbitrary polynomial size in a reasonable average-case model of learning from random examples only.



*Supported in part by NSF award CCF-0209064

†Supported in part by NSF award CCF-0347282, by NSF award CCF-0523664, and by a Sloan Foundation Fellowship.

# 1 Introduction

**1.1 Motivation and background.** Any Boolean function $f : \{0,1\}^n \to \{0,1\}$ can be expressed as a disjunction of conjunctions of Boolean literals, i.e. as an OR of ANDs. Such a logical formula is said to be a *disjunctive normal formula*, or DNF. Learning polynomial-size DNF formulas (disjunctions of poly($n$) many conjunctions) from random examples is an outstanding open question in computational learning theory, dating back more than 20 years to Valiant's introduction of the PAC (Probably Approximately Correct) learning model [Val84].

The most intensively studied variant of the DNF learning problem is PAC learning DNF under the uniform distribution. In this problem the learner must generate a high-accuracy hypothesis with high probability when given uniform random examples labeled according to the unknown target DNF. (See Section 2 for a detailed definition of uniform distribution learning.) Despite much effort, no polynomial-time algorithms are known for this problem.

A tantalizing question that has been posed as a goal by many authors (see e.g. [Jac97, JT97, BBL98, Blu03b, Ser04]) is to learn *monotone* DNF, which only contain unnegated Boolean variables, under the uniform distribution. Besides being a natural restriction of the uniform distribution DNF learning problem, this problem is interesting because several impediments to learning general DNF under uniform – known lower bounds for Statistical Query based algorithms [BFJ+94], the apparent hardness of learning the subclass of $\log(n)$-juntas [Blu03a] – do not apply in the monotone case. This paper solves a natural average-case version of this problem.

**1.2 Previous work.** Many partial results have been obtained on learning monotone DNF under the uniform distribution. Verbeurgt [Ver90] gave an $n^{O(\log n)}$-time uniform distribution algorithm for learning any poly($n$)-term DNF, monotone or not. Several authors [KMSP94, SM00, BT96] have given results on learning monotone $t$-term DNF for larger and larger values of $t$; most recently, [Ser04] gave a uniform distribution algorithm that learns any $2^{O(\sqrt{\log n})}$-term monotone DNF to any constant accuracy $\epsilon = \Theta(1)$ in poly($n$) time. O'Donnell and Servedio [OS06] have recently shown that poly($n$)-leaf *decision trees* that compute monotone functions (a subclass of poly($n$)-term monotone DNF) can be learned to any constant accuracy under uniform in poly($n$) time. Various other problems related to learning different types of monotone functions under uniform have also been studied, see e.g. [KLV94, BBL98, Ver98, HM91, AM02].

Aizenstein and Pitt [AP95] first proposed a model of random DNF formulas and gave an exact learning algorithm that learns random DNFs generated in this way. As noted in [AP95] and [JS06], this model admits a trivial learning algorithm in the uniform PAC setting. Jackson and Servedio [JS05] gave a uniform distribution algorithm that learns log-depth decision trees on average in a natural random model. Previous work on average-case uniform PAC DNF learning, also by Jackson and Servedio, is described below.

**1.3 Our results.** The main result of this paper is a polynomial-time algorithm that can learn random poly($n$)-term monotone DNF with high probability. (We give a full description of the exact probability distribution defining our random DNFs in Section 5; briefly, the reader should think of our random $t$-term monotone DNFs as being obtained by independently drawing $t$ monotone conjunctions uniformly from the set of all conjunctions of length $\log_2 t$ over variables $x_1, \ldots, x_n$. Although many other distributions could be considered, this seems a natural starting point. Some justification for the choice of term length is given in Sections 5 and 7.)

**Theorem 1. [Informally]** *Let $t(n)$ be any function such that $t(n) \le poly(n)$, and let $c > 0$ be any fixed constant. Then random monotone $t(n)$-term DNFs are PAC learnable (with failure probability $\delta = n^{-c}$) to accuracy $\epsilon$ in $poly(n, 1/\epsilon)$ time under the uniform distribution. The algorithm outputs* *a monotone DNF as its hypothesis.*

**1.4 Our technique.** Jackson and Servedio [JS06] showed that for any $\gamma > 0$, a result similar to Theorem 1 holds for random $t$-term monotone DNF with $t \le n^{2-\gamma}$. The main open problem stated in [JS06] was to prove Theorem 1. Our work solves this problem by using the previous algorithm to handle $t \le n^{3/2}$, developing new Fourier lemmas for monotone DNF, and using these lemmas together with more general versions of techniques from [JS06] to handle $t \ge n^{3/2}$.

The crux of our strategy is to establish a connection between the term structure of certain monotone DNFs and their low-order Fourier coefficients. There is an extensive body of research on Fourier properties of monotone Boolean functions [BT96, MO03, BBL98], polynomial-size DNF [Jac97, Man94], and related classes such as constant-depth circuits and decision trees [LMN93, KM93, JKS02, OS06]. These results typically establish that *every* function in the class has a Fourier spectrum with certain properties; unfortunately, the Fourier properties that have been obtainable to date for general statements of this sort have not been sufficient to yield polynomial-time learning algorithms.

We take a different approach by carefully defining a set of conditions, and showing that *if a monotone DNF $f$ satisfies these conditions then the structure of the terms of $f$ will be reflected in the low-order Fourier coefficients of $f$.* In [JS06], the degree two Fourier coefficients were shown to reveal the structure of the terms for certain (including random) monotone DNFs having at most $n^{2-\gamma}$ terms. In this work we develop new lemmas about the Fourier coefficients of more general monotone DNF, and use these new lemmas to establish a connection between term structure and constant degree Fourier coefficients for monotone DNFs with any polynomial number of terms. Roughly speaking, this connection holds for monotone DNF that satisfy the following conditions:

- each term has a reasonably large fraction of assignments which satisfy it and no other term;
- for each small tuple of distinct terms, only a small fraction of assignments simultaneously satisfy all terms in the tuple; and
- for each small tuple of variables, only a small number of terms contains the entire tuple.

The "small" tuples referred to above should be thought of as tuples of constant size. The constant degree coefficients capture the structure of the terms in the following sense: tuples of variables that all co-occur in some term will have a large magnitude Fourier coefficient, and tuples of variables that do not all co-occur in some term will have a small magnitude Fourier coefficient (even if subsets of the tuple do co-occur in some terms). We show this in Section 3.

Next we show a reconstruction procedure for obtaining the monotone DNF from tuple-wise co-occurrence information. Given a hypergraph with a vertex for each variable, the procedure turns each co-occurrence into a hyperedge, and then searches for all hypercliques of size corresponding to the term length. The hypercliques that are found correspond to the terms of the monotone DNF hypothesis that the algorithm constructs. This procedure is described in Section 4; we show that it succeeds in constructing a high-accuracy hypothesis if the monotone DNF $f$ satisfies a few additional conditions. This is a significant generalization of a reconstruction procedure from [JS06] that was based on finding cliques in a graph (in the $n^{2-\gamma}$-term DNF setting, the algorithm deals only with co-occurrences of pairs of variables so it is sufficient to consider only ordinary graphs rather than hypergraphs).

The ingredients described so far thus give us an efficient algorithm to learn any monotone DNF that satisfies all of the required conditions. Finally, we show that random monotone DNF satisfy all the required conditions with high probability. We do this in Section 5 via a fairly delicate probabilistic argument. Section 6 combines the above ingredients to prove Theorem 1. We close the paper by showing that our technique lets us easily recapture the result of [HM91] that read-$k$ monotone DNF are learnable in polynomial time under the uniform distribution.

## 2 Preliminaries

We write $[n]$ to denote the set $\{1,\ldots,n\}$ and use capital letters for subsets of $[n]$. We will use calligraphic letters such as $\mathcal{C}$ to denote sets of sets and script letters such as $\mathscr{X}$ to denote sets of sets of sets. We write log to denote $\log_2$ and ln to denote the natural log. We write $U_n$ to denote the uniform distribution over the Boolean cube $\{0,1\}^n$.

A Boolean function $f:\{0,1\}^n \rightarrow \{0,1\}$ is *monotone* if changing the value of an input bit from 0 to 1 never causes the value of $f$ to change from 1 to 0. We denote the input variables to $f$ as $x_1,\ldots,x_n$. A *$t$-term monotone DNF* is a $t$-way OR of ANDs of Boolean variables (no negations). Recall that every monotone Boolean function has a unique representation as a reduced monotone DNF. We say that a term $T$ of such a monotone DNF is *uniquely satisfied* by input $x$ if $x$ satisfies $T$ and no other term of $f$.

Our learning model is an "average-case" variant of the well-studied uniform distribution PAC learning model. Let $D_C$ be a probability distribution over some fixed class $C$ of Boolean functions over $\{0,1\}^n$, and let $f$ (drawn from $D_C$) be an unknown target function. A learning algorithm $A$ for $D_C$ takes as input an accuracy parameter $0<\epsilon<1$ and a confidence parameter $0<\delta<1$. During its execution, algorithm $A$ has access to an *random example oracle* $EX(f,U_n)$, which, when queried generates a random labeled example $(x,f(x))$, where $x$ is drawn from $U_n$. The learning algorithm outputs a hypothesis $h$, which is a Boolean function over $\{0,1\}^n$. The error of this hypothesis is defined to be $\Pr_{U_n}[h(x)\neq f(x)]$. We say that *$A$ learns $D_C$ under $U_n$* if for every $0<\epsilon,\delta<1$, with probability at least $1-\delta$ (over both the random examples used for learning and the random draw of $f$ from $D_C$) algorithm $A$ outputs a hypothesis $h$ which has error at most $\epsilon$.

## 3 Fourier coefficients and the term structure of monotone DNF

Throughout Section 3 let $f(x_1,\ldots,x_n)$ be a monotone DNF and let $S\subseteq\{1,\ldots,n\}$ be a fixed subset of variables. We write $s$ to denote $|S|$ throughout this section. The Fourier coefficient, written $\hat{f}(S)$, measures the correlation between $f$ and the parity of the variables in $S$.

The main result of this section is Lemma 3, which shows that under suitable conditions on $f$, the value $|\hat{f}(S)|$ is "large" if and only if $f$ has a term containing all the variables of $S$. To prove this, we observe that the inputs which uniquely satisfy such a term will make a certain contribution to $\hat{f}(S)$. (In Section 3.1 we explain this in more detail and show how to view $\hat{f}(S)$ as a sum of contributions from inputs to $f$.) It remains then to show that the contribution from other inputs is small. The main technical novelty comes in Sections 3.2 and 3.3, where we show that all other inputs which make a contribution to $\hat{f}(S)$ must satisfy the terms of $f$ in a special way, and use this property to show that under suitable conditions on $f$, the fraction of such inputs must be small.

**3.1 Rewriting $\hat{f}(S)$.** We observe that $\hat{f}(S)$ can be expressed in terms of $2^s$ conditional probabilities, each of which is the probability that $f$ is satisfied conditioned on a particular setting of the variables in $S$. That is:

$$\begin{aligned}\hat{f}(S) &\stackrel{\text{def}}{=} \mathbf{E}_{x\in U^n}\left[(-1)^{\sum_{i\in S}x_i}\cdot f(x)\right] = \frac{1}{2^n}\sum_{x\in\{0,1\}^n}(-1)^{\sum_{i\in S}x_i}\cdot f(x)\\ &= \frac{1}{2^n}\sum_{U\subseteq S}(-1)^{|U|}\sum_{x\in Z_S(U)}f(x) = \frac{1}{2^s}\sum_{U\subseteq S}(-1)^{|U|}\Pr_x[f(x)=1 \mid x\in Z_S(U)],\end{aligned}$$

where $Z_S(U)$ denotes the set of those $x \in \{0,1\}^n$ such that $x_i = 1$ for all $i \in U$ and $x_i = 0$ for all $i \in S \setminus U$. If $f$ has some term $T$ containing all the variables in $S$, then $\Pr_x[f(x) = 1 \mid x \in Z_S(S)]$ is at least as large as $\Pr_x[T$ is uniquely satisfied in $f \mid x \in Z_S(S)]$. On the other hand, if $f$ has no such term, then $\Pr_x[f(x) = 1 \mid x \in Z_S(S)]$ does not receive this contribution. We will show that this contribution is the chief determinant of the magnitude of $\hat{f}(S)$.

It is helpful to rewrite $\hat{f}(S)$ as a sum of contributions from each input $x \in \{0,1\}^n$. To this end, we decompose $f$ according to the variables of $S$. Given a subset $U \subseteq S$, we will write $g_U$ to denote the disjunction of terms in $f$ that contain every variable indexed by $U \subseteq S$ and no variable indexed by $S \setminus U$, but with the variables indexed by $U$ removed from each term. (So for example if $f = x_1x_2x_4x_6 \vee x_1x_2x_5 \vee x_1x_2x_3 \vee x_3x_5 \vee x_1x_5x_6$ and $S = \{1,2,3\}$ and $U = \{1,2\}$, then $g_U = x_4x_6 \vee x_5$.) Thus we can split $f$ into disjoint sets of terms: $f = \bigvee_{U \subseteq S}(t_U \wedge g_U)$, where $t_U$ is the term consisting of exactly the variables indexed by $U$.

Suppose we are given $U \subseteq S$ and an $x$ that belongs to $Z_S(U)$. We have that $f(x) = 1$ if and only if $g_{U'}(x)$ is true for some $U' \subseteq U$. (Note that $t_{U'}(x)$ is true for every $U' \subseteq U$ since $x$ belongs to $Z_S(U)$.) Thus we can rewrite the Fourier coefficients $\hat{f}(S)$ as follows: (Below we write $I(P)$ to denote the indicator function that takes value 1 if predicate $P$ is true and value 0 if $P$ is false.)

$$\begin{aligned}
\hat{f}(S) &= \frac{1}{2^n}\sum_{U \subseteq S}(-1)^{|U|}\sum_{x \in Z_S(U)} f(x) = \sum_{U \subseteq S}(-1)^{|U|}\frac{1}{2^n}\sum_{x \in Z_S(U)} I\left(\bigvee_{U' \subseteq U} g_{U'}(x)\right) \\
&= \sum_{U \subseteq S}(-1)^{|U|}\frac{1}{2^s}\frac{1}{2^n}\sum_{x \in \{0,1\}^n} I\left(\bigvee_{U' \subseteq U} g_{U'}(x)\right) = \sum_{x \in \{0,1\}^n}\frac{1}{2^s}\frac{1}{2^n}\sum_{U \subseteq S}(-1)^{|U|} I\left(\bigvee_{U' \subseteq U} g_{U'}(x)\right).
\end{aligned}$$

We can rewrite this as

$$\hat{f}(S) = \sum_{x \in \{0,1\}^n} \mathrm{Con}_S(x), \quad \text{where} \quad \mathrm{Con}_S(x) \stackrel{\text{def}}{=} \frac{1}{2^s}\frac{1}{2^n}\sum_{U \subseteq S}(-1)^{|U|} I\left(\bigvee_{U' \subseteq U} g_{U'}(x)\right). \tag{1}$$

The value $\mathrm{Con}_S(x)$ may be viewed as the "contribution" that $x$ makes to $\hat{f}(S)$. Recall that when $f$ has a term $T$ which contains all the variables in $S$, those $x \in Z_S(S)$ which uniquely satisfy $T$ will contribute to $\hat{f}(S)$. We will show that under suitable conditions on $f$, the other $x$'s make little or no contribution.

**3.2 Bounding the contribution to $\hat{f}(S)$ from various inputs.** The variable $\mathcal{C}$ will denote a subset of $\mathcal{P}(S)$, the power set of $S$; i.e. $\mathcal{C}$ denotes a collection of subsets of $S$. We may view $\mathcal{C}$ as defining a set of $g_U$'s (those $g_U$'s for which $U$ belongs to $\mathcal{C}$).

We may partition the set of inputs $\{0,1\}^n$ into $2^{|\mathcal{P}(S)|} = 2^{2^s}$ parts according to what subset of the $2^s$ functions $\{g_U\}_{U \subseteq S}$ each $x \in \{0,1\}^n$ satisfies. For $\mathcal{C}$ a subset of $\mathcal{P}(S)$ we denote the corresponding piece of the partition by $P_{\mathcal{C}}$; so $P_{\mathcal{C}}$ consists of precisely those $x \in \{0,1\}^n$ that satisfy $\left(\bigwedge_{U \in \mathcal{C}} g_U\right) \wedge \left(\bigwedge_{U \notin \mathcal{C}} \overline{g}_U\right)$. Note that for any given fixed $\mathcal{C}$, each $x$ in $P_{\mathcal{C}}$ has exactly the same contribution $\mathrm{Con}_S(x)$ to the Fourier coefficient $\hat{f}(S)$ as every other $x'$ in $P_{\mathcal{C}}$; this is simply because $x$ and $x'$ will satisfy exactly the same set of $g_{U'}$'s in (1). More generally, we have the following (proved in Appendix A):

**Lemma 1.** *Let $\mathcal{C}$ be any subset of $\mathcal{P}(S)$. Suppose that there exist $U_1, U_2 \in \mathcal{C}$ such that $U_1 \subsetneq U_2$. Then for any $y, z$ where $y \in P_{\mathcal{C}}$ and $z \in P_{\mathcal{C} \setminus U_2}$, we have that:* $\mathrm{Con}_S(y) = \mathrm{Con}_S(z)$.

Given a collection $\mathcal{C}$ of subsets of $S$, we write $\mathrm{Con}_S(\mathcal{C})$ to denote $\sum_{x \in P_{\mathcal{C}}} \mathrm{Con}_S(x)$, and we refer to this quantity as the contribution that $\mathcal{C}$ makes to the Fourier coefficient $\hat{f}(S)$. It is clear that we have $\hat{f}(S) = \sum_{\mathcal{C} \subseteq \mathcal{P}(S)} \mathrm{Con}_S(\mathcal{C})$.

The following lemma, proved in Appendix B, establishes a broad class of $\mathcal{C}$'s for which $\mathrm{Con}_S(\mathcal{C})$ is zero:

**Lemma 2.** *Let $\mathcal{C}$ be any collection of subsets of $S$. If $\bigcup_{U \in \mathcal{C}} U \neq S$ then $\mathrm{Con}_S(x) = 0$ for each $x \in P_{\mathcal{C}}$ and hence $\mathrm{Con}_S(\mathcal{C}) = 0$.*

It remains to analyze those $\mathcal{C}$'s for which $\bigcup_{U \in \mathcal{C}} U = S$; for such a $\mathcal{C}$ we say that $\mathcal{C}$ *covers* $S$.

Recall from the previous discussion that $\mathrm{Con}_S(\mathcal{C}) = |P_{\mathcal{C}}| \cdot \mathrm{Con}_S(x)$ where $x$ is any element of $P_{\mathcal{C}}$. Since $|\mathrm{Con}_S(x)| \leq \frac{1}{2^n}$ for all $x \in \{0,1\}^n$, for any collection $\mathcal{C}$, we have that

$$|\mathrm{Con}_S(\mathcal{C})| \;\leq\; \Pr_{x \in U_n}[x \in P_{\mathcal{C}}] \;=\; \Pr_{x \in U_n}[(\bigwedge_{U \in \mathcal{C}} g_U) \wedge (\bigwedge_{U \notin \mathcal{C}} \overline{g}_s)] \;\leq\; \Pr_{x \in U_n}[(\bigwedge_{U \in \mathcal{C}} g_U)].$$

We are interested in bounding this probability for $\mathcal{C} \neq \{S\}$ (we will deal with the special case $\mathcal{C} = \{S\}$ separately later). Recall that each $g_U$ is a disjunction of terms; the expression $\bigwedge_{U \in \mathcal{C}} g_U$ is satisfied by precisely those $x$ that satisfy at least one term from each $g_U$ as $U$ ranges over all elements of $\mathcal{C}$. For $j \geq 1$ let us define a quantity $B_j$ as follows

$$B_j \stackrel{\text{def}}{=} \max_{i_1, \ldots, i_j} \Pr_{x \in U_n}[x \text{ simultaneously satisfies terms } T_{i_1}, \ldots, T_{i_j} \text{ in } \vee_{U \subseteq S}(g_U)]$$

where the max is taken over all $j$-tuples of distinct terms in $\vee_{U \subseteq S}(g_U)$. Then it is not hard to see that by a union bound, we have

$$|\mathrm{Con}_S(\mathcal{C})| \leq B_{|\mathcal{C}|} \prod_{U \in \mathcal{C}} (\# g_U), \tag{2}$$

where $\# g_U$ denotes the number of terms in the monotone DNF $g_U$.

The idea of why (2) is a useful bound is as follows. Intuitively, one would expect that the value of $B_j$ decreases as $j$ (the number of terms that must be satisfied) increases. One would also expect the value of $\# g_U$ to decrease as the size of $U$ increases (if $U$ contains more variables then fewer terms in $f$ will contain all of those variables). This means that there is a trade-off which helps us bound (2): if $|\mathcal{C}|$ is large then $B_{|\mathcal{C}|}$ is small, but if $|\mathcal{C}|$ is small then (since we know that $\bigcup_{U \in \mathcal{C}} U = S$) some $U$ is large and so $\prod_{U \in \mathcal{C}} \# g_U$ will be smaller.

**3.3 Bounding $\hat{f}(S)$ based on whether $S$ co-occurs in some term of $f$.** We are now ready to state formally the conditions on $\hat{f}$ that allow us to detect a co-occurrence of variables in the value of the corresponding Fourier coefficient.

**Lemma 3.** *Let $f : \{0,1\}^n \to \{-1,1\}$ be a monotone DNF. Fix a set $S \subset [n]$ of size $|S| = s$ and let*

$$\mathscr{Y} = \{\mathcal{C} \subseteq \mathcal{P}(S) : \mathcal{C} \text{ covers } S \text{ and } S \notin \mathcal{C}\}.$$

*Suppose that we define $\alpha, \beta_1, \ldots, \beta_{2^s}$ and $\Phi : \mathscr{Y} \to \mathbb{R}$ so that:*

**(C1)** *Each term in $f$ is uniquely satisfied with probability at least $\alpha$;*

**(C2)** *For $1 \leq j \leq 2^s$, each $j$-tuple of terms in $f$ is simultaneously satisfied with probability at most $\beta_j$; and*

**(C3)** *For every* $\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}$ *we have* $\prod_{U \in \mathcal{C}_{\mathscr{Y}}} (\#g_U) \leq \Phi(\mathcal{C}_{\mathscr{Y}})$.

*Then*

1. *If the variables in* $S$ *do not simultaneously co-occur in any term of* $f$*, then*

$$|\hat{f}(S)| \leq \Upsilon \quad \textit{where} \quad \Upsilon := \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} \left(2^s \beta_{|\mathcal{C}_{\mathscr{Y}}|} \Phi(\mathcal{C}_{\mathscr{Y}})\right);$$

2. *If the variables in* $S$ *do simultaneously co-occur in some term of* $f$*, then* $|\hat{f}(S)| \geq \frac{\alpha}{2^s} - 2 \cdot \Upsilon$.

Using Lemma 3, if $f$ satisfies conditions **(C1)** through **(C3)** with values of $\beta_j$ and $\Phi(\cdot)$ so that there is a "gap" between $\alpha/2^s$ and $3\Upsilon$, then we can determine whether all the variables in $S$ simultaneously co-occur in a term by estimating the magnitude of $\hat{f}(S)$.

*Proof.* Let $\mathcal{C}^\star$ denote the 'special' element of $P(S)$ that consists solely of the subset $S$, i.e. $\mathcal{C}^\star = \{S\}$, and let $\mathscr{X} = \{\mathcal{C} \subseteq \mathcal{P}(S) : \mathcal{C}$ covers $S$ and $S \in \mathcal{C}$ and $\mathcal{C} \neq \mathcal{C}^\star\}$. Using Lemma 2, we have

$$\hat{f}(S) = \mathrm{Con}_S(\mathcal{C}^\star) + \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} \mathrm{Con}_S(\mathcal{C}_{\mathscr{Y}}) + \sum_{\mathcal{C}_{\mathscr{X}} \in \mathscr{X}} \mathrm{Con}_S(\mathcal{C}_{\mathscr{X}}). \tag{3}$$

We first prove point 1. Suppose that the variables of $S$ do not simultaneously co-occur in any term of $f$. Then $g_S$ is the empty disjunction and $\#g_S = 0$, so $\mathrm{Con}_S(\mathcal{C}) = 0$ for any $\mathcal{C}$ containing $S$. Thus in this case we have $\hat{f}(S) = \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} \mathrm{Con}_S(\mathcal{C}_{\mathscr{Y}})$; using (2) and condition **(C3)**, it follows that $|\hat{f}(S)|$ is at most $\sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} B_{|\mathcal{C}_{\mathscr{Y}}|} \Phi(\mathcal{C}_{\mathscr{Y}})$. It is not hard to see that $B_{|\mathcal{C}_{\mathscr{Y}}|} \leq 2^s \beta_{|\mathcal{C}_{\mathscr{Y}}|}$ (we give a proof in Appendix C). So in this case we have

$$|\hat{f}(S)| \leq \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} |\mathrm{Con}_S(\mathcal{C}_{\mathscr{Y}})| \leq \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} B_{|\mathcal{C}_{\mathscr{Y}}|} \Phi(\mathcal{C}_{\mathscr{Y}}) \leq \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} \left(2^s \beta_{|\mathcal{C}_{\mathscr{Y}}|} \Phi(\mathcal{C}_{\mathscr{Y}})\right) = \Upsilon.$$

Now we turn to point 2. Suppose that the variables of $S$ do co-occur in some term of $f$. Let $x$ be any element of $P_{\mathcal{C}^\star}$, so $x$ satisfies $g_U$ if and only if $U = S$. It is easy to see from (1) that for such an $x$ we have $\mathrm{Con}_S(x) = (-1)^{|S|}/(2^n 2^s)$. We thus have that

$$\mathrm{Con}_S(\mathcal{C}^\star) = \frac{(-1)^{|S|}}{2^s} \cdot \Pr[x \in P_{\mathcal{C}^\star}] = \frac{(-1)^{|S|}}{2^s} \Pr[g_S \wedge (\bigwedge_{U \subsetneq S} \overline{g}_U)]. \tag{4}$$

Since $S$ co-occurs in some term of $f$, we have that $g_S$ contains at least one term $T$. By condition **(C1)**, the corresponding term $(T \wedge (\wedge_{i \in S} x_i))$ of $f$ is uniquely satisfied with probability at least $\alpha$. Since each assignment that uniquely satisfies $(T \wedge (\wedge_{i \in S} x_i))$ (among all the terms of $f$) must satisfy $g_S \wedge (\bigwedge_{U \subsetneq S} \overline{g}_U)$, we have that the magnitude of (4) is at least $\alpha/2^s$.

We now show that $|\sum_{\mathcal{C}_{\mathscr{X}} \in \mathscr{X}} \mathrm{Con}_S(\mathcal{C}_{\mathscr{X}})| \leq \Upsilon$, which completes the proof, since we already have that $|\sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} \mathrm{Con}_S(\mathcal{C}_{\mathscr{Y}})| \leq \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} |\mathrm{Con}_S(\mathcal{C}_{\mathscr{Y}})| \leq \Upsilon$. First note that if the set $\mathcal{C}_{\mathscr{X}} \setminus \{S\}$ does not cover $S$, then by Lemmas 1 and 2 we have that $\mathrm{Con}_S(x) = 0$ for each $x \in P_{\mathcal{C}_{\mathscr{X}}}$ and thus $\mathrm{Con}_S(\mathcal{C}_{\mathscr{X}}) = 0$. So we may restrict our attention to those $\mathcal{C}_{\mathscr{X}}$ such that $\mathcal{C}_{\mathscr{X}} \setminus \{S\}$ covers $S$. Now since such a $\mathcal{C}_{\mathscr{X}} \setminus \{S\}$ is simply some $\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}$, and each $\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}$ is obtained as $\mathcal{C}_{\mathscr{X}} \setminus \{S\}$ for at most one $\mathcal{C}_{\mathscr{X}} \in \mathscr{X}$, we have

$$\left| \sum_{\mathcal{C}_{\mathscr{X}} \in \mathscr{X}} \mathrm{Con}_S(\mathcal{C}_{\mathscr{X}}) \right| \leq \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} |\mathrm{Con}_S(\mathcal{C}_{\mathscr{Y}})| \leq \Upsilon.$$

■

## 4 Hypothesis formation

In this section, we show that if a target monotone DNF $f$ satisfies the conditions of Lemma 3 and two other simple conditions stated below (see Theorem 4), then it is possible to learn $f$ from uniform random examples.

**Theorem 4.** *Let $f$ be a $t$-term monotone DNF. Fix $s \in [n]$. Suppose that*

- *For all sets $S \subset [n], |S| = s$, conditions* **(C1)** *through* **(C3)** *of Lemma 3 hold for certain values $\alpha$, $\beta_j$, and $\Phi(\cdot)$ satisfying $\Delta > 0$, where $\Delta := \alpha/2^s - 3 \cdot \Upsilon$. (Recall that $\Upsilon := \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} \left(2^s \beta_{|\mathcal{C}_{\mathscr{Y}}|} \Phi(\mathcal{C}_{\mathscr{Y}})\right)$, where $\mathscr{Y} = \{\mathcal{C} \subseteq \mathcal{P}(S) : \mathcal{C}$ covers $S$ and $S \notin \mathcal{C}\}$.)*

**(C4)** *Every set $S$ of $s$ co-occurring variables in $f$ appears in at most $\gamma$ terms (here $\gamma \geq 2$); and*

**(C5)** *Every term of $f$ contains at most $\kappa$ variables (note that $s \leq \kappa \leq n$).*

*Then algorithm $\mathcal{A}$ (described formally in Appendix D) PAC learns $f$ to accuracy $\epsilon$ with confidence $1 - \delta$ given access to $EX(f, U_n)$, and runs in time $poly(n^{s+\gamma}, t, 1/\Delta, \gamma^\kappa, 1/\epsilon, \log(1/\delta))$.*

*Proof.* Lemma 3 implies that for each set $S \subset [n], |S| = s$,

- if the variables in $S$ all co-occur in some term of $f$, then $|\hat{f}(S)|$ is at least $\Delta/2$ larger than $\Upsilon + \Delta/2$;
- if the variables in $S$ do not all co-occur in some term of $f$, then $|\hat{f}(S)|$ is at least $\Delta/2$ smaller than $\Upsilon + \Delta/2$.

A straightforward application of Hoeffding bounds (to estimate the Fourier coefficients using a random sample of uniformly distributed examples) shows that Step 1 of Algorithm $\mathcal{A}$ can be executed in $\text{poly}(n^s, 1/\Delta, \log(1/\delta))$ time, and that with probability $1 - \delta/3$ the $S$'s that are marked as "good" will be precisely the $s$-tuples of variables that co-occur in some term of $f$.

Conceptually, the algorithm next constructs the hypergraph $G_f$ that has one vertex per variable in $f$ and that includes an $s$-vertex hyperedge if and only if the corresponding $s$ variables co-occur in some term of $f$. Clearly there is a $k$-hyperclique in $G_f$ for each term of $k$ variables in $f$. So if we could find all of the $k$-hypercliques in $G_f$ (where again $k$ ranges between $s$ and $\kappa$), then we could create a set $HC_f$ of monotone conjunctions of variables such that $f$ could be represented as an OR of $t$ of these conjunctions. Treating each of the conjunctions in $HC_f$ as a variable in the standard elimination algorithm for learning disjunctions (see e.g. Chapter 1 of [KV94]) would then enable us to properly PAC learn $f$ to accuracy $\epsilon$ with probability at least $1 - \delta/3$ in time polynomial in $n$, $t$, $|HC_f|$, $1/\epsilon$, and $\log(1/\delta)$. Thus, $\mathcal{A}$ will use a subalgorithm $\mathcal{A}'$ to find all of the $k$-hypercliques in $G_f$ and will then apply the elimination algorithm over the corresponding conjunctions to learn the final approximator $h$.

We now explain the subalgorithm $\mathcal{A}'$ for locating the set $HC_f$ of $k$-hypercliques. For each set $S$ of $s$ co-occurring variables, let $N_S \subseteq ([n] \setminus S)$ be defined as follows: a variable $x_i$ is in $N_S$ if and only if $x_i$ is present in some term that contains all of the variables in $S$. Since by assumption there are at most $\gamma$ terms containing such variables and each term contains at most $\kappa$ variables, this means that $|N_S| < \kappa\gamma$. The subalgorithm will use this bound as follows. For each set $S$ of $s$ co-occurring variables, $\mathcal{A}'$ will determine the set $N_S$ using a procedure $\mathcal{A}''$ described shortly. Then, for each $s \leq k \leq \kappa$ and each $(k - s)$-element subset $N'$ of $N_S$, $\mathcal{A}'$ will test whether or not $N' \cup S$ is a $k$-hyperclique in $G_f$. The set of all $k$-hypercliques found in this way is $HC_f$. For each $S$, the number of sets tested in this process is at most

$$\sum_{i=0}^{\kappa} \binom{|N_S|}{i} \leq \sum_{i=0}^{\kappa} \binom{\kappa\gamma}{i} \leq \left(\frac{e\kappa\gamma}{\kappa}\right)^\kappa = (e\gamma)^\kappa.$$

Thus, $|HC_f| = O(n^s(e\gamma)^\kappa)$, and this is an upper bound on the time required to execute Step 2 of subalgorithm $\mathcal{A}'$.

Finally, we need to define the procedure $\mathcal{A}''$ for finding $N_S$ for a given set $S$ of $s$ co-occurring variables. Fix such an $S$ and let $N_\gamma$ be a set of at most $\gamma$ variables in $([n] \setminus S)$ having the following properties:

**(P1)** In the projection $f_{N_\gamma \leftarrow 0}$ of $f$ in which all of the variables of $N_\gamma$ are fixed to 0, the variables in $S$ do not co-occur in any term; and

**(P2)** For every set $N'_\gamma \subset N_\gamma$ such that $|N'_\gamma| = |N_\gamma| - 1$, the variables in $S$ do co-occur in at least one term of $f_{N'_\gamma \leftarrow 0}$.

We will use the following claim (proved in Appendix E):

**Claim 5.** *$N_S$ is the union of all sets $N_\gamma$ of cardinality at most $\gamma$ that satisfy* **(P1)** *and* **(P2)**.

There are only $O(n^\gamma)$ possible candidate sets $N_\gamma$ to consider, so our problem now reduces to the following: given a set $N$ of at most $\gamma$ variables, determine whether the variables in $S$ co-occur in $f_{N \leftarrow 0}$.

Recall that since $f$ satisfies the three conditions **(C1)**, **(C2)** and **(C3)**, Lemma 3 implies that $|\hat{f}(S)|$ is either at most $\Upsilon$ (if the variables in $S$ do not co-occur in any term of $f$) or at least $\frac{\alpha}{2^s} - 2 \cdot \Upsilon$ (if the variables in $S$ do co-occur in some term). We now claim that the function $f_{N \leftarrow 0}$ has this property as well: i.e., $|\widehat{f_{N \leftarrow 0}}(S)|$ is either at most the same value $\Upsilon$ (if the variables in $S$ do not co-occur in any term of $f_{N \leftarrow 0}$) or at least the same value $\frac{\alpha}{2^s} - 2 \cdot \Upsilon$ (if the variables in $S$ do co-occur in some term of $f_{N \leftarrow 0}$). To see this, observe that the function $f_{N \leftarrow 0}$ is just $f$ with some terms removed. Since each term in $f$ is uniquely satisfied with probability at least $\alpha$ (this is condition **(C1)**), the same must be true of $f_{N \leftarrow 0}$ since removing terms from $f$ can only increase the probability of being uniquely satisfied for the remaining terms. Since each $j$-tuple of terms in $f$ is simultaneously satisfied with probability at most $\beta_j$ (this is condition **(C2)**), the same must be true for $j$-tuples of terms in $f_{N \leftarrow 0}$. Finally, for condition **(C3)**, the value of $\#g_U$ can only decrease in passing from $f$ to $f_{N \leftarrow 0}$. Thus, the upper bound of $\Upsilon$ that follows from applying Lemma 3 to $f$ is also a legitimate upper bound when the lemma is applied to $|\widehat{f_{N \leftarrow 0}}(S)|$, and similarly the lower bound of $\frac{\alpha}{2^s} - 2 \cdot \Upsilon$ is also a legitimate lower bound when the lemma is applied to $f_{N \leftarrow 0}$. Therefore, for every $|N| \leq \gamma$, a sufficiently accurate (within $\Delta/2$) estimate of $\widehat{f_{N \leftarrow 0}}(S)$ (as obtained in Step 1 of subalgorithm $\mathcal{A}''$) can be used to determine whether or not the variables in $S$ co-occur in any term of $f_{N \leftarrow 0}$.

To obtain the required estimate for $\widehat{f_{N \leftarrow 0}}$, observe that for a given set $N$, we can simulate a uniform example oracle for $f_{N \leftarrow 0}$ by filtering the examples from the uniform oracle for $f$ so that only examples setting the variables in $N$ to 0 are accepted. Since $|N| \leq \gamma$, the filter accepts with probability at least $1/2^\gamma$. A Hoeffding bound argument then shows that the Fourier coefficients $\widehat{f_{N \leftarrow 0}}(S)$ can be estimated (with probability of failure no more than a small fraction of $\delta$) from an example oracle for $f$ in time polynomial in $n$, $2^\gamma$, $1/\Delta$, and $\log(1/\delta)$.

Algorithm $\mathcal{A}''$, then, estimates Fourier coefficients of restricted versions of $f$, using a sample size sufficient to ensure that all of these coefficients are sufficiently accurate over all calls to $\mathcal{A}''$ with probability at least $1 - \delta/3$. These estimated coefficients are then used by $\mathcal{A}''$ to locate the set $N_S$ as just described. The overall algorithm $\mathcal{A}$ therefore succeeds with probability at least $1 - \delta$, and it is not hard to see that it runs in the time bound claimed. ■

**Required parameters.** In the above description of Algorithm $\mathcal{A}$, we assumed that it is given the values of $s, \alpha, \Upsilon, \gamma$, and $\kappa$. In fact it is not necessary to assume this; a standard argument gives a variant of the algorithm which succeeds without being given the values of these parameters.

The idea is simply to have the algorithm "guess" the values of each of these parameters, either exactly or to an adequate accuracy. The parameters $s, \gamma$ and $\kappa$ take positive integer values bounded by poly$(n)$. The other parameters $\alpha, \Upsilon$ take values between 0 and 1; a standard argument shows that if approximate values $\alpha'$ and $\Upsilon'$ (that differ from the true values by at most $1/\text{poly}(n)$) are used instead of the true values, the algorithm will still succeed. Thus there are at most poly$(n)$ total possible settings for $(s, \gamma, \kappa, \alpha, \Upsilon)$ that need to be tried. We can run Algorithm $\mathcal{A}$ for each of these candidate parameter settings, and test the resulting hypothesis; when we find the "right" parameter setting, we will obtain a high-accuracy hypothesis (and when this occurs, it is easy to recognize that it has occurred, simply by testing each hypothesis on a new sample of random labeled examples). This parameter guessing incurs an additional polynomial factor overhead. Thus Theorem 4 holds true for the extended version of Algorithm $\mathcal{A}$ that takes only $\epsilon, \delta$ as input parameters.

## 5 Random Monotone DNF

**5.1 The random monotone DNF model.** Let $\mathcal{M}_n^{t,k}$ be the probability distribution over monotone $t$-term DNF induced by the following process: each term is independently and uniformly chosen at random from all $\binom{n}{k}$ monotone ANDs of size exactly $k$ over $x_1, \ldots, x_n$.

Given a value of $t$, throughout this section we consider the $\mathcal{M}_n^{t,k}$ distribution where $k = \lfloor \log t \rfloor$ (we will relax this and consider a broader range of values for $k$ in Section 7). To motivate this choice, consider a random draw of $f$ from $\mathcal{M}_n^{t,k}$. If $k$ is too large relative to $t$ then a random $f \in \mathcal{M}_n^{t,k}$ will likely have $\Pr_{x \in U_n}[f(x) = 1] \approx 0$, and if $k$ is too small relative to $t$ then a random $f \in \mathcal{M}_n^{t,k}$ will likely have $\Pr_{x \in U_n}[f(x) = 1] \approx 1$; such functions are trivial to learn to high accuracy using either the constant-0 or constant-1 hypothesis. A straightforward analysis (see e.g. [JS06]) shows that for $k = \lfloor \log t \rfloor$ we have that $\mathbf{E}_{f \in \mathcal{M}_n^{t,k}}[\Pr_{x \in U_n}[f(x) = 1]]$ is bounded away from both 0 and 1, and thus we feel that this is an appealing and natural choice.

**5.2 Probabilistic analysis.** In this section we will establish various useful probabilistic lemmas regarding random monotone DNF of polynomially bounded size.

**Assumptions:** Throughout the rest of Section 5 we assume that $t(n)$ is any function such that $n^{3/2} \le t(n) \le \text{poly}(n)$. To handle the case when $t(n) \le n^{3/2}$, we will use the results from [JS06]. Let $a(n)$ be such that $t(n) = n^{a(n)}$. For brevity we write $t$ for $t(n)$ and $a$ for $a(n)$ below, but the reader should keep in mind that $a$ actually denotes a function $\frac{3}{2} \le a = a(n) \le O(1)$. Because of space limitations all proofs are given in Appendix F.

The first lemma provides a bound of the sort needed by condition **(C3)** of Lemma 3:

**Lemma 6.** *Let $|S| = s = \lfloor a \rfloor + 2$. Fix any $\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}$. Let $\delta_{\text{terms}} = n^{-\Omega(\log n)}$. With probability at least $1 - \delta_{\text{terms}}$ over the random draw of $f$ from $\mathcal{M}_n^{t,k}$, we have that for some absolute constant $c$ and all sufficiently large $n$,*

$$\prod_{U \in \mathcal{C}_{\mathscr{Y}}} (\#g_U) \le c \cdot \frac{t^{|\mathcal{C}_{\mathscr{Y}}|-1} k^{2^s}}{\sqrt{n}}. \tag{5}$$

The following lemma shows that for $f$ drawn from $\mathcal{M}_n^{t,k}$, with high probability each term is "uniquely satisfied" by a noticeable fraction of assignments as required by condition **(C1)**. (Note that since $k = O(\log n)$ and $t > n^{3/2}$, we have $\delta_{\text{usat}} = n^{-\Omega(\log \log n)}$ in the following.)

**Lemma 7.** *Let* $\delta_{\mathrm{usat}} := \exp(\frac{-tk}{3n}) + t^2(\frac{k}{n})^{\log\log t}$*. For* $n$ *sufficiently large, with probability at least* $1-\delta_{\mathrm{usat}}$ *over the random draw of* $f = T_1 \vee \cdots \vee T_t$ *from* $\mathcal{M}_n^{t,k}$*,* $f$ *is such that for all* $i = 1, \ldots, t$ *we have* $\Pr_x[T_i$ *is satisfied by* $x$ *but no other* $T_j$ *is satisfied by* $x\ ] \geq \frac{\Theta(1)}{2^k}$*.*

We now upper bound the probability that any $j$ distinct terms of a random DNF $f \in \mathcal{M}_n^{t,k}$ will be satisfied simultaneously (condition **(C2)**). (In the following lemma, note that for $j = \Theta(1)$, since $t = n^{\Theta(1)}$ and $k = \Theta(\log n)$ we have that the quantity $\delta_{\mathrm{simult}}$ is $n^{-\Theta(\log\log n)}$.)

**Lemma 8.** *Let* $1 \leq j \leq 2^s$*, and let* $\delta_{\mathrm{simult}} := \frac{t^j e^{jk-\log k}(jk-\log k)^{\log k}}{n^{\log k}}$*. With probability at least* $1-\delta_{\mathrm{simult}}$ *over the random draw of* $f = T_1 \vee \cdots \vee T_t$ *from* $\mathcal{M}_n^{t,k}$*, for all* $1 \leq \iota_1 < \cdots < \iota_j \leq t$ *we have* $\Pr[T_{\iota_1} \wedge \ldots \wedge T_{\iota_j}] \leq \beta_j$*, where* $\beta_j := \frac{k}{2^{jk}}$*.*

Finally, the following lemma shows that for all sufficiently large $n$, with high probability over the choice of $f$, every set $S$ of $s$ variables appears in at most $\gamma$ terms, where $\gamma$ is independent of $n$ (see condition **(C4)**).

**Lemma 9.** *Fix any constant* $c > 0$*. Let* $s = \lfloor a \rfloor + 2$ *and let* $\gamma = a + c + 1$*. Let* $\delta_\gamma = n^{-c}$*. Then for* $n$ *sufficiently large, with probability at least* $1 - \delta_\gamma$ *over the random draw of* $f$ *from* $\mathcal{M}_n^{t,k}$*, we have that every* $s$*-tuple of variables appears in at most* $\gamma$ *terms of* $f$*.*

## 6 Proof of Theorem 1

**Theorem 1. [Formally]** *Let* $t(n)$ *be any function such that* $t(n) \leq poly(n)$*, let* $a(n) = O(1)$ *be such that* $t(n) = n^{a(n)}$*, and let* $c > 0$ *be any fixed constant. Then for any* $n^{-c} < \delta < 1$ *and* $0 < \epsilon < 1$*,* $\mathcal{M}_n^{t(n),\lfloor \log t(n) \rfloor}$ *is PAC learnable under* $U_n$ *in poly(*$n^{2a(n)+c+3}$*,*$(a(n)+c+1)^{\log t(n)}$*,*$t(n)$*,*$1/\epsilon$*,* $\log 1/\delta$*) time.*

*Proof.* The result is proved for $t(n) \leq n^{3/2}$ already in [JS06], so we henceforth assume that $t(n) \geq n^{3/2}$. We use Theorem 4 and show that for $s = \lfloor a(n) \rfloor + 2$, random monotone $t(n)$-term DNFs, with probability at least $1-\delta$, satisfy conditions **(C1)**–**(C5)**with values $\alpha, \beta_j, \Phi(\cdot), \Delta, \gamma$, and $\kappa$ such that $\Delta > 0$ and the quantities $n^{s+\gamma}, 1/\Delta$, and $\gamma^\kappa$ are polynomial in $n$. This will show that the extended version of Algorithm $\mathcal{A}$ defined in Section 4 PAC learns random monotone $t(n)$-term DNFs in time poly$(n, 1/\epsilon)$. Let $t = t(n)$ and $k = \lfloor \log t \rfloor$, and let $f$ be drawn randomly from $\mathcal{M}_n^{t,k}$. By Lemmas 6–9, with probability at least $1 - \delta_{\mathrm{usat}} - \delta_\gamma - 2^{2^s}\delta_{\mathrm{terms}} - \delta_{\mathrm{simult}}$, $f$ will satisfy **(C1)**–**(C5)** with the following values:

$$\textbf{(C1)}\ \alpha > \tfrac{\Theta(1)}{2^k};\ \ \textbf{(C2)}\ \beta_j \leq \tfrac{k}{2^{jk}} \text{ for } 1 \leq j \leq 2^s;$$
$$\textbf{(C3)}\ \Phi(\mathcal{C}_{\mathscr{Y}}) \leq O(1)\tfrac{t^{|\mathcal{C}_{\mathscr{Y}}|-1}k^{2^s}}{\sqrt{n}} \text{ for all } \mathcal{C}_{\mathscr{Y}} \in \mathscr{Y};\ \textbf{(C4)}\ \gamma \leq a(n)+c+1;\ \ \textbf{(C5)}\ \kappa = k = \lfloor \log t \rfloor,$$

which gives us that $n^{s+\gamma} = n^{2a+c+3}$ and $\gamma^\kappa = (a+c+1)^{\lfloor \log t \rfloor}$. Finally, we show that $\Delta = \Omega(1/t)$ so $1/\Delta$ is polynomial in $n$:

$$\begin{aligned}\Delta &= \alpha/2^s - 3 \cdot \Upsilon = \frac{\Theta(1)}{t2^s} - 3 \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} 2^s \beta_{|\mathcal{C}_{\mathscr{Y}}|} \Phi(\mathcal{C}_{\mathscr{Y}}) \geq \frac{\Theta(1)}{t2^s} - \Theta(1) \sum_{\mathcal{C}_{\mathscr{Y}} \in \mathscr{Y}} 2^s \frac{k}{t^{|\mathcal{C}_{\mathscr{Y}}|}} \cdot \frac{t^{|\mathcal{C}_{\mathscr{Y}}|-1}k^{2^s}}{\sqrt{n}} \\ &= \frac{\Theta(1)}{t2^s} - \frac{\Theta(1)k^{2^s+1}}{t\sqrt{n}} = \Omega(1/t).\end{aligned}$$

■

## 7 Discussion

**Robustness of parameter settings.** Throughout Sections 5 and 6 we have assumed for simplicity that the term length $k$ in our random $t$-term monotone DNF is exactly $\lfloor \log t \rfloor$. In fact, the results extend to a broader range of $k$'s; one can straightforwardly verify that by very minor modifications of the given proofs, Theorem 1 holds for $\mathcal{M}_n^{t,k}$ for any $(\log t) - O(1) \le k \le O(\log t)$.

**Relation to previous results.** Our results are powerful enough to subsume some known "worst-case" results on learning restricted classes of monotone DNF formulas. Hancock and Mansour [HM91] have shown that read-$k$ monotone DNF (in which each Boolean variable $x_i$ occurs in at most $k$ terms) are learnable under the uniform distribution in poly($n$) time for constant $k$. Their result extends an earlier result of Kearns *et al.* [KLV94] showing that read-once DNF (which can be assumed monotone without loss of generality) are polynomial-time learnable under the uniform distribution. It is not hard to see that (a very restricted special case of) our algorithm can be used to learn read-$k$ monotone DNF in polynomial time; we give some details in Appendix G.

## References


[AM02] K. Amano and A. Maruoka. On learning monotone boolean functions under the uniform distribution. In *Proceedings of the 13th International Conference on Algorithmic Learning Theory (ALT)*, pages 57–68, 2002.

[AP95] Howard Aizenstein and Leonard Pitt. On the learnability of disjunctive normal form formulas. *Machine Learning*, 19:183, 1995.

[BBL98] A. Blum, C. Burch, and J. Langford. On learning monotone boolean functions. In *Proceedings of the Thirty-Ninth Annual Symposium on Foundations of Computer Science*, pages 408–415, 1998.

[BFJ+94] A. Blum, M. Furst, J. Jackson, M. Kearns, Y. Mansour, and S. Rudich. Weakly learning DNF and characterizing statistical query learning using Fourier analysis. In *Proceedings of the Twenty-Sixth Annual Symposium on Theory of Computing*, pages 253–262, 1994.

[Blu03a] A. Blum. Learning a function of $r$ relevant variables (open problem). In *Proc. 16th Annual COLT*, pages 731–733, 2003.

[Blu03b] A. Blum. Machine learning: a tour through some favorite results, directions, and open problems. FOCS 2003 tutorial slides, available at http://www-2.cs.cmu.edu/ avrim/Talks/FOCS03/tutorial.ppt, 2003.

[BT96] N. Bshouty and C. Tamon. On the Fourier spectrum of monotone functions. *Journal of the ACM*, 43(4):747–770, 1996.

[HM91] T. Hancock and Y. Mansour. Learning monotone $k$-$\mu$ DNF formulas on product distributions. In *Proceedings of the Fourth Annual Conference on Computational Learning Theory*, pages 179–193, 1991.

[Jac97] J. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *Journal of Computer and System Sciences*, 55:414–440, 1997.

[JKS02] Jeffrey C. Jackson, Adam R. Klivans, and Rocco A. Servedio. Learnability beyond $AC^0$. In *STOC '02: Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 776–784, New York, NY, USA, 2002. ACM Press.

[JS05] Jeffrey Jackson and Rocco Servedio. Learning random log-depth decision trees under the uniform distribution. *SIAM J. on Computing*, 34(5), 2005.

[JS06] J. Jackson and R. Servedio. On learning random DNF formulas under the uniform distribution. *Theory of Computing*, 2(8):147–172, 2006. (Preliminary version in RANDOM 2005).

[JT97] J. Jackson and C. Tamon. Fourier analysis in machine learning. ICML/COLT 1997 tutorial slides, available at http://learningtheory.org/resources.html, 1997.

[KLV94] M. Kearns, M. Li, and L. Valiant. Learning Boolean formulas. *Journal of the ACM*, 41(6):1298–1328, 1994.

[KM93] E. Kushilevitz and Y. Mansour. Learning decision trees using the Fourier spectrum. *SIAM J. on Computing*, 22(6):1331–1348, 1993.

[KMSP94] L. Kučera, A. Marchetti-Spaccamela, and M. Protassi. On learning monotone DNF formulae under uniform distributions. *Information and Computation*, 110:84–95, 1994.

[KV94] M. Kearns and U. Vazirani. *An introduction to computational learning theory.* MIT Press, Cambridge, MA, 1994.

[LMN93] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

[Man94] Y. Mansour. *Learning Boolean functions via the Fourier transform*, pages 391–424. Kluwer Academic Publishers, 1994.

[MO03] E. Mossel and R. O'Donnell. On the noise sensitivity of monotone functions. *Random Structures and Algorithms*, 23(3):333–350, 2003.

[MR95] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, New York, NY, 1995.

[OS06] R. O'Donnell and R. Servedio. Learning monotone decision trees in polynomial time. In *Proceedings of the 21st Conference on Computational Complexity (CCC)*, pages 213–225, 2006.

[Ser04] R. Servedio. On learning monotone DNF under product distributions. *Information and Computation*, 193(1):57–74, 2004. Preliminary version in *Proc. COLT'01.*

[SM00] Y. Sakai and A. Maruoka. Learning monotone log-term DNF formulas under the uniform distribution. *Theory of Computing Systems*, 33:17–33, 2000.

[Val84] L. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[Ver90] K. Verbeurgt. Learning DNF under the uniform distribution in quasi-polynomial time. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 314–326, 1990.

[Ver98] K. Verbeurgt. Learning sub-classes of monotone DNF on the uniform distribution. In *Proceedings of the Ninth Conference on Algorithmic Learning Theory*, pages 385–399, 1998.

## A Proof of Lemma 1

Consider (1). Fix any subset $U$ of $S$. We shall show that the indicator variable $I\left(\bigvee_{U'\subseteq U} g_{U'}(x)\right)$ takes the same value on $y$ and on $z$.

Recall that $y$ satisfies precisely those $g_r$'s such that $r \in \mathcal{C}$, and $z$ satisfies precisely those $g_r$'s such that $r \in (\mathcal{C} \setminus U_2)$. We have that:

1. $\bigvee_{U'\subseteq U} g_{U'}(y)$ is true if and only if there exists some $U' \subseteq U$, $U' \in \mathcal{C}$; and
2. $\bigvee_{U'\subseteq U} g_{U'}(z)$ is true if and only if there exists some $U'' \subseteq U$, $U'' \in (\mathcal{C} \setminus U_2)$.

Since $U_1 \subsetneq U_2$ and $U_1, U_2 \in \mathcal{C}$, there exists a $U'$ as described above if and only if there exists a $U''$ as described above. (Lemma 1) ■

## B Proof of Lemma 2

Before proving Lemma 2 we first introduce some notation and make a few easy observations. Let $\mathrm{odd}(U) \subset \mathcal{P}(S)$ be the set of all the odd-sized subsets of $S$ that are supersets of $U$, and let $\mathrm{even}(U)$ be defined similarly. For any $U \subsetneq S$ we have $|\mathrm{odd}(U)| = |\mathrm{even}(U)|$ since there are exactly $2^{|S|-|U|}$ subsets of $S$ containing $U$, half of which are even and half of which are odd. Note that if $U$ is the entire set $S$, then $S$ is the only superset of $U$, and of course $|S|$ cannot be both even and odd. Finally, note that given subsets $U_1, \ldots, U_k$ of $S$, we have:

$$\cap_{i=1}^{k} \mathrm{odd}(U_i) = \mathrm{odd}(\cup_{i=1}^{k} U_i). \tag{6}$$

(This just says that the intersection of the $\mathrm{odd}(U_i)$'s is equal to the set of all odd subsets of $S$ that contain the union of all the $U_i$'s.) A similar equality $\cap_{i=1}^{k} \mathrm{even}(U_i) = \mathrm{even}(\cup_{i=1}^{k} U_i)$ also holds.

Now we can give the proof:

**Proof of Lemma 2.** We know that each $x$ in $P_{\mathcal{C}}$ makes the same contribution to $\hat{f}(S)$. So fix any $x \in P_{\mathcal{C}}$; it suffices to show that the quantity $\sum_{U\subseteq S}(-1)^{|U|} I\left(\bigvee_{U'\subseteq U} g_{U'}(x)\right)$ is zero. This quantity will be zero if $x$ satisfies an equal number of $\bigvee_{U'\subseteq U} g_{U'}(x)$ for which $|U|$ is even, and for which $|U|$ is odd. The $U$ for which $x$ satisfies $\bigvee_{U'\subseteq U} g_{U'}(x)$ are the $U$ for which there exist some $U' \in \mathcal{C}$ such that $U' \subseteq U$. Thus, we need to count the number of even and odd-sized $U \subseteq S$ containing some $U' \in \mathcal{C}$, and show that $|\cup_{U'\in\mathcal{C}} \mathrm{odd}(U')| = |\cup_{U'\in\mathcal{C}} \mathrm{even}(U')|$ . Let $\mathcal{C} = \{U_1, \ldots, U_k\} \subseteq \mathcal{P}(\mathcal{S})$. By inclusion-exclusion,

$$\left|\cup_{U'\in\mathcal{C}} \mathrm{odd}(U')\right| = \sum_{i=1}^{k} |\mathrm{odd}(U_i)| - \sum_{i_1,i_2} |\mathrm{odd}(U_{i_1}) \cap \mathrm{odd}(U_{i_2})| \ldots + (-1)^{k-1} \left|\cap_{i=1}^{k} \mathrm{odd}(U_i)\right|, \tag{7}$$

and we have a similar expression for $|\cup_{U'\in\mathcal{C}} \mathrm{even}(U')|$ (identical to the RHS of (7) except with "even" everywhere in place of "odd").

By (6) we can rewrite each intersections of some $\mathrm{odd}(U_i)$'s as $\mathrm{odd}(\cup U_i)$, and similarly we can rewrite each intersection of some $\mathrm{even}(U_i)$'s as $\mathrm{even}(\cup U_i)$'s. Thus the RHS of (7) can be rewritten as a sum of $|\mathrm{odd}(\cup U_i)|$'s, and similarly $|\cup_{U'\in\mathcal{C}} \mathrm{even}(U')|$ can be rewritten as an identical sum of $|\mathrm{even}(\cup U_i)|$'s. Since by assumption each of these $\cup U_i$'s cannot be the whole set $S$, for each $\cup U_i$ we have $|\mathrm{odd}(\cup U_i)| = |\mathrm{even}(\cup U_i)|$. Therefore all the terms of $|\cup_{U'\in\mathcal{C}} \mathrm{odd}(U')|$ in (7) will match up with all the terms of $|\cup_{U'\in\mathcal{C}} \mathrm{even}(U')|$. It follows that $|\cup_{U'\in\mathcal{C}} \mathrm{odd}(U')|$ is indeed equal to $|\cup_{U'\in\mathcal{C}} \mathrm{even}(U')|$, and the lemma is proved. ■

## C Proof that $B_j \leq 2^s \beta_j$

We would like to bound the probability of simultaneously satisfying any fixed collection of $j$ terms in the DNF $f' = \vee_{U \subseteq S}(g_U)$. We have that for $1 \leq j \leq 2^s$, each $j$-tuple of terms in $f$ is simultaneously satisfied with probability at most $\beta_j$. Consider any fixed sequence $T'_{i_1}, \ldots, T'_{i_j}$ of terms from $f'$. Let $T_{i_1}, \ldots, T_{i_j}$ denote the sequence of terms in $f$ from which the terms $T'_{i_1}, \ldots, T'_{i_j}$ were derived, i.e. each term $T$ consists of $T' \wedge (\wedge_{i \in U} x_i)$ for some $U \subseteq S$. Since $T_{i_1} \wedge \cdots \wedge T_{i_j}$ is simply a monotone conjunction and $T'_{i_1} \wedge \cdots \wedge T'_{i_j}$ is simply the corresponding conjunction obtained by removing at most $|S| = s$ variables from $T_{i_1} \wedge \cdots \wedge T_{i_j}$, we have that $\Pr_x[T'_{i_1} \wedge \cdots \wedge T'_{i_j}] \leq 2^s \beta_j$.

## D The learning algorithm

**Algorithm** $\mathcal{A}$ (inputs are $\epsilon, \delta, s, \alpha, \Upsilon, \gamma, \kappa$, and access to $EX(f, U_n)$)

1. Define $\Delta := \alpha/2^s - 3 \cdot \Upsilon$.
2. For each $S \subset [n], |S| = s$, empirically estimate $\hat{f}(S)$ to within $\pm\Delta/3$ (with confidence $1 - \delta/3$ that all estimates have the required accuracy); let $\tilde{f}(S)$ be the empirical estimate thus obtained. Mark as "good" each $S$ for which $|\tilde{f}(S)| \geq \Upsilon + \frac{\Delta}{2}$.
3. Let $G_f$ denote the following $n$-vertex hypergraph: the vertices of $G_f$ correspond to variables $x_1, \ldots, x_n$, and $G_f$ contains each $s$-vertex hyperedge $S$ if and only if $S$ was marked as "good" in the previous step.
4. Run algorithm $\mathcal{A}'$ to identify the set $HC_f$ of all of the $k$-hypercliques in $G_f$, as $k$ ranges over $\{s, \ldots, \kappa\}$.
5. Run the standard elimination algorithm for disjunctions—with $\epsilon$ as the accuracy input parameter and $\delta/3$ as the confidence—over the "features" that are the monotone conjunctions corresponding to the hypercliques identified in the previous step. Output the resulting hypothesis $h$ (which is a monotone DNF).

**Algorithm** $\mathcal{A}'$ (input is the list of "good" sets $S$ identified in Step 1 of Algorithm $\mathcal{A}$)

1. For each good set $S$, run Algorithm $\mathcal{A}''$ to identify the set $N_S$ of all variables in $[n] \setminus S$ that occur in some term that also contains all variables in $S$.
2. For all $s \leq k \leq \kappa$, using brute-force search over all subsets $N'$ of at most $(k - s)$ many elements from $N_S$, check whether $N' \cup S$ is a $k$-hyperclique in $G_f$.

**Algorithm** $\mathcal{A}''$ (input is a good set $S$)

1. For each subset $N$ of at most $\gamma$ variables from $[n] \setminus S$, perform the following:
   (a) Empirically estimate $\widehat{f_{N \leftarrow 0}}(S)$ to additive accuracy $\pm\Delta/3$; let $\widetilde{f_{N \leftarrow 0}}(S)$ be the empirical estimate thus obtained. Mark each $N$ for which $\widetilde{f_{N \leftarrow 0}}(S) \geq \Upsilon + \frac{\Delta}{2}$.
2. Let $N_S$ be the union of all the $N$'s that were marked in the previous step. Return $N_S$.

## E Proof of Claim 5

We first show that the union of all sets satisfying **(P1)** and **(P2)** is a subset of $N_S$. To see this, note that if variable $x_i$ is not in $N_S$ (i.e. $x_i$ does not co-occur with $S$ in any term), then any set $N_\gamma$ that includes $x_i$ cannot satisfy both properties. This is because if $N_\gamma$ satisfies **(P1)** (i.e. $S$ does not co-occur in any term of $f_{N_\gamma \leftarrow 0}$), then the set $N'_\gamma = N_\gamma \setminus x_i$ will also be such that $S$ do not co-occur in any term of $f_{N'_\gamma \leftarrow 0}$, since $x$ does not co-occur with $S$ in any term.

Next, consider the minimal monotone DNF representation $D_f$ of the target $f$. Let $D_{f_S}$ be the monotone DNF expression obtained from $D_f$ by removing from $D_f$ all terms in which the variables of $S$ do not co-occur and then fixing all of the variables in $S$ to 1. Since $D_{f_S}$ has at most $\gamma$ terms, there is an equivalent minimal CNF $C_{f_S}$ in which each clause contains at most $\gamma$ variables. For each clause $C_i$ in $C_{f_S}$, the set of variables in $C_i$ satisfies both **(P1)** and **(P2)**: setting all of the variables in $C_i$ to 0 falsifies both $C_{f_S}$ and $D_{f_S}$ and therefore removes from $f$ all terms in which the variables of $S$ co-occur; but setting any proper subset of the variables in $C_i$ to 0 does not falsify $D_{f_S}$ and therefore leaves at least one term in $f$ in which the variables of $S$ co-occur. Furthermore, all of the variables in $D_{f_S}$ are also relevant in $C_{f_S}$, so every variable in $D_{f_S}$ appears in at least one clause of $C_{f_S}$. It follows that the union of the variables in the sets $N_\gamma$ satisfying **(P1)** and **(P2)** is a superset of the set of variables in $D_{f_S}$, that is, the set $N_S$. (Claim 5) ■

## F Proofs of Probabilistic Results for Random Monotone DNF

**Proof of Lemma 6.** We prove the lemma assuming that that $\emptyset \notin \mathcal{C}_{\mathscr{Y}}$. This is sufficient because if $\emptyset \in \mathcal{C}_{\mathscr{Y}}$, then $\mathcal{C}'_{\mathscr{Y}} = \mathcal{C}_{\mathscr{Y}} \setminus \emptyset$ is still contained in $\mathscr{Y}$, and applying the result to $\mathcal{C}'_{\mathscr{Y}}$ gives that $\prod_{U \in \mathcal{C}'_{\mathscr{Y}}} (\#g_U) \leq c \cdot \frac{t^{|\mathcal{C}'_{\mathscr{Y}}|-1} k^{2^s}}{\sqrt{n}}$ with probability at least $1 - \delta_{\text{terms}}$. Since $\#g_\emptyset \leq t$ and $|\mathcal{C}'_{\mathscr{Y}}| = |\mathcal{C}_{\mathscr{Y}}| - 1$, the conclusion of the lemma holds for $\mathcal{C}_{\mathscr{Y}}$ as well.

Fix any $\emptyset \neq U \in \mathcal{C}_{\mathscr{Y}}$ (note that since $U \in \mathcal{C}_{\mathscr{Y}}$ we also have $U \neq S$, and hence $|U| \leq s - 1 = \lfloor a \rfloor + 1$.). Recall that $f$ is chosen by picking each term $T_i$ to be a uniformly chosen set of $k$ distinct variables. The probability (over a random choice of $f$) that $T_1$ contains all the elements of $U$ and none of the elements of $S \setminus U$ is $\binom{n-s}{k-|U|} / \binom{n}{k}$; let us write $p_U$ to denote this quantity. Using the facts that $k = \Theta(\log n)$ and $1 \leq |U| < s = O(1)$, one can verify that

$$\frac{1}{2} (k/n)^{|U|} \leq p_U \leq (k/n)^{|U|}. \tag{8}$$

Since each of the $t$ terms of $f$ is chosen independently, we have that $\#g_U$ is binomially distributed according to $B(t, p_U)$, so $t \cdot \frac{1}{2} (\frac{k}{n})^{|U|} \leq \mathbf{E}[\#g_U] = tp_U \leq t(\frac{k}{n})^{|U|}$. Now recall that the Chernoff bound gives that $\Pr[X \geq (1+\zeta)\mathbf{E}[X]] \leq e^{-\zeta^2 tp/3}$ where $X$ is an independent and identically distributed random variable. For $X$ drawn from $B(t,p)$ and taking $\zeta = 1$, we get

$$\Pr\left[\#g_U > 2t\,(k/n)^{|U|}\right] \leq \Pr[\#g_U > 2tp_U] \leq \exp(-tp_U/3) \leq \exp(-t(k/n)^{|U|}/6). \tag{9}$$

Suppose first that $|U| \leq s - 2 = \lfloor a \rfloor$. If $|U| = 1$, then since $t \geq n^{3/2}$ we have that (9) is at most $\exp(-\sqrt{n} \log n)$. On the other hand, if $|U| > 1$ then $\lfloor a \rfloor \geq 2$, and since $t/n^{|U|} \geq t/n^{\lfloor a \rfloor} \geq 1$ we have that (9) $\leq \exp(-k^{|U|}/6) \leq n^{-\Omega(\log n)}$.

Now suppose that $|U| = s - 1$. In this case we use the following form of the Hoeffding bound (see e.g. Exercise 4.1 in [MR95]): if $\zeta > 2e - 1$, then $\Pr[X > (1+\zeta)\mathbf{E}[X]] \leq 2^{-(1+\zeta)\mathbf{E}[X]}$ for $X$ drawn from $B(t,p)$. Let $\zeta$ be such that $(1+\zeta)t(k/n)^{|U|} = t^{1/2a}$; note that this gives

$$1 + \zeta = t^{(1/(2a))-1}(n/k)^{|U|} = (\sqrt{n}/t)(n/k)^{|U|} \geq (\sqrt{n}/t)(n/k)^a = \sqrt{n}/\text{polylog}(n) \gg 2e,$$

so we may indeed apply the Hoeffding bound for this choice of $\zeta$. Using (8), we obtain

$$\Pr[\#g_U > t^{1/2a}] \le \Pr[\#g_U > (1+\zeta)tp_U] \le 2^{\frac{-t^{1/2a}}{2}} \le 2^{-\sqrt{n}/2}.$$

Taking a union bound over all possible sets $U \neq \emptyset$ (at most $2^s = O(1)$ many possibilities), we have that with probability at least $1 - \delta_{\text{terms}}$ over the draw of $f$, every such set $U \in \mathcal{C}_{\mathscr{Y}}$ satisfies

- if $|U| \le s-2$ then $\#g_U \le 2t(k/n)^{|U|}$; and
- if $|U| = s-1$ then $\#g_U \le t^{1/2a}$.

We henceforth assume the above conditions are satisfied, and now show that this gives the bound (5).

We partition $\mathcal{C}_{\mathscr{Y}}$ according to the size of $U$: let $\mathcal{C}^A_{\mathscr{Y}} = \{U \in \mathcal{C}_{\mathscr{Y}} : |U| = s-1\}$ and $\mathcal{C}^B_{\mathscr{Y}} = \mathcal{C}_{\mathscr{Y}} \setminus \mathcal{C}^A_{\mathscr{Y}} = \{U \in \mathcal{C}_{\mathscr{Y}} : |U| \le s-2\}$. Then

$$\prod_{U \in \mathcal{C}_{\mathscr{Y}}} (\#g_U) = \prod_{U \in \mathcal{C}^A_{\mathscr{Y}}} (\#g_U) \prod_{U \in \mathcal{C}^B_{\mathscr{Y}}} (\#g_U) \le t^{|\mathcal{C}^A_{\mathscr{Y}}|/2a} \cdot (2t)^{|\mathcal{C}^B_{\mathscr{Y}}|} (k/n)^{\sum_{U \in \mathcal{C}^B_{\mathscr{Y}}} |U|}.$$

By definition of $\mathcal{C}_{\mathscr{Y}}$ we have that $\sum_{U \in \mathcal{C}_{\mathscr{Y}}} |U| \ge s$. Now if $|\mathcal{C}^A_{\mathscr{Y}}| = 0$ , then we have

$$\prod_{U \in \mathcal{C}_{\mathscr{Y}}} (\#g_U) \le (2t)^{|\mathcal{C}_{\mathscr{Y}}|} (k/n)^{\sum_{U \in \mathcal{C}_{\mathscr{Y}}} |U|} \le (2t)^{|\mathcal{C}_{\mathscr{Y}}|} (k/n)^s = 2^{|\mathcal{C}_{\mathscr{Y}}|} \frac{t^{|\mathcal{C}_{\mathscr{Y}}|-1} k^s}{n^{s-a}} \le 2^{|\mathcal{C}_{\mathscr{Y}}|} \frac{t^{|\mathcal{C}_{\mathscr{Y}}|-1} k^s}{n}.$$

On the other hand, if $|\mathcal{C}^A_{\mathscr{Y}}| > 0$, then

$$\prod_{U \in \mathcal{C}_{\mathscr{Y}}} (\#g_U) \le 2^{|\mathcal{C}^B_{\mathscr{Y}}|} t^{|\mathcal{C}^A_{\mathscr{Y}}|/2a + |\mathcal{C}^B_{\mathscr{Y}}|} (k/n)^{|\mathcal{C}^B_{\mathscr{Y}}|}.$$

Since $|\mathcal{C}_{\mathscr{Y}}| - (\frac{2a-1}{2a})|\mathcal{C}^A_{\mathscr{Y}}| = |\mathcal{C}^A_{\mathscr{Y}}|/2a + |\mathcal{C}^B_{\mathscr{Y}}|$, observing that $|\mathcal{C}^B_{\mathscr{Y}}| \le 2^s$ it suffices to show that

$$2^{|\mathcal{C}^B_{\mathscr{Y}}|} k^{2^s} \frac{t^{|\mathcal{C}_{\mathscr{Y}}|}}{n^{\frac{2a-1}{2}|\mathcal{C}^A_{y}| + |\mathcal{C}^B_{\mathscr{Y}}|}} \le c \cdot \frac{t^{|\mathcal{C}_{\mathscr{Y}}|-1} k^{2^s}}{\sqrt{n}},$$

which holds when $\frac{2a-1}{2}|\mathcal{C}^A_{\mathscr{Y}}| + |\mathcal{C}^B_{\mathscr{Y}}| \ge a + 1/2$. This inequality follows from the fact that $|\mathcal{C}_{\mathscr{Y}}| \ge 2$ (since $S \notin \mathcal{C}_{\mathscr{Y}}$ and $\cup_{U \in \mathcal{C}_{\mathscr{Y}}} U = S$), $|\mathcal{C}^A_{\mathscr{Y}}| \ge 1$, and $a \ge \frac{3}{2}$. ■

**Proof of Lemma 7.** For a monotone $t$-term DNF $f = T_1 \vee \cdots \vee T_t$, let $f^i$ denote the projected function obtained from $f$ by removing the term $T_i$ from $f$ and restricting all of the variables which were present in term $T_i$ to 1. For $\ell \neq i$ we write $T^i_\ell$ to denote the term obtained by setting all variables in $T_i$ to 1 in $T_\ell$, i.e. $T^i_\ell$ is the term in $f^i$ corresponding to $T_\ell$. Now the probability that $T_i$ is satisfied and no other $T_j$ is satisfied is given by $\Pr[T_i] \cdot \Pr[\overline{T^i_\ell}$ for all $\ell \neq i \; |T_i] = \Pr[T_i] \cdot \Pr[\overline{f^i}]$. Since $\Pr[T_i] = \frac{1}{2^k}$, it suffices to bound $\Pr[\overline{f^i}]$ from below. As in [JS06], we show that the following four facts all hold with probability $1 - \delta_{usat}$:

1. $\Pr[\overline{f^i}] \ge \prod_{\ell : \ell \neq i} \Pr[\overline{T^i_\ell}]$.
2. $\prod_{\ell : T^i_\ell \equiv T_\ell} \Pr[\overline{T^i_\ell}] > 1/16$.
3. $|\{T^i_\ell : \ell \neq i \wedge T^i_\ell \not\equiv T_\ell\}| \le \frac{2tk^2}{n}$.

4. No term in $f^i$ has fewer than $k - \log\log t$ variables.

Together, these conditions imply that

$$\Pr[\overline{f^i}] \geq \prod_{\ell : T_\ell \equiv T_\ell^i} \Pr[\overline{T_\ell^i}] \prod_{\ell : T_\ell \not\equiv T_\ell^i, \ell \neq i} \Pr[\overline{T_\ell^i}] \geq \frac{1}{16}\left(1 - \frac{\log t}{2^k}\right)^{2tk^2/n} \geq \frac{1}{32}.$$

We now prove (1)–(4). To prove (1) note that

$$\Pr[\overline{f}] = \Pr[\overline{T_1} \wedge \overline{T_2} \wedge \cdots \wedge \overline{T_t}] = \Pr[\overline{T_1} | \overline{T_2} \wedge \cdots \wedge \overline{T_t}] \Pr[\overline{T_2} | \overline{T_3} \wedge \cdots \wedge \overline{T_t}] \cdots \Pr[\overline{T_{t-1}} | \overline{T_t}] \Pr[\overline{T_t}]$$

which is at least $\prod_{i=1}^t \Pr[\overline{T_i}]$ since $f$ is monotone. (Conditioning on terms being unsatisfied can only increase the number of variables set to 0 and thus can only increase the chances a particular term is unsatisfied).

For any $i$ and $\ell$ such that $T_\ell^i \equiv T_\ell$, we have $\Pr[\overline{T_\ell^i}] = \Pr[\overline{T_\ell}] = 1 - \Pr[T_\ell] = 1 - \frac{1}{2^k}$. Certainly there are at most $t$ such $T_\ell^i$, so (2) follows from the fact that $k = \lfloor \log t \rfloor$ so $(1 - \frac{1}{2^k})^t > 1/16$.

For (3), first we prove that with probability at least $1 - \exp(\frac{-tk}{3n})$, any variable appears in at most $\frac{2tk}{n}$ many terms. Each variable $v_j$ appears in each fixed term $T_\ell$ with probability $k/n$. Since the terms are chosen independently, the number of occurrences of $v_j$ is binomially distributed according to $B(t,p)$ with $p = k/n$. Now recall that the Chernoff bound gives that $\Pr[X \geq (1+\zeta)\mathbf{E}[X]] \leq e^{-\zeta^2 tp/3}$ where $X$ is drawn from $B(t,p)$. Taking $\zeta = 1$, we get that $\Pr[X > \frac{2tk}{n}] < \exp(\frac{-tk}{3n})$. If $T_\ell \not\equiv T_\ell^i$ then $T_\ell$ must contain some variable from $T_i$. Assuming every variable appears in at most $2tk/n$ terms, and term $T_i$ has at most $k$ variables, there can be at most $k \cdot 2tk/n$ such terms.

Finally, Lemma 3.5 of [JS06] gives that (4) holds with probability at least $1 - t^2(\frac{k^2}{n})^{\log\log t}$. Thus we have that conditions (1)–(4) all hold with probability at least $1 - \delta_{usat}$. ∎

**Proof of Lemma 8.** Fix any sequence $\iota_1 < \cdots < \iota_j$ of $j$ terms. Let $v \leq jk$ be the number of distinct variables that occur in these terms. First, we will bound the probability that $v > w := jk - \log k$. Consider any particular fixed set of $w$ variables. The probability that none of the $j$ terms includes any variable outside of the $w$ variables is precisely $(\binom{w}{k}/\binom{n}{k})^j$. Thus, the probability that $v \leq w$ is by the union bound:

$$\Pr[v \leq w] \leq \binom{n}{w}\left(\frac{\binom{w}{k}}{\binom{n}{k}}\right)^j \leq \left(\frac{en}{w}\right)^w \left(\frac{w}{n}\right)^{jk} \leq \frac{e^{jk - \log k}(jk - \log k)^{\log k}}{n^{\log k}}.$$

Taking a union bound over all (at most $t^j$) sequences $1 \leq \iota_1 < \cdots < \iota_j \leq t$, we have that with probability $1 - \delta_{\text{simult}}$, every sequence of $j$ terms contains at least $w$ distinct variables, and thus for every sequence we have $\Pr[T_{\iota_1} \wedge \cdots \wedge T_{\iota_j}] \leq 2^{-w} = k/2^{jk}$. ∎

**Proof of Lemma 9.** For any fixed $r \in \{1, \ldots, t\}$ and any fixed $S$ such that $|S| = s$, we have Pr[all variables in $S$ occur in $T_r$] $= \frac{k(k-1)\cdots(k-s+1)}{n(n-1)\cdots(n-s+1)} \leq \left(\frac{k}{n}\right)^s$. Since terms are chosen independently, the probability that the variables in $S$ co-occur in a fixed collection of $\gamma+1$ terms is at most $\left(\frac{k}{n}\right)^{s(\gamma+1)}$. By the union bound, the probability that these variables co-occur in any collection of $\gamma+1$ terms is at most $\binom{t}{\gamma+1} \cdot (\frac{k}{n})^{s(\gamma+1)} \leq \left(\frac{tk^s}{n^s}\right)^{\gamma+1}$. Using the union bound again, we have that the probability that any $s$-tuple of variables co-occurs in more than $\gamma$ terms is at most $\binom{n}{s} \cdot \left(\frac{tk^s}{n^s}\right)^{\gamma+1}$. Recalling that $t = n^a$, that $s = \lfloor a \rfloor + 2$, and that $k = \lfloor \log t \rfloor = O(\log n)$, we have that this probability is at most $\text{polylog}(n) \cdot n^{a(\gamma+1)-(a+1)\gamma} = \text{polylog}(n) \cdot n^{a-\gamma}$. By our choice of $\gamma$ this is at most $\delta_\gamma$, and the proof is done. ∎

## G Learning read-$k$ monotone DNF with our algorithm

Note first that we may assume the unknown target read-$k$ DNF $f$ has $\frac{\epsilon}{2} \le \Pr[f(x) = 1] \le 1 - \frac{\epsilon}{2}$, since otherwise it is trivial to learn to accuracy $\epsilon$.

We show that we can apply Theorem 4 to learn $f$. Any read-$k$ DNF has at most $kn$ total occurrences of variables, so we certainly have that $f$ is a $t(n)$-term DNF with $t(n) = O(n)$. We will take $s = 1$. Since $f$ is a read-$k$ DNF, we may take $\gamma = 2$ in condition **(C4)**. By the usual reasoning, we may suppose without loss of generality that each term of $f$ contains at most $O(\log \frac{n}{\epsilon})$ many variables (this is because the probability that any longer term is ever satisfied by any example in a poly$(n/\epsilon)$-size set of random examples is negligibly small). Thus we may take $\kappa = O(\log \frac{n}{\epsilon})$ in condition **(C5)**.

Turning to Lemma 3, since $s = 1$ we have that the collection $\mathscr{Y}$ is in fact empty – for $S = \{x_i\}$, the only $\mathcal{C} \subseteq \mathcal{P}(S)$ that cover $S$ are $\mathcal{C} = \{\emptyset, \{x_i\}\}$ and $\mathcal{C} = \{\{x_i\}\}$, both of which clearly contain $S$. We thus have $\Upsilon = 0$, so $\Delta = \frac{\alpha}{2}$ and it remains only to prove that $\alpha$ is "not too small," i.e. that each term in $f$ is uniquely satisfied with probability at least $\Omega(1/\text{poly}(n/\epsilon))$. An easy argument in [HM91] gives precisely the desired result; they show that for any monotone read-$k$ DNF $f$ that has $\Pr[f(x) = 0] = p$, every term $T$ that contains $\mathcal{C}$ variables satisfies Pr[$T$ is true and all other terms are false] $\ge p2^{-k|\mathcal{C}|}$. Since we have $p \ge \frac{\epsilon}{2}$ and $\mathcal{C} \le \kappa = O(\log \frac{n}{\epsilon})$, we obtain $\alpha \ge \Omega(1/\text{poly}(n/\epsilon))$ as required. So we may apply Theorem 4 to conclude that our algorithm learns $f$ in time poly$(n, 1/\epsilon, \log(1/\delta))$.