Exact learning of DNF formulas using DNF hypotheses

[Extended Abstract]

Lisa Hellerstein*
Department of Computer and Information Science
Polytechnic University, Brooklyn NY 11201
hstein@duke.poly.edu

Vijay Raghavan †
Electrical Engineering and Computer Science Department
Vanderbilt University, Nashville TN 37235
raghavan@vuse.vanderbilt.edu

ABSTRACT

We show the following:

(a) For any $\epsilon > 0$, $(\log n)^{(3+\epsilon)}$-term DNF cannot be polynomial-query learned with membership and strongly proper equivalence queries.

(b) For any function $f(n) \in o(\frac{\sqrt{n}}{\log n})$, m-term DNF formulas cannot be polynomial-query learned by a membership and equivalence query algorithm that uses $m \cdot f(n)$-term DNF formulas as hypotheses.

(c) Read-thrice DNF formulas are not learnable with membership and proper equivalence queries.

(d) $\log n$-term DNF formulas can be polynomial-query learned with membership and proper equivalence queries. (This complements a result of Bshouty, Goldman, Hancock, and Matar stating that $\sqrt{\log n}$-term DNF can be so learned in polynomial *time*.)

Using purely information theoretic techniques, these results extend and improve what is currently known. (For example, a weaker version of (a) was known only under a barely plausible complexity theoretic assumption, (b) was previously unknown, and (c) was known under the assumption P $\neq$ NP.)

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*concept learning*; F.2.m [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

*Research partially funded by NSF Grants CCR-9877122 and EIA-9806207
†Research partially funded by NSF Grant CCR-9820840.

STOC'02, May 19-21, 2002, Montreal, Quebec, Canada.

General Terms

Theory

Keywords

Computational Learning Theory, Disjunctive Normal Form, DNF, Representations of Boolean Functions, Certificates, Algorithms, Complexity Theory

1. INTRODUCTION

The learnability of DNF formulas is a central problem in computational learning theory, and has been studied in a variety of learning models ([3, 9, 14, 13, 18, 19, 22, 26, 27, 4, 6]). Here we present new results on learning DNF formulas, using DNF formulas as hypotheses, in the membership and equivalence query model (sometimes called the *exact* model) of Angluin [2].

There are two types of potential barriers to polynomial-time learning of a class C in the exact model, *information theoretic* and *complexity theoretic*. An information theoretic barrier exists when the number of polynomial-size queries that would need to be asked in order to learn C is super-polynomial, even assuming unlimited computational time. A complexity theoretic barrier exists when learning C requires the solution of a computationally difficult problem, so that the existence of a polynomial time learning algorithm would have complexity theoretic implications (e.g. it might imply that RP = NP).

Most of the results in this paper address questions of *polynomial-query learnability*, that is, determining whether particular classes C can be learned using a polynomial number of polynomial-size queries. Thus these results concern the information theoretic barrier to learning. Showing that a class is not polynomial-query learnable immediately implies that it is not polynomial-time learnable. Showing that a class *is* polynomial-query learnable does not imply that is polynomial-time learnable. However, it does show that there is no information-theoretic barrier to learning the class in polynomial time.

Proper Learning of DNF: We consider the following question: at what point does the class of k-term DNF formulas cease to be learnable with membership and proper equivalence queries, when the value of k is increased as a function of n?

The class of k-term DNF formulas consists of all DNF formulas with at most k terms. A membership and *proper* equivalence query algorithm for learning k-term DNF formulas uses hypotheses that are members of the class of k-term DNF formulas. Bshouty et al. showed that $O(\sqrt{\log n})$-term DNF can be learned in polynomial time with membership and proper equivalence queries [13]. Their algorithm can easily be modified to use *strongly* proper equivalence queries, meaning that the hypotheses used have no more terms than the target. On the negative side, Pillaipakkamnatt and Raghavan showed that if a barely plausible complexity-theoretic assumption[1] holds, then there exists some constant $c > 0$ such that DNF formulas with more than $(\log n)^c$ terms cannot be learned in polynomial time using membership and strongly proper equivalence queries [26]. We extend both results.

First, we show that $O(\log n)$-term DNF can be learned with a polynomial *number* of membership and strongly proper equivalence queries. This proves there is no information theoretic barrier to properly learning this class in polynomial time.

Second, we show that for any $\epsilon > 0$, $\Omega((\log n)^{(3+\epsilon)})$-term DNF formulas cannot be learned with a polynomial number of membership and strongly proper equivalence queries, and hence cannot be learned in polynomial time. We show a slightly weaker result for proper (as opposed to strongly proper) learning. [2]

It is worth emphasizing that these and other such results in this paper do not depend on any unproven complexity theoretic assumptions.

It remains an open question, for strongly proper exact learning, where in the range between $\sqrt{\log n}$ and $(\log n)^{(3+\epsilon)}$ the transition from polynomial-time learnability to non-learnability occurs, and where in the range between $\log n$ and $(\log n)^{(3+\epsilon)}$ the transition from polynomial-query learnability to non-learnability occurs. Thus we leave open the question of whether $O(\log n)$-term DNF can be learned in polynomial time with membership and proper equivalence queries. We do show that if $\log n$-term DNF formulas can be learned in polynomial time in the proper PAC + membership query model, then they can be learned in polynomial time (with high probability) with membership and proper equivalence queries. This is interesting, because while polynomial-time learnability with membership and proper equivalence queries is known to imply polynomial-time learnability (with high probability) in the PAC+membership query model [2], the opposite relation does not hold in general (assuming the existence of one-way functions) [8]. We also observe that there is a relation between the learnability problem for $\log n$-term DNF and a restricted version of a classical DNF truth table minimization problem.

Improper exact learning of $\log n$-term DNF: Another avenue to explore in trying to extend the border of learnability of k-term DNF formulas in polynomial time is to allow a certain amount of "impropriety." A natural approach is to use DNF formulas with more than k terms as hypotheses. Blum and Rudich, Bshouty et al., and Bshouty gave algorithms for learning $\log n$-term DNF formulas that do precisely this [10, 13, 11]. (There are also improper algorithms for learning $\log n$-term DNF formulas that use non-DNF hypotheses [21, 12, 5].) Strangely, while these results match the information theoretic results mentioned above, no known improper algorithm extends learnability beyond our upperbound of $(\log n)^3$ for proper learning.

In Section 3.1, we present what can be regarded as a simplified version of the algorithms of Blum et al. and Bshouty et al. for learning $\log n$-term DNF. Our algorithm is an easy consequence of the fact [15] that a k-term DNF formula has at most 2^k prime implicants. It is not as simple an algorithm as Bshouty's algorithm in [11] because it relies on universal sets, but we believe it is worth noting.

Exact learning of general DNF: Can our results for learning k-term DNF formulas be extended to learning general DNF formulas? In Section 4, we show that m-term DNF formulas cannot be polynomial-query learned by a membership and equivalence query algorithm that uses $m \cdot f(n)$-term DNF formulas as hypotheses. Here $f(n)$ is $o(\frac{\sqrt{n}}{\log n})$. This is admittedly a modest start, but if this result could be improved to make $f(n)$ superpolynomial in n and m, it would show that general DNF formulas are not polynomial-query learnable, and hence not polynomial time learnable, with membership and proper equivalence queries.

Read-Thrice DNF etc.: In some contexts, the size of a DNF formula is considered to be the number of literal occurrences in the formula. In Section 4.2, we prove negative results for this notion of size and then use those results to prove that the class of read-thrice DNF formulas (DNF formulas in which each variable appears at most thrice) does not have polynomial-size certificates of non-membership. It follows that read-thrice DNF cannot be learned in polynomial time with membership and proper equivalence queries.

This improves earlier results. Non-learnability of read-thrice DNF formulas was first shown under the assumption NP$\neq$co-NP in [1] and later under the weaker assumption P$\neq$NP [26]. Our result here removes all complexity-theoretic assumptions.

Organization of the paper: The rest of the paper is structured as follows. Section 2 provides the background for the remaining sections, Section 3 has positive results on proper learning of k-term DNF formulas and Section 4 has negative results.

2. PRELIMINARIES

Boolean Functions: Let $V_n = \{v_1, v_2, \ldots, v_n\}$ be a set of Boolean variables. An *assignment* a to V_n is an n-vector in $\{0,1\}^{V_n}$. We also view a as a function $a : V_n \to \{0,1\}$. For $b \in \{0,1\}$, $a_{v_i \leftarrow b}$ is the assignment to V_n such that $a_{v_i \leftarrow b}(v_j) = a(v_j)$ for $j \neq i$, and $a_{v_i \leftarrow b}(v_i) = b$. Similarly, $a_{\neg v_i}$ is the assignment to V_n such that $a_{\neg v_i}(v_j) = a(v_j)$ for $j \neq i$ and $a_{\neg v_i}(v_i) = \neg a(v_i)$.

In general, all functions in this paper may be assumed to be Boolean unless otherwise specified. Let f be a function defined on V_n. The *dimension* of f is n. Let a be an assignment to V_n. If $f(a) = 1$ then a is a *positive assignment* of f, otherwise it is a *negative assignment*. An assignment a is *justifying* for v_i (w.r.t. f) if $f(a) \neq f(a_{\neg v_i})$. A variable v_i is *relevant* for f if there is a justifying assignment for v_i. For $b \in \{0,1\}$ and $v_i \in V_n$, $f_{v_i \leftarrow b}$ denotes the function defined on V_n such that for all $a \in \{0,1\}^{V_n}$, $f_{v_i \leftarrow b}(a) = f(a_{v_i \leftarrow b})$.

[1]The assumption was that there exists some constant $\epsilon > 0$ such that NP is not contained in DTIME(2^{n^ϵ}).

[2]Negative results for proper learning are harder to prove than for strongly proper learning, roughly speaking because for $z' < z$, proving that z'-term DNF is not properly learnable does not immediately imply that z-term DNF is not properly learnable.

Implicants, DNF Formulas: A *literal* over a variable v_i is either v_i or $\overline{v_i}$. An *implicant* of a function $f(v_1, \ldots, v_n)$ is a conjunction t of literals over a subset of the variables in V_n such that for $a \in \{0,1\}^{V_n}$, $t(a) = 1 \Rightarrow f(a) = 1$. An implicant t of f is *prime* if deleting any literal from t causes t not to be an implicant of f. For a conjunction t' of literals, we write $t' \subseteq t$ if the literals of t' are a subset of the literals of t. The notation $|t|$ denotes the number of literals in t. For a Boolean function f, we write $|f|$ to denote the *DNF-size* of f, i.e., the smallest number of terms needed to represent f as a DNF formula. (If f is a constant function, then $|f| = 0$.) A DNF formula is *irredundant* if no term or literal within a term can be deleted without changing the function computed by the formula.

Certificates, Universal Sets: Let $A \subseteq \{0,1\}^{V_n}$. Two functions f and g defined on V_n are *consistent on* A if for all $a \in A$, $f(a) = g(a)$.

A *certificate* that a property P holds for a function f defined on V_n is a set $A \subseteq \{0,1\}^{V_n}$ such that for all functions g defined on V_n, if g does not have property P, then f and g are not consistent on A. The *size of a certificate* A is the number of assignments in A.

Let $C = \cup_{n>0} C_n$ be a set of functions, where C_n denotes the functions in C defined on V_n. Let $t : \mathcal{N} \to \mathcal{N}$. The class C has *certificates of non-membership of size* $t(n)$ if for all functions $g \notin C$ of dimension $n > 0$, g has a certificate of size at most $t(n)$ that $g \notin C$.

An (n, k)-universal set is a set U of assignments to V_n such that for all subsets $\{v_{i_1}, \ldots, v_{i_k}\}$ of V_n of size k, and for all $\sigma \in \{0,1\}^k$, there exists an assignment $a \in U$ such that $\sigma = a(v_{i_1})a(v_{i_2})\ldots a(v_{i_k})$. (This definition assumes $n \geq k$. If not, we define an (n, k) universal set to be the set of all $2^n \leq 2^k$ assignments to V_n.)

Learning models: We use the standard exact model [2] of learning with membership and equivalence queries, as applied to the problem of learning classes of Boolean formulas. Here, the task of the learning algorithm is to "learn" an unknown *target* formula f that is a member of a known class (i.e. set) of formulas C. We assume that each $f \in C$ has associated with it a natural number $s(f)$, called the *size* of f. For our purposes, C will be a set of DNF formulas, and we define $s(f)$ to be the number of terms of f.

Where it does not cause confusion, we blur the distinction between the formula f and the function (concept) it represents.

The learning algorithm is allowed to make two types of queries to gather information about f. A *membership query* asks "What is $f(a)$?," where a is an element of the domain of f. The answer to the query is the value of $f(a)$. An *equivalence query* asks "Is h equivalent to f?" where h is a representation of a function with the same domain and range as f. If h represents the same function as f, the response is "yes," otherwise the response is an element a in the domain of f such that $f(a) \neq h(a)$. The element a is called a *positive counterexample* if $f(a) = 1$, and a *negative counterexample* if $f(a) = 0$. If $h \in C$, the equivalence query is *proper*. If, in addition, $s(h) \leq s(f)$, h is *strongly proper*.

The learning algorithm is required to *exactly identify* f, that is, to output a hypothesis c that represents exactly the same function as f. If a membership and equivalence query algorithm exactly identifies every possible target formula from C, we say that the algorithm *learns* C.

An algorithm that learns with membership and *proper* equivalence queries uses proper equivalence queries and outputs a proper hypothesis (one in C). The analogous requirement holds for *strongly proper* equivalence queries.

We denote by C_n the formulas in C representing functions on n variables. For $f \in C_n$, we let $\sigma(f)$ denote the size of the smallest representation in C_n that is equivalent to f, i.e., $\sigma(f) = \min_{g \in C_n}\{s(g) | g \equiv f\}$.

We describe the running time of a learning algorithm as a function of the dimension n of the target formula f, and sometimes also as a function of $\sigma(f)$. An algorithm for learning C is *polynomial-time* if there exists a polynomial $p(x, y)$, such that for all $n > 0$, and for all target formulas $f \in C_n$, the algorithm runs in time at most $p(\sigma(f), n)$.

A membership and equivalence query algorithm for learning C is *polynomial-query* if there exists a polynomial $p(x, y)$ such that for all $n > 0$, and for all target formulas $f \in C_n$, the number of queries made by the algorithm is at most $p(\sigma(f), n)$, and the hypotheses used in the equivalence queries (and the output hypothesis) have size at most $p(\sigma(f), n)$.

A class C is *polynomial-time learnable* (*polynomial-query learnable*) with membership and equivalence queries if there exists a polynomial-time (polynomial-query) algorithm for learning C with membership and equivalence queries. The analogous definitions hold for proper and strongly proper equivalence queries.

2.1 The connection between certificates and learning

We exploit known relationships between certificates and learning (from [17]), herein stated as a single proposition. Although the original results were more general, Proposition 1 is stated only for classes of Boolean formulas. (A weaker version of Proposition 1 appeared in [16].) We assume that the formulas are encoded as bit strings in a standard way. We also assume that C and s are well-behaved in the following ways: (1) C is such that testing whether a given formula g is in C can be done in time polynomial in the length of the encoding of g, (2) for some polynomial $p(x, y)$, the number of bits in the encoding of any formula $f \in C_n$ is at most $p(s(f), n)$ and (3) Given the encoding of a formula $f \in C_n$, $s(f)$ can be computed in time polynomial in n and the number of bits in the encoding of f. Note that for f a DNF formula, our definition of $s(f)$ as the number of terms of f satisfies the last two properties.

We define the size of an arbitrary Boolean function g (with respect to a class of formulas C) to be the size of the smallest formula in C representing g. We denote the size of g by $\sigma(g)$. If g cannot be represented by any formula in C, we define its size to be infinite.

We denote by $C_{m,n}$ the set of formulas $f \in C_n$ such that $s(f) \leq m$.

Proposition 1. *[17] Let C be a class of Boolean formulas with associated size function s such that C and s are well-behaved as described above. Suppose there exist functions $p(m, n)$ and $q(m, n)$, increasing in m, such that for all $m, n > 0$, if g is a Boolean function defined on V_n such that $\sigma(g) > p(m, n)$, then there exists a certificate of size at most $q(m, n)$ proving that $\sigma(g) > m$. Then C can be learned by a membership and proper equivalence query algorithm that makes at most $O(\sigma(f)(1 + q(\sigma(f), n)) \log |C_{\sigma(f),n}|)$ queries using hypotheses of size at most $p(\sigma(f), n)$ in learning any target $f \in C_n$.*

Conversely, suppose C can be learned by a membership and proper equivalence query algorithm that makes at most $z(m,n)$ queries and uses hypotheses of size at most $t(m,n)$ to learn a target formula $f \in C_{m,n}$. Then for all $m,n>0$, if a Boolean function g defined on V_n is such that $\sigma(g) > t(m,n)$, then there exists a certificate of size at most $z(m,n)+1$ proving that $\sigma(g) > m$.

Proposition 1 makes no reference to the running time of the algorithms. Although the proof of the first part of the proposition is constructive, even if $p(m,n)$ and $q(m,n)$ are polynomials, the constructed algorithm does not run in time polynomial in the size and dimension of the target formula; in general, the best that can be said is that these algorithms can be made to run in polynomial time with the aid of a Σ_2^p oracle [17, 20].

2.2 Basic propositions on k-term DNF

Chandra and Markowsky proved the following useful bound.

PROPOSITION 2. *[15] If a function f is representable by a k-term DNF formula, then f has at most 2^k prime implicants.*

The next two propositions are used by Bshouty et al. [13]. The second is implied by the first, based on the result of Naor and Naor that an (n,k) universal set of size $O(k2^{3k}\log n)$ can be constructed in time polynomial in n and 2^k [25].

PROPOSITION 3. *[13] Let f be a k-term DNF formula over the variables in V_n. Let t be a conjunction of literals over a subset of the variables in V_n. Let A_t be the set of assignments produced by setting the variables in t so that t is satisfied, and setting the other variables in V_n according to the assignments of an $(n-|t|,k)$-universal set. More precisely, if $\{v_{i_1},\ldots,v_{i_{n-|t|}}\}$ are the literals not appearing in t, then let U be an $(n-|t|,k)$-universal set and let $A_t = \{b \in \{0,1\}^{V_n} | t(b)=1$ and there exists a $u \in U$ such that $u(v_l) = b(v_{i_l})$ for all $l \in \{1,\ldots,n-|t|\}\}$. Then t is an implicant of f iff $f(a)=1$ for all $a \in A_t$.*

PROPOSITION 4. *[13] Let f be a k-term DNF formula over the variables in V_n. Given a membership oracle for f and a conjunction of literals t over variables in V_n, testing whether t is a prime implicant of f can be done in time polynomial in n and 2^k.*

3. PROPER LEARNING OF K-TERM DNF FORMULAS

3.1 A simple polynomial time improper algorithm for learning $\log n$-term DNF with DNF hypotheses

The following algorithm learns $\log n$-term DNF formulas with membership and equivalence queries, using n-term DNF formulas as hypotheses. The basic idea of the algorithm is the same as the standard algorithm for learning monotone DNF [2]: repeatedly collect prime implicants and add them to the hypothesis until a formula equivalent to the target is obtained.

The algorithm works in *phases*, maintaining at all times a current set S (initially empty) of prime implicants of the target f. Each phase starts with an equivalence query using the hypothesis h obtained from the disjunction of all the prime implicants in S. If the answer to the equivalence query is "yes", the algorithm stops and outputs h. Otherwise, a counterexample a is received; since the terms of h are prime implicants of f, $f(a)=1$.

The algorithm then produces a prime implicant p of f satisfied by a (and thus not already in S), adds it to S, and proceeds to the next phase. The production of p is done as follows: Let t be the conjunction of all literals satisfied by a. For each literal l of t, delete l from t and test, using the method in Proposition 4, whether t is still an implicant of f. If not, replace l in t and proceed to the next literal.

By Proposition 2, since f is a $\log n$-term DNF, f has at most n prime implicants, so the running time of the algorithm is polynomial.

3.2 Certificates for k-term DNF formulas

The simplicity of the algorithm of section 3.1 masks the difficulty of obtaining a *proper* learning algorithm for $\log n$-term DNF. In this section, we show that it is possible to at least get a polynomial-*query* proper learning algorithm for $\log n$-term DNF. We do this by showing that k-term DNF formulas have certificates of non-membership that are merely exponential in k (Theorem 1). We use the following lemma of Blum and Rudich.

LEMMA 1. *[10] If a is an assignment satisfying exactly one term s of a k-term DNF formula, then a is a justifying assignment (with respect to that formula) for all but at most k of the variables that appear in s.*

We can now prove the following theorem.[3]

THEOREM 1. *The class of k-term DNF formulas has certificates of non-membership of size*

$$O(nk^2 2^k \log n + k^3 2^{3k} (\log n)^2).$$

PROOF. Let $f(v_1,\ldots,v_n)$ be such that $|f|>k$. Consider the following procedure. To begin, let $f'=f$ and let $l=n$. Repeat the following until termination. If $|f'_{v_l\leftarrow 0}| \le k$ and $|f'_{v_l\leftarrow 1}| \le k$, then terminate. Otherwise, let c be such that $|f'_{v_l\leftarrow c}| > k$, let $f' = f'_{v_l\leftarrow c}$, and decrement l.

The procedure terminates, because setting all the variables of f' to constant values would produce a function f'' such that $|f''|=0$.

Upon termination, clearly $|f'|>k$. Also, $|f'_{v_l\leftarrow 0}| \le k$ and $|f'_{v_l\leftarrow 1}| \le k$. Because $f' = v_l f'_{v_l\leftarrow 1} \vee \overline{v_l} f'_{v_l\leftarrow 0}$, $|f'| \le 2k$.

Also upon termination, $f' = f_{v_{l+1}\leftarrow b_{l+1},\ldots,v_n\leftarrow b_n}$ for some $b_{l+1},\ldots,b_n \in \{0,1\}$. If A is a certificate that $|f'|>k$, then $\{a_{v_{l+1}\leftarrow b_{l+1},\ldots,v_n\leftarrow b_n} | a \in A\}$ is a certificate that $|f|>k$.

Thus we may assume without loss of generality that $|f| \le 2k$, and that there exists a variable $v_j \in V_n$ such that $|f_{v_j\leftarrow 0}| \le k$ and $|f_{v_j\leftarrow 1}| \le k$. (Essentially the same technique was used in [17].) For concreteness, let F be an

[3]In the full version of the paper we also show that k-term DNF formulas have certificates of non-membership of size $O(k^3 2^{(1+2\log_2 3)k})$. That is, for any fixed constant k, the class of k-term DNF formulas has constant-size certificates of non-membership. It is therefore learnable with a number of queries *linear* in n and exponential in k. An algorithm using *time* and number of queries linear in n but doubly exponential in k was previously developed by Berggren [7].

irredundant DNF formula representing f such that F has $|f|$ terms.

For each term t in F, let A_t be the set of assignments produced by setting the variables in t so that t is satisfied, and setting the other variables of F according to the assignments of an $(n-|t|, k)$-universal set (as in Proposition 3). Let $A = \cup_{t \in F} A_t$.

For each assignment $a \in A$, let $W(a) = \{a_{\neg v_i} | v_i \in V_n\}$ and let $W = \cup_{a \in A} W(a)$. Let $P(a)$ denote the set of variables v_i such that $f(a_{\neg v_i}) = 0$ (and hence $f(a) \neq f(a_{\neg v_i})$, since $f(a) = 1$).

For each assignment $a \in A$, define $X(a)$ as follows. Let the variables not in $P(a) \cup \{v_j\}$ be $\{v_{i_1}, \ldots, v_{i_{n-|P(a)\cup\{v_j\}|}}\}$, and let U be an $(n - |P(a) \cup \{v_j\}|, 2k)$-universal set. Then let $X(a)$ be the set of assignments $b \in \{0,1\}^{V_n}$ such that $b(v_i) = a(v_i)$ for all $v_i \in P(a)$, and there exists a $u \in U$ such that $u(v_l) = b(v_{i_l})$ for all $l \in \{1, \ldots, n - |P(a) \bigcup \{v_j\}|\}$. Note that if $v_j \notin P(a)$, then for each $b \in X(a)$, $X(a)$ also contains $b_{\neg v_j}$, and hence $|X(a)| = 2|U|$.

Let $Y(a)$ be the set of negative assignments of f in $X(a)$ and let $Y = \cup_{a \in A} Y(a)$.

We claim that $C = A \cup W \cup Y$ is a certificate that $|f| > k$. The theorem follows from the claim after some arithmetic, using the well-known fact (easy to show non-constructively) that there exists an (n, k) universal set of size $O(k2^k \log(n))$.

To prove the claim, let h be a function with $|h| \leq k$. We will show that C contains an assignment on which h and f differ. Fix H to be a DNF formula representing h with at most k terms.

If there exists an assignment that satisfies f and falsifies h, then some term of F is not an implicant of h. In this case, by Proposition 3, f and h are not consistent on A.

Now suppose every assignment satisfying f also satisfies h. Since $f(a) = 1$ for all $a \in A$, h is consistent with f on A. Let H' be a DNF consisting of a minimal set of terms from H such that H' is satisfied by all assignments in A. The function h' defined by H' is consistent with f on A, and H' has at most k terms. By Proposition 3, every assignment satisfying f also satisfies h'.

Since $|h'| \leq k < |f|$, $h' \neq f$ and so the DNF H' must contain at least one term s that is not an implicant of f. Furthermore, there must be an assignment a in A which satisfies s but falsifies all other terms in H'. (If not, deleting s from H' results in a smaller DNF which is still satisfied by the assignments in A, contradicting the minimality of H'.) Either $h'(b) \neq f(b)$ for some $b \in W(a)$ or $h'(b) = f(b)$ for all $b \in W(a)$. In the former case $h'(b) = 1$ and $f(b) = 0$, because all assignments satisfying f also satisfy h'. Since H' contains a subset of the terms of H, $h(b) = 1$ also. It follows that f and h are not consistent on W, and we are done.

In the latter case, since a satisfies only term s in H', by Lemma 1, $P(a)$ must include all but at most k of the variables appearing in in s. Let S be the set of these at most k variables. Since s is not an implicant of f, there exists an assignment d satisfying s but falsifying f and hence also falsifying $f_{v_j \leftarrow d(v_j)}$. Let F_j be a DNF representing $f_{v_j \leftarrow d(v_j)}$ such that F_j has $|f_{v_j \leftarrow d(v_j)}|$ terms. For each term of F_j, choose a literal falsified by d. Let T be the set of chosen literals. Since $|f_{v_j \leftarrow d(v_j)}| \leq k$, $|T| \leq k$. Every assignment b that, like d, satisfies the literals in T, and is such that $b(v_i) = a(v_i)$ for all $v_i \in P(a)$, $b(v_i) = d(v_i)$ for all $v_i \in S$, and $b(v_j) = d(v_j)$, will satisfy h' and falsify f. Because $|S| \leq k$ and $|T| \leq k$, such an assignment exists in $Y(a)$. Further, since h is satisfied by all assignments satisfying h', this assignment also satisfies h. Hence f is not consistent with h on $Y(a)$. □

By combining Theorem 1 with Proposition 1, and noting that the implied algorithm can be made strongly proper by repeating it for increasing values of k, we obtain the following corollary.

COROLLARY 1. *The class of* $\log n$*-term DNF formulas is polynomial-query learnable with membership and strongly proper equivalence queries.*

3.3 The open question for $\log n$-term DNF

The question of whether $\log n$-term DNF formulas can be learned in polynomial *time* with membership and proper equivalence queries remains open. The next theorem, proved in the full paper, shows that a possible approach to answering the question affirmatively would be to develop a polynomial-time algorithm in the weaker proper PAC + membership query model.

THEOREM 2. *If* $\log n$*-term DNF formulas are polynomial-time learnable in the proper PAC + membership query model, then they are polynomial-time learnable (with high probability) in the membership and proper equivalence query model.*

The problem of $\log n$-term DNF learnability is also related to a restricted version of the following classical problem, which is known to be NP-complete [23]: Given a truth table of a Boolean function f and an integer k, determine whether $|f| \leq k$. Note that the size of the input to this problem is exponential in n. Now consider a restriction of the NP-complete problem where k is fixed to be n. It is not too hard to show (by simulating the learning algorithm) that if $\log n$-term DNF is learnable in polynomial time with membership and proper equivalence queries, then this restriction of the NP-complete problem can be solved in time polynomial in its input size.[4]

4. NEGATIVE RESULTS FOR LEARNING DNF

We first prove lower bounds on the size of certificates for k-term DNF. The details of the construction are then used to prove a series of non-learnability results involving DNF formulas.

4.1 Non-learnability of DNF using minimum number of terms

We use the following construction of Umans [28, 29]. Let ϕ be a DNF formula. Let $X = a_1 \ldots a_N$ be an implicant of ϕ such that X is not equivalent to ϕ. Define

$$\phi' = \phi \wedge (\overline{a_1} \vee \overline{a_2} \vee \ldots \vee \overline{a_N}).$$

[4]If we fix k to be $\sqrt{n}$, instead of n, the resulting problem is easy. Since $\sqrt{n}$-term DNF formulas have at most $2^{\sqrt{n}}$ prime implicants, we can use the truth table to find all prime implicants of f, construct all $\sqrt{n}$-term DNF formulas made up disjunctions of those prime implicants, and test each for equivalence with f. This algorithm runs in time polynomial in the input size.

Multiplying out the terms of ϕ' (and deleting any resulting terms that are identically false), let $s_1, \ldots, s_m$ be the terms of the resulting DNF. Define new variables $w_1, w_2, \cdots, w_m$ and $d_1, d_2, \cdots, d_m$ and let

$$\gamma' = (\bigvee_{i=1}^{m} s'_i) \vee \bigvee_{i=1}^{N} (\bigvee_{j=1}^{m} u_i^j)$$

where $s'_i = w_1 \ldots w_{i-1} s_i w_{i+1} \ldots w_m$ and $u_i^j = \overline{a_i} d_j w_1 \ldots w_m$.

The first of the next two lemmas was shown by Umans.

Lemma 2. *[29] Let g' be the function represented by γ'. Then $|g'| = m(k+1)$, where k is the size of the smallest implicant X' of ϕ such that $X' \subseteq X$.*

Lemma 3. *If γ' is not representable as a DNF formula with at most $m(k+1)$ terms, then a certificate of this fact must include, for every conjunction X' consisting of k literals in $\{a_1, \ldots, a_N\}$, an assignment that falsifies ϕ and satisfies X'.*

Proof. Let $X' = a_{l_1} a_{l_2} \ldots a_{l_k}$.

Consider $\gamma'' = (\bigvee_{i=1}^{m} s'_i) \vee \bigvee_{i=1}^{k} \bigvee_{j=1}^{m} z_i^j$, where $z_i^j = u_i^j = \overline{a_{l_i}} d_j w_1 \ldots w_N$.

Since γ'' has $m(k+1)$ terms which are a subset of the terms of γ', the certificate must contain an assignment b such that $\gamma'(b) = 1$ and $\gamma''(b) = 0$. Thus b must set $\overline{a_{i^*}} d_{j^*} w_1 \ldots w_N$ to true for some i^* such that a_{i^*} is not in X' and some j^*. Also, b sets all the w_i's to 1 and d_{j^*} to 1, and falsifies every term in γ''. Hence b must satisfy X', lest b satisfy term $z_i^{j^*}$ of γ'' for some i. Finally, b must falsify $\phi \wedge (\overline{a_1} \vee \ldots \vee \overline{a_N})$ in order to falsify the s'_i's. Since b satisfies $\overline{a_{i^*}}$, it must therefore falsify ϕ. $\square$

We now define a specific DNF formula ϕ and apply Umans's construction to it. Let C be an unbounded fan-in circuit of AND, OR, NOT gates that computes the majority function over the set of variables $S = \{a_1, \ldots, a_N\}$ and has size $\theta(N)$. It is well known that such a circuit exists (see, for example, [24]). Let t be the number of gates of C. Let $c_1, \ldots, c_t$ be Boolean variables associated with the output of the t gates, with c_t associated with the output of the entire circuit C.

Define ϕ to be the DNF formula made up of disjunction of the term c_t and the terms collected by iterating i from 1 through t and applying the following rules: If the inputs to the i^{th} gate are associated with the variables $v_1, \ldots, v_l$ and the gate is an AND, include terms $c_i \overline{v_j}$ for all v_j, and the term $\overline{c_i} v_1 \ldots v_l$; if the gate is an OR then include terms $\overline{c_i} v_j$ for each v_j, and the term $c_i \overline{v_1} \ldots \overline{v_l}$; if the i^{th} gate is a NOT whose input is associated with the variable v, then include the terms $c_i v$ and $\overline{c_i}\ \overline{v}$.

Since C computes the majority function, it is easy to show the following proposition (we assume the output of the majority function is 1 if *at least* half its inputs are 1):

Proposition 5. *A conjunction $S' \subseteq S$ is an implicant of ϕ if and only if $|S'| \geq N/2$.*

Using this ϕ in Umans's construction, we get a particular DNF formula γ'. Let f be the function computed by γ'. Since $m = \theta(N^2)$, γ' is defined on $\theta(N^2)$ variables and $|f| = \theta(N^3)$. For simplicity, we assume in what follows that N is even.

Lemma 4. *$|f| = m(\frac{N}{2}+1)$ and if $k < \frac{N}{2}$, then any certificate that $|f| > m(k+1)$ has at least $\binom{N}{k} / \binom{\frac{N}{2}-1}{k}$ assignments.*

Proof. By Proposition 5 and Lemma 2, $|f| = m(\frac{N}{2}+1)$. Let $k < \frac{N}{2}$. By Lemma 3, a certificate that γ' cannot be represented by a DNF of size $m(k+1)$ must contain, for every conjunction $X' \subseteq X$ such that $|X'| = k$, an assignment that falsifies ϕ and satisfies X'. There are $\binom{N}{k}$ such X'. Every assignment that falsifies ϕ sets at most $N/2 - 1$ variables in S to 1, and hence satisfies at most $\binom{\frac{N}{2}-1}{k}$ conjunctions $X' \subseteq S$ such that $|S'| = k$. The lemma follows. $\square$

We now prove our main non-learnability results.

Theorem 3. *The following hold:*

1. *For any $\epsilon > 0$, $\Omega((\log n)^{(3+\epsilon)})$-term DNF is not polynomial-query learnable with membership and strongly proper equivalence queries.*
2. *DNF is not polynomial-query learnable with membership and strongly proper equivalence queries.*

Proof. The second statement follows immediately from the first. We prove the first.

Let f be as constructed above. Let n be the dimension of f. Then $n = \theta(N^2)$. Setting k to $\frac{N}{2} - 1$ in Lemma 4 shows that it takes a certificate of size at least $\binom{N}{\frac{N}{2}-1}$ to prove that $|f| > m(\frac{N}{2})$. Since $\binom{N}{\frac{N}{2}-1} = \Omega(\frac{2^N}{\sqrt{N}})$, it is not polynomial in n and in $|f|$. Since $|f| = \theta(N^3)$, it follows from Proposition 1 that for some $p(n) = \theta(n^{\frac{3}{2}})$, $p(n)$-term DNF is not polynomial-query learnable with membership and strongly proper equivalence queries.

We now use a padding trick to improve the above non-learnability result to $(\log n)^{3+\epsilon}$-term DNF. Let $d > 1$ be a constant such that $|f| \leq dN^3$. Let $n' = 2^{dN^{(3/(3+\epsilon))}}$. Without loss of generality, assume $n' > N$. Add irrelevant[5] variables to f so that it is defined on n' variables, and call the new function f'. Then $|f'| = |f| \leq dN^3 \leq (\log n')^{(3+\epsilon)}$ and it takes a certificate of size at least $\binom{N}{\frac{N}{2}-1}$ to prove that $|f'| > m(\frac{N}{2})$. Since $\binom{N}{\frac{N}{2}-1}$ is still superpolynomial in n', the second statement of the lemma then follows immediately from Proposition 1 and the fact that $|f'| \leq (\log n)^{(3+\epsilon)}$. $\square$

The previous theorem implies that a polynomial-query algorithm for learning DNF with membership and proper equivalence queries would sometimes have to use hypotheses with more terms than the target formula. But how much larger would the hypotheses have to be? The following theorem gives a lower bound.

Theorem 4. *For any function $t(n) \in o(\frac{\sqrt{n}}{\log n})$, m-term DNF formulas cannot be polynomial-query learned by a membership and equivalence query algorithm that uses $m \cdot t(n)$-term DNF formulas as hypotheses.*

[5]A padding using relevant variables can also be obtained using the idea of "twinned" variables, which is described in the full paper.

PROOF. Let f be as defined above and n be the dimension of f. For $k = \omega(\log N)$ and $k < \frac{N}{2} - 1$, $\binom{N}{k}/\binom{\frac{N}{2}-1}{k}$ is superpolynomial in n. The ratio between $|f|$ and $m(k+1)$ is $(N/2+1)/(k+1)$. Since $n = \theta(N^2)$, the theorem follows from Proposition 1 and Lemma 4. $\square$

Theorems 3 and 4 apply to the membership and strongly proper equivalence query model. The following corollary applies to the membership and proper equivalence query model.

COROLLARY 2. *If $s(n)$ is $\Omega((\log n)^{(3+\epsilon)})$ and bounded above by a polynomial in n, then $s(n)$-term DNF is not poly-query learnable with membership and proper equivalence queries.*

PROOF. Let $s(n) = c(\log n)^{(3+\epsilon)} + q(n)$, where c is a constant and $q(n)$ is bounded above by a polynomial in n.

As in the proof of Theorem 3, pad f with irrelevant variables to produce a new function f' defined on n' total variables. Here, though, let $n' = \lceil 2^{((mN)/(2c))^{1/(3+\epsilon)}} \rceil$.

Let T be the set of assignments α to the n' variables of f' such that α sets all the w_i variables to 0. Then $f'(\alpha) = 0$ for all $\alpha \in T$. Since $|T|$ is superpolynomial in n', for sufficiently large n' it is possible to select a subset $T' \subseteq T$ such that $|T'| = \lfloor q(n') \rfloor$ and each pair of assignments in T' differs on at least two variables. Define a new function $\hat{f}$ on n' variables such that $\hat{f}(\alpha) = f'(\alpha)$ for $\alpha \notin T'$, $\hat{f}(\alpha) = 1$ for $\alpha \in T'$.

Each $\alpha \in T'$ is a justifying assignment (with respect to $\hat{f}$) for each of the n' variables of $\hat{f}$. It follows that any DNF for $\hat{f}$ contains an n'-literal term t_α for each $\alpha \in T'$, such that t_α is satisfied only by α. Thus $|\hat{f}| = m(N/2+1) + \lfloor q(n') \rfloor$.

A small modification to the proof of Lemma 3 (add the terms t_α to the formula γ'') shows that any certificate proving that $|\hat{f}| > m(N/2) + \lfloor q(n') \rfloor$ must contain an assignment that falsifies ϕ but satisfies X'. By the proof of Theorem 3, it therefore takes a certificate of size at least $\binom{N}{\frac{N}{2}-1}$, which is superpolynomial in n', to prove that $|\hat{f}| > m(N/2) + \lfloor q(n') \rfloor$. For sufficiently large N, $|\hat{f}| = m(N/2+1) + \lfloor q(n') \rfloor > s(n') = c(\log n')^{3+\epsilon} + q(n') \geq m(N/2) + \lfloor q(n') \rfloor$, so it also takes a certificate of at least size $\binom{N}{\frac{N}{2}-1}$ this size to prove that $|\hat{f}| > s(n')$. The corollary follows. $\square$

4.2 Non-learnability of DNF using minimum number of literals

Define $|f|_{lit}$ to be the minimum number of occurrences of literals in any DNF representing f. Umans [29] mentions that a lemma similar to Lemma 2 (with different bounds) holds if we consider the number of literal occurrences in γ'' rather than the number of terms. He also briefly sketches how the proof of Lemma 2 can be modified to to prove this similar lemma. Lemma 5, below, is an explicit statement of such a lemma. (A proof is given in the full paper.)

Let ϕ, ϕ', γ', $\{a_1, \ldots, a_N\}$, and m be as defined before the statement of Lemma 2. Thus, ϕ is an arbitrary DNF formula, and $a_1 \ldots a_N$ is an implicant of ϕ that is not equal to ϕ. Multiplying out the terms of ϕ' (and deleting any terms that are identically false), let q be the number of occurrences of literals in the resulting DNF. Recall that m is the number of terms in the resulting DNF.

LEMMA 5. *Let k be the size of the smallest implicant X' of ϕ such that $X' \subseteq X$ and let g' be the function represented by γ'. If $m > k-2$, then $q + m(m-1) + m(k-1)(m+2) < |g'|_{lit} \leq q + m(m-1) + mk(m+2)$.*

Now let ϕ be the specific DNF formula based on a majority circuit that is defined before Proposition 5. Then $m = \theta(N^2)$ and $q = \theta(N^2)$. Let f be the function represented by γ'. By Proposition 5 and Lemma 5, $q + m(m-1) + m(N/2-1)(m+2) < |f|_{lit} \leq q + m(m-1) + m(N/2)(m+2)$. Arguments similar to those in the previous section yield the following:

LEMMA 6. *Let $k < N/2$ and $m > k-2$. Then $|f|_{lit} > q + m(m-1) + mk(m+2)$ and any certificate that $|f|_{lit} > q + m(m-1) + mk(m+2)$ contains at least $\binom{N}{k}/\binom{\frac{N}{2}-1}{k}$ assignments.*

THEOREM 5. *The following hold:*

1. *For any $\epsilon > 0$, $(\log n)^{(5+\epsilon)}$-literal occurrence DNF is not polynomial-query learnable with membership and strongly proper equivalence queries.*
2. *DNF is not polynomial-query learnable with membership and proper equivalence queries, if the hypotheses are restricted to have no more literal occurrences than the target.*
3. *For any function $t(n) \in o(\frac{\sqrt{n}}{\log n})$, DNF formulas with q literal occurrences cannot be polynomial-query learned by a membership and equivalence query algorithm that uses DNF formulas with $m \cdot t(n)$ literal occurrences as hypotheses.*

4.3 Non-learnability of read-thrice DNF formulas

Let ξ be a DNF formula defined over V_n having s occurrences of literals. Let $0 \leq s' < s$. We construct a new formula η' from ξ as follows. For each $v_i \in V_n$, let $v_{i,1}, v_{i,2} \ldots, v_{i,s'}$ be s' new variables we call the *surrogates* of v_i.

Define DNF formulas $\xi_1, \ldots, \xi_n$ such that each ξ_i is obtained from ξ by replacing, for each variable v_j, all occurrences of v_j by $v_{i+j-1,1}$ (here and below, consider $n+r$ to be equal to r, for all $0 < r < n$). Because $\xi_1, \ldots, \xi_n$ are obtained from ξ by cyclicly permuting the variables of ξ_1, the total number of occurrences of each variable $v_{j,1}$ in $\xi_1, \ldots, \xi_n$ (with or without negation) is s, the total number of literal occurrences in ξ.

For $1 \leq i \leq n$, let $\{y_{i,1}, y_{i,2}, \ldots, y_{i,ns'}\}$ be new variables. Let

$$\begin{aligned}\eta \quad = \quad & y_{1,1}\overline{y_{2,1}}\ \overline{y_{3,1}} \ldots \overline{y_{n,1}}\xi_1 \ \vee \ \overline{y_{1,1}}y_{2,1}\overline{y_{3,1}} \ldots, \overline{y_{n,1}}\xi_2 \ \vee \\ & \ldots \ \vee \ \overline{y_{1,1}}\ \overline{y_{2,1}} \ldots \overline{y_{n-1,1}}y_{n,1}\xi_n\end{aligned}$$

$$\nu_v = \bigvee_{i=1}^{n} NC(v_{i,1}, \ldots, v_{i,s'})$$

and

$$\nu_y = \bigvee_{i=1}^{n} NC(y_{i,1}, \ldots, y_{i,ns'})$$

where $NC(x_1, \ldots, x_t) = x_1\overline{x_2} \vee x_2\overline{x_3} \vee \ldots \vee x_t\overline{x_1}$ (NC stands for non-consensus). Let

$$\eta' = \eta \ \vee \ \nu_v \ \vee \nu_y.$$

LEMMA 7. *There is a DNF formula ξ' equivalent to ξ that has at most s' total occurrences of literals iff there is a read-thrice DNF formula η'' equivalent to η'.*

PROOF. Suppose ξ' is equivalent to ξ, and ξ' has s' total occurrences of literals. For $1 \leq i \leq n$, define ξ'_i to be the formula obtained from ξ by replacing, for each variable v_j, all occurrences of v_j by $v_{i+j-1,1}$. Let $\tau = (y_{1,1}\overline{y_{2,1}}\,\overline{y_{3,1}}\ldots,\overline{y_{n,1}})\xi'_1 \vee(\overline{y_{1,1}}y_{2,1}\overline{y_{3,1}}\ldots\overline{y_{n,1}})\xi'_2\vee\ldots\vee(\overline{y_{1,1}}\,\overline{y_{2,1}}\ldots\overline{y_{n-1,1}}y_{n,1})\xi'_n$. Let $\tau' = \tau \vee \nu_v \vee \nu_y$.

The formula τ' is equivalent to η'. Create a read-thrice DNF η'' equivalent to τ' as follows. Distribute the conjunctions $\overline{y_{1,1}}\ldots\overline{y_{i-1,1}}y_{i,1}\overline{y_{i+1,1}}\ldots\overline{y_{n,1}}$ over the terms of the ξ'_i's to form a DNF. Each variable $y_{i,1}$ appears in the resulting DNF at most ns' times outside ν_v and ν_y (with or without negation), because each ξ'_i has s' occurrences of literals, and hence at most s' terms. For $1 \leq i \leq n$, do the following: For each of the occurrences of $y_{i,1}$ outside ν_v and ν_y, substitute a distinct surrogate of y_i.

Since the ξ'_i's are formed by cyclicly permuting the variables of ξ'_1, each $v_{i,1}$ now appears exactly s' times outside of ν_v and ν_y. For $1 \leq i \leq n$, do the following: For each of the s' occurrences of $v_{i,1}$ outside ν_v and ν_y, substitute a distinct surrogate of v_i. Let η'' be the resulting formula. Then η'' is a read-thrice DNF equivalent to η'.

Conversely, suppose there is a read-thrice DNF formula η'' equivalent to η'. Assume without loss of generality that η'' is irredundant.

We begin by showing that each variable $v_{i,j}$ must occur in one term of the form $v_{i,j}\overline{v_{i,k}}$, for some $k \neq j$. Let a be an assignment that sets all the $y_{s,t}$'s to 0, sets all surrogates of v_k to equal values if $k \neq i$, sets all surrogates of v_i except $v_{i,j}$ to 0, and sets $v_{i,j}$ to 1. Since a satisfies η', it must satisfy some term t of η''. The term t contains $v_{i,j}$, because otherwise $t(a_{v_{i,j}\leftarrow 0}) = 1$, contradicting that $\eta'(a_{v_{i,j}\leftarrow 0}) = 0$. The term t also contains $\overline{v_{i,k}}$ for some $k \neq i$, because otherwise $t(a_{v_{i,1}\leftarrow 1,\ldots,v_{i,s'}\leftarrow 1}) = 1$, contradicting that $\eta'(a_{v_{i,1}\leftarrow 1,\ldots,v_{i,s'}\leftarrow 1}) = 0$. Furthermore, since $v_{i,j}\overline{v_{i,k}}$ is an implicant of η'' and η'' is irredundant, $t = v_{i,j}\overline{v_{i,k}}$.

A symmetric argument shows that $v_{i,j}$ appears in a term of the form $\overline{v_{i,j}}v_{i,k}$ for some $k \neq j$. We call each term $v_{i,j}\overline{v_{i,k}}$ or $\overline{v_{i,j}}v_{i,k}$, $i \neq j$, a *non-consensus* term. Thus each variable $v_{i,j}$ appears at least twice in non-consensus terms of η''.

For $1 \leq i \leq n$, let η''_i be obtained from η'' as follows: Replace each surrogate of y_i by 1, each surrogate of y_j by 0 (for $j \neq i$), and each surrogate of v_j by $v_{j,1}$ (for $1 \leq j \leq n$). Delete any terms that become identically false.

The formula η''_i is equivalent to ξ_i. Suppose t is a term of η'' that becomes a term $\hat{t}$ in the construction of η''_i (i.e. t does not become identically false). Clearly t is not a non-consensus term. We claim that t becomes identically false in the construction of η''_j, for all $j \neq i$. Suppose not. Then t cannot contain any non-negated surrogate of a y variable, nor any non-consensus term, implying that it is satisfied by some assignment a setting all surrogates of the y variables to 0, and setting the surrogates of each v variable to equal values. But $\eta'(a) = 0$, contradiction. Therefore, the terms of η'' that do not become false in constructing η''_i are disjoint from the terms that do not become false in constructing η''_j, for all $j \neq i$. Since the total number of occurrences of surrogates of the v variables in η'' outside the non-consensus terms is at most $s'n$, by the pigeonhole principle there exists an i such that the number of literal occurrences in η''_i is at most s'. Since η''_i is equivalent to ξ_i, there is a DNF equivalent to ξ with at most s' literal occurrences. $\square$

LEMMA 8. *If η' is not read-thrice, then a certificate of this fact has size at least that of the minimum size of a certificate proving that ξ does not have an equivalent DNF with at most s' occurrences of literals.*

PROOF. Let C be a certificate that η' is not read-thrice. Let d be the size of the smallest certificate proving that ξ does not have an equivalent DNF with at most s' occurrences of literals. Suppose $|C| < d$. Define $C' \subseteq C$ to consist of all assignments in C that, for each v_j, set all surrogates of v_j to equal values, and for some i, set all surrogates of y_i to 1 and all surrogates of y_j, where $j \neq i$, to 0. For each $a' \in C'$, let a be the assignment to V_n such that if the surrogates of y_i are set to 1, then for all j, $a(v_j) = a'(v_{j+i-1,1})$ (which in turn equals $a'(v_{j+i-1,2}),\ldots,a'(v_{j+i-1,n})$). Clearly $\eta'(a') = \xi(a)$. Let C'' be the set of all such assignments a.

Since $|C''| < d$, there exists a formula τ with at most s' occurrences of literals such that for all $a \in C''$, $\tau(a) = \xi(a)$. For each i, let τ_i be produced from τ by replacing, for each variable v_j, all occurrences of v_j by $v_{i+j-1,1}$.

Then let $\beta = (y_{1,1}\overline{y_{2,1}}\ldots,\overline{y_{n,1}})\tau_1\vee(\overline{y_{1,1}}y_{2,1}\overline{y_{3,1}}\ldots,\overline{y_{n,1}})\tau_2 \vee\ldots\vee(\overline{y_{1,1}}\ldots,\overline{y_{n-1,1}}y_{n,1})\tau_n$ and let $\beta' = \beta\vee\nu_v\vee\nu_y$. Clearly β' is consistent with η' on the assignments in C. Form a DNF formula from β' by distributing each conjunction $\overline{y_{1,1}}\ldots\overline{y_{i-1,1}}y_{i,1}\overline{y_{i+1,1}}\ldots\overline{y_{n,1}}$ appearing in β' over the terms of τ_i. For each v_i, the total number of occurrences of surrogates of v_i in the resulting DNF, excluding occurrences in ν_v and ν_y, is at most s'. For each y_i, the total number of occurrences of surrogates of y_i, excluding occurrences in ν_v and ν_y, is at most ns'. Replacing each occurrence of a $v_{i,1}$ outside ν_v and ν_y with a distinct surrogate of v_i, and each occurrence of a $y_{i,1}$ outside ν_v and ν_y with a distinct surrogate of y_i, yields a read-thrice formula equivalent to β', and therefore consistent with η' on the assignments in C. This contradicts that C is a certificate that η' is not representable by a read-thrice DNF. $\square$

THEOREM 6. *Read-thrice DNF is not polynomial-query learnable using membership and proper equivalence queries.*

PROOF. Let ξ be equal to the DNF formula γ' defined in Section 4 and used in Section 4.2. Let $s' = q + m(m-1) + m(\frac{N}{2}-1)(m+2)$ and let f be the function expressed by γ', where m, q, and N are also defined as in Sections 4 and 4.2.

Recall that $|f|_{lit} > q + m(m-1) + m(\frac{N}{2}-1)(m+2)$ and $m = \theta(N^2)$. By Lemma 6, any certificate that $|f|_{lit} > q + m(m-1) + m(\frac{N}{2}-1)(m+2)$ contains at least $\binom{N}{\frac{N}{2}-1}$ assignments. By Lemma 7, there is no read-thrice DNF formula equivalent to η'. By Lemma 8, any certificate that the function represented by η' cannot be expressed by a read-thrice DNF formula contains at least $\binom{N}{\frac{N}{2}-1}$ assignments. Since the number of variables in η' is polynomial in N, and $\binom{N}{\frac{N}{2}-1}$ is not polynomial in N, the class of read-thrice DNF formulas does not have certificates of non-membership of size polynomial in n. The theorem follows from Proposition 1. $\square$

5. REFERENCES

[1] A. Aizenstein, T. Hegedus, L. Hellerstein, and L. Pitt. Complexity theoretic hardness results for query learning. *Computational Complexity*, 7(1):19–53, 1998.

[2] D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.

[3] Dana Angluin and Michael Kharitonov. When won't membership queries help? *Journal of Computer System Sciences*, 50(2):336–355, 1995.

[4] Jose L. Balcazar, Jorge Castro, David Guijarro. Abstract Combinatorial Characterizations of Exact Learning via Queries. In *Proc. of the Thirteenth Annual Conference on Computational Learning Theory*, 248-254, 2000

[5] A. Beimel, F. Bergadano, N. H. Bshouty, E. Kushilevitz, and S. Varricchio. On the applications of multiplicity automata in learning. In *IEEE Symposium on Foundations of Computer Science*, 349–358, 1996.

[6] Jose L. Balcazar, Jorge Castro, David Guijarro. A General Dimension for Exact Learning. In *Proc. of the Fourteenth Annual Conference on Computational Learning Theory*, 354-367, 2001

[7] U. Berggren. Linear-time deterministic learning of k-term DNF, In *Proc. of the Sixth Annual ACM Conference on Computational Learning Theory*, 1993.

[8] A. Blum. Separating distribution-free and mistake-bound learning models over the boolean domain. *SIAM Journal on Computing*, 23(5):990–1000, 1994.

[9] 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 *Proc. of Twenty-sixth ACM Symposium on Theory of Computing*, 1994.

[10] A. Blum and S. Rudich. Fast learning of k-term DNF formulas with queries. *Journal of Computer and System Sciences*, 51(3):367–373, 1995.

[11] N. H. Bshouty. Simple learning algorithms using divide and conquer. *Information Processing Letters*

[12] Bshouty N. H. Exact learning via the monotone theory. *Information and Computation*, 123(1):146–153, 1995.

[13] N. H Bshouty, G. A. Goldman, T. R. Hancock, and S. Matar. Asking questions to minimize errors. *Journal of Computer and System Sciences*, 52(2):268–286, 1996.

[14] Nader H. Bshouty. A subexponential exact learning algorithm for DNF using equivalence queries. *Information Processing Letters*, 59(1):37–39, 1996.

[15] A. Chandra and G. Markowsky. On the number of prime implicants. *Discrete Mathematics*, 24:7–11, 1978.

[16] T. Hegedus. Generalized teaching dimensions and the query complexity of learning. In *Proceedings of the 8th Annual Conference on Computational Learning Theory*, 108–117, 1995.

[17] L. Hellerstein, K. Pillaipakkamnatt, V. Raghavan, and D. Wilkins. How many queries are needed to learn? *Journal of the ACM*, 43(5):840–862, 1996.

[18] Jeffrey C. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *Journal of Computer and System Sciences*, 55(3):414–440, 1997.

[19] A. Klivans and R. Servedio. Learning DNF in time $2^{\tilde{o}(n^{1/3})}$. In *Proceedings of the 33rd Annual Symposium on Theory of Computing (STOC)*, 2001.

[20] J. Kobler and W. Lindner. Oracles in Σ_2^p are sufficient for exact learning. In *Proc. 8th Intl. Workshop on Algorithmic Learning Theory (ALT)*, 277–290, 1997.

[21] E. Kushilevitz. A simple algorithm for learning $o(\log n)$-term dnf. *Information Processing Letters*, 61(6):289–292, 1997.

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

[23] W. J. Masek. Some NP-complete set covering problems. Unpublished manuscript, 1979.

[24] D. E. Muller and F. P. Preparata. Bounds to complexities of networks for sorting and for switching. *Journal of the Association for Computing Machinery*, 22:195–201, 1975.

[25] J. Naor and M. Naor. Small bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22:838–856, 1993.

[26] K. Pillaipakkamnatt and V. Raghavan. On the limits of proper learnability of subclasses of DNF formulas. *Machine Learning*, 25(2-3):237–263, 1996.

[27] J. Tarui and T. Tsukiji. Learning DNF by approximating inclusion-exclusion formulae. In *Proc. IEEE Conference on Computational Complexity*, 215–220, 1999.

[28] C. Umans. The minimum equivalent DNF problem and shortest implicants. In *IEEE Symposium on Foundations of Computer Science*, 556–563, 1998.

[29] C. Umans. Hardness of approximating Σ_2^p minimization problems. In *Proc. IEEE Symposium on Foundations of Computer Science*, 465–474, 1999.