Exact Learning when Irrelevant Variables Abound

David Guijarro* **Víctor Lavín**†

Department LSI, Universitat Politècnica de Catalunya,

Jordi Girona Salgado, 1-3, Barcelona 08034, Spain,

{*david, vlavin*}*@lsi.upc.es*

Vijay Raghavan‡

Box 1679-B, Computer Science Department,

Vanderbilt University, Nashville, TN 37235,

raghavan@vuse.vanderbilt.edu

Abstract

We prove the following results. Any Boolean function of $O(\log n)$ relevant variables can be exactly learned with a set of non-adaptive membership queries alone and a *minimum* sized decision tree representation of the function constructed, in polynomial time. In contrast, such a function cannot be exactly learned with equivalence queries alone using general decision trees and other representation classes as hypotheses.

Our results imply others which may be of independent interest. We show that truth-table minimization of decision trees can be done in polynomial time, complementing the well-known result of Masek that truth-table minimization of DNF formulas is NP-hard. The proofs of our negative results show that general decision trees and related representations are not learnable in polynomial time using equivalence queries alone, confirming a folklore theorem.

1 Introduction

Exact learning using queries is a well-studied model in computational learning. Of the many kinds of queries considered by Angluin [1], membership and equivalence queries have established themselves

*Supported by the Esprit EU BRA program under project 20244 (ALCOM-IT) and the EC Working Group Neuro-COLT2 - EP27150

†Supported by the Esprit EU BRA program under project 20244 (ALCOM-IT) and the EC Working Group Neuro-COLT2 - EP27150

‡This work was supported by NSF grant CCR-9510392.

as the standard combination to be used for exact learning.

Some results that characterize learnability with a polynomial number of polynomially sized queries are known. For example, Angluin [2] showed that the presence of the *approximate fingerprint* property for a concept class is a sufficient condition for non-learnability with equivalence queries alone. Using this characterization, Angluin showed that equivalence queries do not suffice for learning deterministic or nondeterministic finite automata, context-free grammars, or general CNF or DNF formulas. Later on, Gavaldà [11] proved this property to be not only sufficient but also a necessary condition. On the other hand, Goldman and Kearns [13] showed that if a concept class is learnable with a polynomial number of membership queries alone, then it has *polynomial teaching dimension*. Moreover, the converse holds provided the concept class is projection closed, as shown by Hellerstein et al. [15]. Finally, it is known [15] that a concept class is learnable with a polynomial number of equivalence and membership queries if and only if that class has *polynomial certificates*.

Decision trees are a popular representation of boolean functions, which form the basic inference engine used in many machine learning programs. The learnability of decision trees has been well-studied in the learning theory community. For example, it is known [14] that μ-decision trees are exactly learnable with equivalence and membership queries, but neither type of query alone suffices for polynomial learning. Bshouty [4] has shown that general decision trees are learnable with extended equivalence queries (the hypotheses used are depth-three formulas) and membership queries. It is not known whether general decision trees are learnable with proper equivalence queries and membership queries, nor is it known whether decision trees are properly PAC-learnable with or without membership queries.

In this paper, we focus on concepts that depend on very few variables. Since Littlestone's seminal work [17] on this topic, such concept classes have been well studied in learning theory [6, 7, 10]. Recently, Damaschke [8, 9] studied exact learning of Boolean functions when irrelevant attributes abound, primarily in the model of learning with membership queries alone. He was able to show that a set X of *nonadaptive* membership queries can be constructed in time polynomial in n, the number of variables, so that X suffices to exactly learn every Boolean function of $O(\log n)$ variables in polynomial time. Moreover, a truth-table representing the learned Boolean function can be constructed in polynomial time as well. Here, we offer a simple alternative proof of this result and show in addition that a decision tree of *minimum* size to represent the learned function can also be constructed in polynomial time. In contrast, we show that such functions cannot be learned with equivalence queries alone using as hypotheses decision trees or other popular representation classes. Our result implies a folk theorem that general decision trees cannot be properly learned with equivalence queries alone.

Our positive result shows that truth-table minimization of decision trees can be done in polynomial time. This may be contrasted with the wellknown result of Masek cited in Garey and Johnson's book [12] that truth-table minimization of DNF formulas is NP-hard.

The rest of the paper is organized as follows. Section 2 has definitions used in the rest of the paper. Section 3 has our positive results on exact learning with non-adaptive membership queries when irrelevant variables abound and Section 4 has the negative results for exact learning such concepts with equivalence queries.

2 Preliminaries

We use standard terminology in learning theory. See [2] for more detailed descriptions of the learning model and a formalization of *concept classes*, *representation classes*, *examples*, etc.

We consider learning algorithms that make equivalence queries or membership to a teacher. The teacher answers the equivalence queries according to a target concept $c \in \mathcal{C}$. More precisely, an *equivalence query* is a representation $r \in \mathcal{R}$ of some concept in $\mathcal{C}$, and the answer is either YES, if the concept represented by r is equivalent to c, or a counterexample in the symmetric difference of c and the concept represented by r, otherwise. A *membership query* is an example and the answer is either YES or NO, depending how the target concept classifies that example.

We are interested in boolean concepts or, in other words, functions defined over a finite set of variables $\{x_1, \ldots, x_n\}$. The classes of representations we consider here are the following:

1. *DT (Decision Trees).* A decision tree d is a binary tree where the leaves are labeled either 0 or 1, and each internal node is labeled with a variable. Given an *assignment* $\alpha \in \{0,1\}^n$, $d(\alpha)$ is evaluated by starting at the root and iteratively applying the following rule, until a leaf is reached: let the variable at the current node be x_i; if the value of α at position i is 1 then branch right; otherwise branch left. If the leaf reached is labeled 0 (resp. 1) then $d(\alpha) = FALSE$ (resp. $TRUE$). The size of a decision tree is its number of nodes. The decision-tree size (DT-size) of a boolean function f is the size of the smallest decision tree that can represent f.

2. *CDNF.* A CDNF formula is a pair (f, g), where f is a DNF formula and g is a CNF formula and $f \equiv g$. It is convenient to define the size of such a formula as $\max\{n, m, p\}$, where n is the number of variables over which the formulas f and g are defined, m is the number of terms in f, and q is the number of clauses in g. The CDNF-size of a boolean function is the size of the smallest CDNF formula that can represent it.

   Equivalent but slightly different definitions of CDNF formulas and their sizes can be found in [4] and [15]. In [4], it is shown that CDNF formulas can be exactly learned in polynomial time with extended equivalence queries (of depth-3 formulas) and membership queries. In [15], it is shown that CDNF formulas are polynomial-query learnable with proper equivalence queries and membership queries. Here, we show that this latter result is tight: we show that proper equivalence queries alone do not suffice for polynomial-query learnability.

3. *Self-Dual DNF (Self-Dual CNF).* A DNF (CNF) formula f is self-dual if $f(\alpha) \neq f(\bar{\alpha})$ for all $\alpha \in \{0,1\}^n$, where $\bar{\alpha}$ is the bitwise complement of α. We direct the reader to [3] for work on the self-dual formulas.

4. $\log n$*–DNF* $\cap \log n$*–CNF.* This class contains the functions representable both as DNF formulas whose terms have at most $\log n$ literals, and CNF formulas whose clauses have at most $\log n$

literals. For our purposes, we consider a representation in this class to be a pair (f, g), where f is a $\log n$-DNF formula and g is a $\log n$-CNF formula. The size of such a pair is the same as its CDNF size.

There are several results in learning theory about this interesting class. Notice that this class contains the class of $\log n$-depth decision trees, shown to be exactly learnable in polynomial time [16] with membership queries. (The output of the learning algorithm is not a $\log n$-depth decision tree but a representation based on Fourier coefficients.) In [5], it was shown that $\log n$*–DNF* $\cap$ $\log n$*–CNF* is learnable with membership queries and an NP-oracle, in a representation of depth-3 boolean formulas. Finally, in [15] it was shown that this class is properly learnable with membership queries and an oracle for NP $\cap$ co-NP. It is open whether this class is properly learnable with membership queries alone in polynomial time. Here we show that the class is not polynomial-query learnable with equivalence queries alone.

5. *BP (Branching programs).* A branching program is a directed, acyclic graph, with a unique node of in-degree 0 (called the *root*), and two nodes of out-degree 0 (called *leaves*) , one labeled 0, and the other labeled 1; each non-leaf node of the graph contains a variable, and has outdegree exactly two. Assignments are evaluated following the same rule as for decision trees.

   A branching program is in the class $\log n$*–depth BP* if its longest path has length at most $\log n$. The size of a branching program is the number of nodes in it.

   General branching programs are known to be not learnable with equivalence and membership queries. (See [19] for a detailed study of subclasses of branching programs according to learnability.) In [15], it is shown that the subclass of $\log n$-depth branching programs is properly learnable with membership queries and an oracle for NP $\cap$ co-NP—it is open whether the oracle can be dispensed with. Here we show that the class is not polynomial-query learnable with equivalence queries alone.

3 Learning with Non-Adaptive Membership Queries

Let $\mathcal{F}_{n,r}$ denote the set of Boolean functions over n variables which depend on at most r variables. The following is a restatement of a result of Damaschke [8, 9].

Theorem 1 *There exists a set X of assignments such that a truth-table for any function $f \in \mathcal{F}_{n,r}$ can be learned solely by making membership queries on the set X. Moreover, the set X and the truth-table can be constructed in time polynomial 2^r and n.*

We independently arrived at a proof of the above theorem; however, Damaschke's result has much sharper bounds for the construction of X and the time to do the inference of the truth-table than our proof provides. For the sake of completeness, we outline our simple proof of the above theorem.

Use the deterministic construction in [18] to obtain a (n, r)-universal set B of size $2^{O(r)} \log n$ in time polynomial in 2^r and n. Let B' be the set of all assignments at Hamming distance 1 from an assignment in B so that $|B'| \leq n \cdot |B|$. Let $X = B \cup B'$. Now, to learn any function $f \in \mathcal{F}_{n,r}$, make membership queries on all the assignments in X. Clearly, if we get a different answer for two assignments which differ only in variable v, then v must be in the set R of relevant variables of f. Conversely, the facts that B is a (n, r)-universal set and that $|R| \leq r$ guarantee us that if a variable v is in R then there are two assignments in X which differ only in variable v for which we will get different answers to membership queries. Thus we can find the set R in the claimed time. Once R is found, we can construct a truth table of $2^{|R|}$ entries by searching in X for every projection over the variables in R and logging the answer to the membership query.

Next, we prove the following theorem to conclude that a decision tree of minimum size can be output for the learned function $f \in \mathcal{F}_{n,r}$ in time polynomial in 2^r.

Theorem 2 *A decision tree of minimum size representing a function f can be constructed in polynomial time given a complete truth-table of f.*

Proof: Let V be the set of r variables over which f is defined and let TT be a truth-table of f of 2^r entries.

We construct a table P of size 3^r that is indexed by partial assignments on the r variables. In position α of this array we will put a decision tree of minimum size that computes the function f_α obtained by projecting f to the partial assignment α. That is,

$$P[\alpha] = \text{a minimum size DT for} f_\alpha$$

Note that once P is computed then a minimum size decision tree for f is obtained by reading off the entry $P[\lambda]$, where λ is the empty partial assignment.

For any tree T, let $|T|$ denote the size of T and for any Boolean function g, let $|g|$ denote the decision tree size of g. For any partial assignment α and for any variable v not assigned a value in α, let $\alpha \cup \{v \leftarrow b\}$ denote the partial assignment by extending α by assigning the value $b \in \{0, 1\}$ to v. Now, P can be constructed by using a dynamic programming approach by observing these facts:

1. For each (complete) assignment α over all r variables, the minimum size decision tree for f_α is just a constant 0 or 1 node obtained by looking up the value of $f(\alpha)$ in the truth-table TT.
2. For every partial assignment α over some set $X \subset V$ of fewer than r variables, if there exists a variable $v \in V - X$ such that $f_{\alpha \cup \{v \leftarrow 0\}} \equiv f_{\alpha \cup \{v \leftarrow 1\}}$, then $f \equiv f_{\alpha \cup \{v \leftarrow 0\}}$. Consequently, the entry $P[\alpha]$ can be filled by copying the entry for $P[\alpha \cup \{v \leftarrow 0\}]$.
3. For every partial assignment α over some set $X \subset V$ of fewer than r variables, if there exists no variable $v \in V - X$ such that $f_{\alpha \cup \{v \leftarrow 0\}} \equiv f_{\alpha \cup \{v \leftarrow 1\}}$, then

$$|f_\alpha| = \min_{v \in V - X} \{1 + |f_{\alpha \cup \{v \leftarrow 0\}}| + |f_{\alpha \cup \{v \leftarrow 1\}}|\}.$$

Consequently, the entry $P[\alpha]$ can be found by first finding a variable v which satisfies the above equation and then constructing a tree with v as the root node and $P[\alpha \cup \{v \leftarrow 0\}]$ $P[\alpha \cup \{v \leftarrow 1\}]$ as the left and right children respectively of v.

Clearly, the dynamic programming approach to filling the entries of P can be accomplished in time polynomial in 2^r. ■

An immediate corollary of the above theorems is that Boolean functions of $O(\log n)$ variables can be exactly learned using nonadaptive membership queries and a decision tree of minimum size for the learned function can be output in time polynomial in n.

4 Learning with Equivalence Queries

4.1 Approximate Fingerprints

We recall the following definitions from [2, 11]. Let $\mathcal{C}$ denote a class of concepts defined over an instance space of Σ^* (where Σ is a finite alphabet, $\mathcal{R}$ a representation class for $\mathcal{C}$, w a string from Σ^*, b an element of $\{0,1\}$, and $\alpha > 0$ a real number. We say that the pair $\langle w, b\rangle$ is an *α-approximate fingerprint with respect to C* if

$$|\{c \in \mathcal{C} : \chi_c(w) = b\}| < \alpha|\mathcal{C}|.$$

That is, the number of concepts in $\mathcal{C}$ that agree with the classification b of w is strictly less than the fraction α of the total number of concepts in $\mathcal{C}$.

A *sequence of concept classes* is a sequence $C_1, C_2, C_3, ...$ such that each C_i is a class of concepts. Such a sequence is *polynomially bounded* (with respect to a given representation $\mathcal{R}$) if there exists a polynomial $p(n)$ such that for all $c \in C_n$, there is a representation $r \in \mathcal{R}$ such that $c(r) = c$ and $|r| \leq p(n)$.

A representation of concepts $\mathcal{R}$ is said to have the *approximate fingerprint* property if there exists a polynomially bounded sequence of concept classes $T_1, T_2, T_3, ...$, and a polynomial $p(n)$ such that for any polynomial $q(n)$, for infinitely many n, T_n contains at least two concepts and if $r \in \mathcal{R}$ and $|r| \leq q(n)$ then there exists a string $w \in \Sigma^*$ of length at most $p(n)$ such that $\langle w, \chi_{c(r)}(w)\rangle$ is a $1/q(n)$-approximate fingerprint with respect to T_n. That is,

$$|\{c \in T_n : \chi_c(w) = \chi_{c(r)}(w)\}| < |T_n|/q(n),$$

where $c(r)$ is the concept associated to r.

Since we are dealing with boolean concepts defined over n variables, the polynomial $p(n)$ that bounds the length of the words, can be set to n. The notion of approximate-fingerprints was introduced as a means of proving non-learnability of certain classes in the equivalence model using the following theorem.

Theorem 3 [2]
Let $\mathcal{R}$ be a representation class of concepts with the approximate fingerprint property. Then there is no algorithm for exact identification of $\mathcal{R}$ using polynomially many equivalence queries of polynomial size.

4.2 The Negative Results

We now show that all the classes considered here cannot be learned with equivalence queries alone, even when the number of relevant variables is $O(\log n)$. Since we do not restrict the hypothesis space of equivalence queries to ones which do contain only hypothesis of $O(\log n)$ relevant variables, our results imply that the general classes (with potentially more relevant variables) are also not learnable with equivalence queries alone. Indeed, this is how the results are stated and proved below.

Our results for all the classes considered here use the same central idea. We first illustrate the technique for the class DT of decision trees.

In order to prove approximate fingerprints for DT, we need the following lemma.

Lemma 4 *Let $k > 0$ be some fixed constant. If d is a decision tree of size at most n^k, over n variables, then there exists an assignment α such that either*

(a) α contains at most $k \log n$ 1's and $d(\alpha) = 1$, or

(b) α contains at most $k \log n$ 0's and $d(\alpha) = 0$.

Proof: Since each of the 2^n assignments reaches some leaf of d after being evaluated, and d has at most n^k leaves, d must have a path p of length at most $\log(n^k) = k \log n$. The assignment α can be constructed by satisfying p first and then filling the rest of the positions with 1's or 0's as necessary.

Theorem 5 *The class DT has approximate fingerprints.*

Proof: Let T_n be the target class of functions that represent the majority of $\log n$ variables chosen out of n variables. Clearly, $|T_n| = \binom{n}{\log n}$.

Clearly, the DT-size of each function in T_n is at most n. Therefore, it suffices to show that for any fixed constant k, and for infinitely many n, and any DT d of size at most n^k, the assignment α whose existence is guaranteed by Lemma 4 satisfies the property that fewer than $|T_n|n^{-k}$ of the the functions in T_n classify α the same way as d.

Without loss of generality, assume that $\log n$ is an even integer. (This avoids floors and ceilings.) The number of functions in T_n that accept (reject) an assignment with $k \log n$ 1's (respectively, 0's) is

$$\sum_{p=\frac{\log n}{2}}^{\log n} \binom{k \log n}{p} \binom{n - k \log n}{\log n - p}.$$

Using n^k as an upperbound for $\binom{k \log n}{p}$ this sum turns out to be less than

$$n^k \sum_{p=0}^{\frac{\log n}{2}} \binom{n - k \log n}{p},$$

which is less than

$$n^k (1 + \frac{\log n}{2}) \binom{n - k \log n}{\frac{\log n}{2}} = r(n).$$

Now $\frac{r(n)}{|T_n|}$ can be shown to be less than

$$\frac{n^k (1 + \frac{\log n}{2}) (\log n)^{\frac{\log n}{2}}}{(n - \log n + 1)^{\frac{\log n}{2}}},$$

which is less than n^{-k} for $n > 2^{ck}$, where c is some constant. ■

We now show that the remaining classes considered here have approximate fingerprints.

Corollary 6 *The following classes have approximate fingerprints:*

1. *CDNF Formulas*
2. *Self-Dual DNF Formulas and Self-Dual CNF Formulas.*
3. $\log n$*–DNF* $\cap$ $\log n$*–CNF*
4. $\log n$*–depth BP*

Proof: The proof follows easily if we first use the following statements for each of the above classes instead of Lemma 4.

Let $k > 0$ be some fixed constant.

- *For every CDNF formula $h = (f, g)$ over n variables, of size at most n^k, there exists an assignment α that either (a) contains at most $k \log(2n)$ 1's and $h(\alpha) = 1$, or (b) contains at most $k \log(2n)$ 0's and $h(\alpha) = 0$.*

  To see this, note that the DNF formula $f + \overline{g}$ has at most $2 \cdot n^k$ terms and is identically true, i.e., it accepts all 2^n assignments. Consequently, such a formula must have a term t that accepts at least $\frac{2^n}{2 \cdot n^k}$ assignments, whence the length of t is at most $k \log(2n)$. The assignment α can now be constructed by satisfying the literals in t and then padding the remaining variables with 0's or 1's according to whether t is a term in f or corresponds to a clause in g respectively.

- *For every self-dual DNF formula f of at most n^k terms over n variables, there exists an assignment α that either (a) contains at most $k\log(2n)$ 1's and $f(\alpha)=1$, or (b) contains at most $k\log(2n)$ 0's and $f(\alpha)=0$.*

  Let $\sim f$ denote the DNF formula obtained from f by complementing every literal in every term of f. Then, by the definition of self-dual formulas, it follows that $\sim f \equiv \overline{f}$. Therefore, $f + \sim f$ is identically true and the existence of α is proved in an identical manner to that of CDNF formula h above.

  Note that an analogous statement holds for self-dual CNF formulas.

- *For every $\log n$–DNF $\cap$ $\log n$–CNF formula f of size at most n^k, there exists an assignment α that either (a) contains at most $\log(n)$ 1's and $f(\alpha)=1$, or (b) contains at most $\log(n)$ 0's and $f(\alpha)=0$.*

  This is obvious.

- *For every $\log n$-depth BP h of size at most n^k over n variables, there exists an assignment α that either (a) contains at most $k\log(n)$ 1's and $h(\alpha)=1$, or (b) contains at most $k\log(n)$ 0's and $h(\alpha)=0$.*

  This follows along the same lines as the proof of Lemma 4.

With this change, the proof of Theorem 5 can be used with very slight modifications to show that all of these classes have approximate fingerprints. In particular, the target class T_n remains the same for all the classes, and each function in T_n has a representation of size at most n in each of these classes. ■

A final note. Since the target class defined in Theorem 5 is monotone, the monotone versions of all the classes mentioned above (including DT) also have approximate fingerprints, and are therefore not learnable with equivalence queries alone.

5 Acknowledgments

We thank O. Watanabe, T. Tsukiji and C. Domingo for communicating the problem of exact learning Boolean functions of few relevant variables with membership queries. We also thank L. Hellerstein for pointing us to the work of Peter Damaschke.

References

[1] D. Angluin. Queries and Concept Learning. *Machine Learning*, 2, (1988), 319-342.

[2] D. Angluin. Negative Results for Equivalence Queries. *Machine Learning*, 5, (1990), 121-150.

[3] J. C. Bioch and T. Ibaraki. Complexity of Identification and Dualization of Positive Boolean Functions. *Information and Computation*, 123:50–63, 1995.

[4] N. Bshouty. Exact Learning Boolean Functions via the Monotone Theory. *Information and Computation*, 123(1):146–153, Nov 1995.

[5] N. Bshouty, R. Cleve, S. Kannan, and C. Tamon. Oracles and Queries that are Sufficient for Exact Learning. *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory*, pages 130–139, 1994.

[6] A. Blum, L. Hellerstein, N. Littlestone. Learning in the Presence of Finitely or Infinitely Many Attributes. *Journal of Computer and System Science*, pages 50:32–40, 1995.

[7] N.H. Bshouty, L. Hellerstein. Attribute-efficient Learning in Query and Mistake Bound Models. *Proceedings of the Ninth Conference on Computational Learning Theory*, pages 235–243, 1996.

[8] P. Damaschke. Adaptive versus Nonadaptive Attribute-Efficient Learning. *Proceedings of the 35th ACM Symposium on Theory of Computing*, pages 590–596, 1998.

[9] P. Damaschke. Computational Aspects of Parallel Attribute-Efficient Learning. *To appear in: Proceedings of the 9th International Workshop on Algorithmic Learning Theory*, 1998.

[10] A. Dhagat, L. Hellerstein. PAC Learning with Irrelevant Attributes. *Proceedings of the 35th IEEE Foundations of Computer Science*, pages 64–75, 1994.

[11] R. Gavaldà. On the Power of Equivalence Queries. *Proceedings of the 2nd European Conference on Computational Learning Theory*, (1993), 193-203.

[12] M.R. Garey and D.S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York, 1979.

[13] S. Goldman, M. Kearns. On the Complexity of Teaching. *Proceedings of the 4rd Workshop on Computational Learning Theory*, (1991), 303-314.

[14] T. Hancock. Identifying μ-Formula Decision Trees with Queries. *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 23–37, 1990.

[15] L. Hellerstein, K. Pillaipakkamnatt, V. Raghavan, D. Wilkins. How Many Queries are Needed to Learn? *Proceedings of the 27th ACM Symposium on the Theory of Computing,* (1995), 190-199.

[16] E. Kushilevitz and Y. Mansour. Learning Decision Trees Using the Fourier Spectrum. *SIAM Journal of Computing*, 22(6):1331–1348, 1993.

[17] N. Littlestone. Learning Quickly When Irrelevant Variables Abound. *Machine Learning*, 2:285–318, 1988.

[18] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *Proceedings of the 22nd ACM Symposium on the Theory of Computing,* (1990), 312-323.

[19] D. Wilkins. Learning Restricted-Read Branching Programs with Queries. *Ph.D. Thesis*, Vanderbilt University, 1995.