Yale University
Department of Computer Science

Lower Bounds on Learning Random Structures with Statistical Queries

Dana Angluin David Eisenstat
Leonid (Aryeh) Kontorovich Lev Reyzin

YALEU/DCS/TR-1421
December 2009

Lower Bounds on Learning Random Structures with Statistical Queries

Dana Angluin* David Eisenstat† Leonid (Aryeh) Kontorovich‡ Lev Reyzin§

Abstract

We show that random DNF formulas, random log-depth decision trees and random deterministic finite acceptors cannot be weakly learned with a polynomial number of statistical queries with respect to an arbitrary distribution.

1 Introduction

Polynomial time learning algorithms have been given for random log-depth decision trees by Jackson and Servedio [2] and montone and general random DNF formulas by Sellie [4, 5] with respect to the uniform distribution. These algorithms can be implemented using statistical queries, which were introduced by Kearns [3] to characterize a wide class of noise tolerant learning algorithms.

Blum et al. [1] gave upper and lower bounds on the number of statistical queries required to learn concepts from a given class in terms of a distribution-dependent statistical query dimension of the class. A corollary of their characterization is that parity functions with $\log n$ relevant variables cannot be weakly learned using a polynomial number of statistical queries with respect to the uniform distribution. Because log-depth decision trees and DNF formulas can represent such parity functions, they are not weakly learnable using a polynomial number statistical queries with respect to the uniform distribution.

The key difference between these negative results and the positive results cited above is the random choice of a structure (formula or decision tree) to be learned. We show that the random choice of a structure is not sufficient for the positive results cited above; in addition it is necessary to make some assumptions about the input distribution.

In particular, we consider the problem of learning the behavior of a random structure using statistical queries, where the distribution on examples is arbitrary, and therefore may depend on the random structure to be learned – a natural setting for boosting. For the cases when the random structure is a DNF formula, a log-depth decision tree, or a deterministic finite acceptor, we show that with probability $1 - o(1)$, there is a distribution on the inputs that embeds a nontrivial parity computation in the random structure. This implies that a polynomial number of statistical queries is not sufficient even for weak learning of these random structures.

*Department of Computer Science, Yale University, New Haven, CT, USA. Research supported by the National Science Foundation under Grant CCF-0916389.

†Department of Computer Science, Brown University, Providence, RI, USA.

‡Department of Computer Science, Ben Gurion University of the Negev, Israel.

§Yahoo! Research, New York, NY, USA. This material is based upon work supported by the National Science Foundation under a National Science Foundation Graduate Research Fellowship and under Grant # 0937060 to the Computing Research Association for the Computing Innovation Fellowship program.

2 Random DNF formulas

In this section we prove the lower bound for random DNF formulas. The general framework is analogous for random log depth decision trees and random deterministic finite acceptors, but the embeddings of the parity problems are somewhat more complex.

We adopt the model used by Sellie [5] of random DNF formulas on n variables $V = \{v_1, \ldots, v_n\}$. Each term is a conjunction of $c\log(n)$ literals created by selecting a random subset of $c\log(n)$ variables and negating each variable independently with probability $1/2$. The target random DNF formula is a disjunction of independently selected terms. Sellie gives a polynomial time algorithm to learn a random DNF with at most $n^c \log\log(n)$ terms under the uniform distribution on inputs.

For clarity of description, we first consider random monotone DNF formulas, in which the step of negating variables is omitted. The general case is described later. Given a positive integer ℓ, let $n = 2^{3\ell}$; then we have $\ell = \frac{1}{3}\log(n)$ and $2^\ell = n^{1/3}$. Let ϕ denote a random monotone DNF formula of $t = 2^{\ell-1}$ terms, where each term contains ℓ variables.

We show that with probability $1 - o(1)$, no variable occurs more than once in ϕ, that is, ϕ is a read once formula. Because each term consists of a randomly chosen set of ℓ variables, the probability that each set avoids the variables chosen in previous sets is

$$\left(1 - \frac{\ell}{n}\right)\left(1 - \frac{2\ell}{n}\right)\cdots\left(1 - \frac{(t-1)\ell}{n}\right),$$

which is $1 - O(\log(n)/n^{1/3})$. Thus we assume that all variables in ϕ are distinct. For example, for $\ell = 3$ and $n = 512$, a possible value might be

$$\phi = v_{14}v_{133}v_{170} \vee v_{22}v_{101}v_{337} \vee v_{55}v_{266}v_{413} \vee v_{10}v_{332}v_{507}.$$

2.1 Embedding Parity

We consider the problem of learning a parity function with ℓ relevant variables from a total of $m = n/t$ variables $Y = \{y_1, y_2, \ldots, y_m\}$, given examples drawn from the uniform distribution. Because $\ell = \Theta(\log(n))$ and $m = \Theta(n^{2/3})$, such a function cannot be weakly learned using a polynomial number of statistical queries with respect to the uniform distribution [1].

Let L denote the set of literals over Y and consider any mapping f from V to L. An assignment a to the variables Y induces an assignment $a(f)$ to the variables V by $a(f)(v_i) = a(f(v_i))$, that is, the value assigned to variable v_i is the value assigned by a to the literal $f(v_i) \in L$. Similarly, a distribution D over assignments a to variables in Y induces a distribution D_f over the assignments $a(f)$ to variables in V. A mapping f from V to L is an *equi-grouping* if exactly $n/(2m) = t/2$ variables in V are mapped to each literal in L.

Fix a parity function with ℓ relevant variables from Y. It can be expressed as a DNF formula ψ of $t = 2^{\ell-1}$ terms consisting of those terms containing exactly one literal for every relevant variable, in which the number of positive literals is odd. For example, if the relevant variables are $\{y_{33}, y_{57}, y_{108}\}$, we have

$$\psi = y_{33}y_{57}y_{108} \vee y_{33}y'_{55}y'_{108} \vee y'_{33}y_{57}y'_{108} \vee y'_{33}y'_{57}y_{108}.$$

Note that ψ and ϕ each contain t terms of ℓ literals each. We describe an embedding of ψ into ϕ.

Choose an arbitrary bijection between the terms of ϕ and the terms of ψ, and for each term, an arbitrary bijection between the variables in the term of ϕ and the literals in the corresponding

term of ψ. If v_i is a variable in ϕ, let $g(v_i)$ be the corresponding literal in ψ. Because the variables in the terms of ϕ are all distinct, g maps exactly $t/2$ distinct variables of ϕ to each literal of ψ.

Extend g arbitrarily to an equi-grouping by dividing the remaining variables in V into groups of size $t/2$ and mapping each group to a unique one of the remaining literals in L. The uniform distribution U on assignments to Y induces the distribution U_g on assignments to V, in which groups of $t/2$ variables are assigned the same value, the groups corresponding to a literal and its complement receive complementary values, and groups corresponding to different variables are independent.

For every assignment a to the variables Y, $\psi(a) = \phi(f(a))$, so this construction embeds the parity function ψ into the random monotone DNF formula ϕ.

2.2 Reduction

Our goal now is to describe a reduction showing that a learning algorithm A that weakly learns a random monotone DNF formula (over n variables with t terms and ℓ variables per term) with respect to an arbitrary distribution using statistical queries could be used to weakly learn an arbitrary parity function (over m variables with ℓ relevant variables) with respect to the uniform distribution using statistical queries.

For a parity learning problem we are given the set of variables Y with $m = |Y|$, the desired error tolerance ϵ (where weak learning means that predictions are correct with probability at least $1/2+\epsilon$) and access to a statistical query oracle $STAT(U, \psi)$ for the uniform distribution U on assignments to Y and an unknown parity function ψ with ℓ relevant variables. Recall that a statistical query specifies two arguments: a polynomial time evaluatable predicate mapping each labeled example to 0 or 1, and a tolerance $\alpha > 0$, and the answer is a number that is within α of the expected value of the predicate on labeled examples drawn according to the distribution.

For the reduction, we arbitrarily choose an equi-grouping h mapping V to L. We then run A with variables V and error tolerance ϵ, simulating access to a statistical query oracle $STAT(U_h, \phi)$, where ϕ is a DNF formula that embeds ψ with respect to h. (That is, $\psi(a) = \phi(a(h))$ for all assignments a to the variables Y.)

We simulate a statistical query (χ, α) made by A as follows. The parameter χ is a function that takes as input an assignment to V and a label from $\{0, 1\}$, and returns as output an element of $\{0, 1\}$. The tolerance α is a rational number between 0 and 1. This query is transformed to (χ', α), where

$$\chi'(a, b) = \chi(a(h), b).$$

That is, χ' transforms the assignment a to Y into the assignment $a(h)$ to V, keeps the label b, and applies χ. The query (χ', α) is asked of the statistical query oracle $STAT(U, \psi)$ for the parity problem, and the answer is returned as the answer to the query (χ, α). Even though we do not know a correct function ϕ embedding ψ, this transformation allows us to answer the statistical queries of A correctly for some such ϕ.

Moreover, the induced distribution over formulas ϕ is uniform over all monotone read once formulas with n variables, t terms and ℓ variables per term, a $1 - o(1)$ fraction of the whole. Thus, if A weakly learns random monotone DNF formulas with n variables, t terms and ℓ variables per term with respect to an arbitrary distribution using statistical queries and fails with probability $o(1)$, this reduction gives a randomized algorithm to learn with probability $1 - o(1)$ any parity

function over m variables with ℓ relevant variables to the same level of accuracy ϵ using the same number of statistical queries with the same tolerances.

The extension to general (non-monotone) DNF formulas is straightforward; a general DNF formula with n variables, t terms and ℓ variables per term is read once with probability $1-o(1)$, and embedding a parity function ψ into a general read once formula just requires mapping literals (rather than variables) over V to literals over Y and modifying the definition of the induced assignment $a(g)$ appropriately. Thus we conclude the following.

Theorem 1. *No algorithm can with probability $1-o(1)$ weakly learn random monotone (or general) DNF formulas with $n = 2^{3\ell}$ variables, $t = 2^{\ell-1}$ terms and ℓ variables per term with respect to an arbitrary distribution using a polynomial number of statistical queries.*

2.3 Extensions

This technique can also be used to show lower bounds for DNF formulas with more or fewer terms. If $t^2\ell$ is $o(n)$, then a random DNF with n variables, t terms and ℓ variables per term will be read once with probability $1 - o(1)$. If the number of terms is larger than $2^{\ell-1}$, it can be trimmed by choosing one literal per excess term to fix to the value 0 so that the term is eliminated under the constructed distribution. If the number of terms is smaller than $2^{\ell-1}$, we can choose a number of literals per term to fix to the value 1 so that the term effectively becomes smaller under the constructed distribution. For example, we could use the same logic to embed parity functions with $\Theta(\log\log(n))$ relevant variables (which are still not weakly learnable with a polynomial number of statistical queries) by using $\Theta(\log(n))$ terms.

To handle these cases, instead of just choosing an equi-grouping h from V to L, the reduction first randomly chooses an appropriate number of variables from V to fix to 0 or 1, and then chooses an equi-grouping on the rest. The resulting induced distribution on assignments to V is constant on some variables, and behaves as before on the rest.

3 Random Decision Trees

3.1 Model

Let $\Sigma = \{0, 1\}$ and for integers $n \geq 0$, let $[n] = \{0, \ldots, n-1\}$. A *decision tree* of integer depth $k \geq 0$ consists of maps $\alpha : (\bigcup_{\ell \in [k]} \Sigma^\ell) \to [n]$ and $\beta : \Sigma^k \to \Sigma$, where $n = 2^k$ is the number of variables, α determines the variable queried for a given history, and β determines the final labeling.

For a random decision tree, we take $k \geq 1$ to be a parameter. The values of α are chosen uniformly at random. The values of β are chosen so that for all histories $x \in \Sigma^{k-1}$, one of $\beta(x0)$ and $\beta(x1)$ is 0 and the other is 1, where both outcomes have equal probability. This is one of the models of random decision trees considered by Jackson and Servedio [2]; results for the others should be similar.

3.2 Embedding

Our goal here is to show that an arbitrary decision tree of depth k can be embedded with probability $1 - o(1)$ in a random decision tree of depth $3k$ (with a bit of work, $(2 + \epsilon)k$) by letting the joint distribution of variables depend on the random decision tree.

Formally, fix some decision tree (α, β) of depth k and let D be a distribution on Σ^n, where $n = 2^k$ as before. Let (α', β') be a random decision tree of depth $k' = 3k$ and let $n' = 2^{k'} = n^3$. For all $x \in \Sigma^k$, let $y(x) \in \Sigma^{k'-1-k}$ be chosen uniformly at random. Let H' be the set of prefixes of strings $xy(x)$. The set H' has $O(n \log n)$ elements, so by a union bound, with probability $1 - o(1)$, the map α' takes distinct values in $[n'] \setminus [n]$ on H'. We ignore this probability $o(1)$ case from now on.

We choose a distribution D' on $\Sigma^{n'}$ as follows. The variables indexed by $[n]$ are distributed as in D. For $x \in \Sigma^k$, we let variable $\alpha'(x)$ copy variable $\alpha(x)$. For $xz \in H'$ where z is a proper prefix of $y(x)$, fix variable $\alpha'(xz)$ to the unique $b \in \Sigma$ such that $xzb \in H'$. Fix variable $\alpha'(xy(x))$ to the unique $b \in \Sigma$ such that $\beta'(xy(x)b) = \beta(x)$. At this point, each variable in $[n]$ has at most $n - 1$ copies; choose unused variables uniformly at random without replacement to bring each up to exactly $n - 1$ copies.

The effect of these fixings is that statistical queries on (α, β, D) get identical results on (α', β', D'). Moreover, given some information that is uncorrelated with (α, β, D), there is a simple "compilation" process that achieves the reverse translation. This information is how to compute the values of the variables indexed by $[n'] \setminus [n]$ from the values of the variables indexed by $[n]$. The learner does not have access to the random decision tree, so all it sees is that $n-1$ copies were made of each variable in $[n]$ and stored in random locations, and that the rest were fixed uniformly at random.

3.3 Parity

By the above embedding, any polynomial-time learner that can learn random decision trees with statistical queries under a distribution that depends on the tree can learn parity functions on $O(\log n)$ variables efficiently, which is impossible [1]. Using as a black box such a learner for random decision trees, we simply "fake" a random decision tree embedding the parity problem and translate the statistical queries to the parity oracle by the above method.

Theorem 2. *No algorithm can with probability* $1-o(1)$ *weakly learn random log depth decision trees with* n *variables with respect to an arbitrary distribution using a polynomial number of statistical queries.*

4 Random Deterministic Finite Acceptors

Let $\Sigma = \{0, 1\}$ and let $Q = [n]$. We consider the standard model of random deterministic finite acceptors. Let the entries of $\delta : Q \times \Sigma \to Q$ be chosen uniformly at random, let $q_0 = 0$, and let $F \subseteq Q$ be chosen uniformly at random. Let $\phi : \{1, 2, \ldots\} \to \Sigma^*$ be the bijection defined by

$$\begin{aligned} \phi(1) &= \epsilon \\ \phi(2m) &= \phi(m)0 \\ \phi(2m+1) &= \phi(m)1. \end{aligned}$$

In other words, ϕ carries m to its binary expansion after the first 1. Let $\ell = \lceil \log^2 n \rceil$ and let $V = \{\phi(m) : m \in \{\ell, \ldots, 2\ell - 1\}\}$ be the set of *variables*. V is the set of paths to leaf nodes in a complete binary tree with ℓ leaves. Our goal is to make a random machine compute parity on a relevant subset of variables $U \subseteq V$. The resulting class cannot be learned efficiently with statistical queries. The input is a list of those variables with value 1 in any order, where each variable is

followed by a string that will prepare the machine to accept the next variable. Even though each variable may have a different string, the contents and assignments of these strings will not reveal any information about the relevant variables.

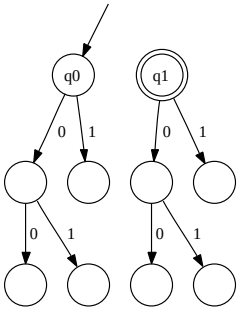

Figure 1: The even state is q_0 and the odd state is q_1. With probability $1 - o(1)$, there are two non-overlapping trees with $\ell - 1$ nodes rooted at q_0 and q_1. We don't yet commit to the outgoing arcs of the leaves.

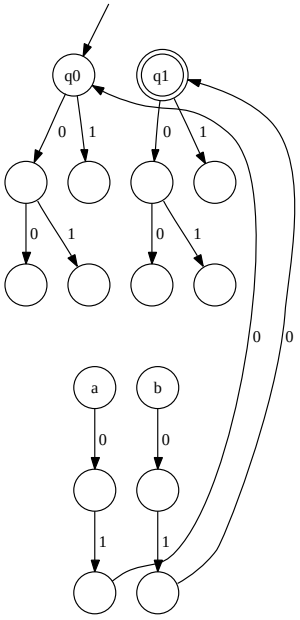

Figure 2: With constant probability, a pair $(a, b) \in Q^2$ chosen uniformly at random can reach (q_0, q_1) via the same string while avoiding the trees.

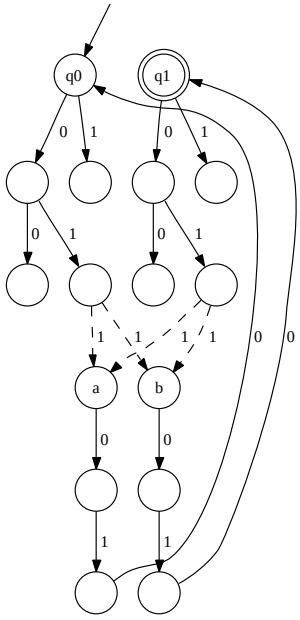

Figure 3: Now we choose the outgoing arcs corresponding to variable 011. With constant probability, there is a path back to (q_0, q_1). The heavy dashes signify a relevant variable; the light ones, an irrelevant variable. These cases are equally likely and independent of the string to prepare for another variable.

We let $q_0 = 0$ be the even state and $q_1 = 1$ be the odd state. With probability 1/4, we have $q_0 \in Q \setminus F$ and $q_1 \in F$. Let $V' = \{\phi(m) : m \in \{1, \ldots, \ell - 1\}\}$, or the labels to the interior of a complete binary tree with ℓ leaves. Since $|V'| = O(\log^2 n)$, with probability $1 - o(1)$, there are two non-overlapping trees rooted at q_0 and q_1, that is, the set of states $R = \{\delta(q, v') : q \in \{q_0, q_1\},\ v' \in V'\}$ has cardinality $2|V'|$. In this case, $\delta(q, v)$ for $q \in \{q_0, q_1\}$ and $v \in V$ are all independent. We now show that with constant probability, for two random states $(a, b) \in Q^2$, there is a short string x such that $\{\delta(a, x), \delta(b, x)\} = \{q_0, q_1\}$ and neither path touches the states in R. The latter stipulation is important because it enables a bijection that swaps the values of $\delta(q_0, v)$ and $\delta(q_1, v)$, which allows us to conclude that the synchronizing strings don't give away the relevant variables.

The proof of existence of the short string is as follows. Let $(a, b) \in Q^2$ be chosen uniformly at random, assume n is even, and let $X = \{\phi(m) : m \in \{n^2/4, \ldots, n^2/2 - 1\}\}$. We show that if $x \in X$ is chosen u.a.r., then with probability $(2 - o(1))/n^2$, we have $\{\delta(a, x), \delta(b, x)\} = \{q_0, q_1\}$, that is, x is *good*. Also, if $y \in X$ is chosen independently, then the probability that both x and y are good is $(4 - o(1))/n^4$. By inclusion-exclusion, the probability that exactly one string in X passes is at least

$$\begin{aligned} |X|(2 - o(1))/n^2 - 2\binom{|X|}{2}(4 - o(1))/n^4 &= (1 - o(1))/2 - 2\binom{n^2/4}{2}(4 - o(1))/n^4 \\ &= (1 - o(1))/2 - (1 - o(1))/4 \\ &= (1 - o(1))/4. \end{aligned}$$

The key observation is that the success event of x and the success event of y are very close to being independent Bernoulli trials with success probability $2/n^2$. The strings under consideration have length roughly $2 \log n$. If on the final letter of the string we have reached two different states whose outward transitions have not been committed in any way, the success probabilities *will* be independent and exactly $2/n^2$. Of course, something may go wrong before then: we may touch a state in R or have the paths loop or touch one another. With only $O(\log^2 n)$ states to avoid however, this event is probability $o(1)$.

Finally, we observe that with constant probability, there will be $\log^2 n/8$ variables that function properly. This is enough to make $n^{\Theta(\log n)}$ parity functions, ensuring that a superpolynomial number queries are needed in expectation.

Theorem 3. *No algorithm can with probability* $1 - o(1)$ *weakly learn random deterministic finite acceptors with* n *states with respect to an arbitrary distribution using a polynomial number of statistical queries.*

5 Discussion

As described in Section 1, there are polynomial time learning algorithms for random decision trees and random DNF formulas with respect to the uniform distribution, and these algorithms can be implemented with statistical queries. However, it is open whether random deterministic finite acceptors of n states can be learned with respect to the uniform distribution on strings of length $\Theta(n^\alpha)$ for any positive constant α.

6 Acknowledgements

During parts of this work Aryeh Kontorovich was in the Department of Mathematics of the Weizmann Institute and Lev Reyzin was in the Department of Computer Science of Yale University.

References

[1] Blum, A., Furst, M., Jackson, J., Kearns, M., Mansour, Y., and Rudich, S. Weakly learning DNF and characterizing statistical query learning using Fourier analysis. In *STOC '94: Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 1994), ACM, pp. 253–262.

[2] Jackson, J. C., and Servedio, R. A. Learning random log-depth decision trees under uniform distribution. *SIAM J. Comput. 34*, 5 (2005), 1107–1128.

[3] Kearns, M. Efficient noise-tolerant learning from statistical queries. *J. ACM 45*, 6 (1998), 983–1006.

[4] Sellie, L. Learning random monotone dnf under the uniform distribution. In *COLT* (2008), pp. 181–192.

[5] Sellie, L. Exact learning of random DNF over the uniform distribution. In *STOC* (2009), pp. 45–54.