PROD. TYPE: COM
PP: 1-11 (col.fig.: nil)
TCS5793 MODC+
ED: Rajani Devi
PAGN: Wittal – SCAN: nil
ARTICLE IN PRESS
Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science ▮▮▮ (▮▮▮▮) ▮▮▮–▮▮▮
Theoretical Computer Science
www.elsevier.com/locate/tcs

On learning embedded midbit functions[☆]

Rocco A. Servedio*

Department of Computer Science, Columbia University, New York, NY 10027, USA

Abstract

A midbit function on ℓ binary inputs $x_1, \ldots, x_\ell$ outputs the middle bit in the binary representation of $x_1 + \cdots + x_\ell$. We consider the problem of Probably Approximately Correct (PAC) learning *embedded* midbit functions, where the set $S \subset \{x_1, \ldots, x_n\}$ of relevant variables on which the midbit depends is unknown to the learner.

To motivate this problem, we first point out that a result of Green et al. implies that a polynomial time learning algorithm for the class of embedded midbit functions would immediately yield a fairly efficient (quasipolynomial time) (PAC) learning algorithm for the entire complexity class *ACC*. We then give two different subexponential learning algorithms, each of which learns embedded midbit functions under any probability distribution in $2^{\sqrt{n}\log n}$ time. Finally, we give a polynomial time algorithm for learning embedded midbit functions under the uniform distribution.
© 2005 Published by Elsevier B.V.

Keywords: PAC learning; Embedded midbit functions

1. Introduction

A central goal of computational learning theory is to understand the computational complexity of learning various classes of Boolean functions. While much research has been devoted to learning syntactic classes such as decision trees, DNF formulas, and constant depth circuits, researchers have also considered various "semantically defined" classes as well. A natural and important class of this sort is the class of *embedded symmetric functions* which was studied by Blum et al. [5]. (Recall that a Boolean function is symmetric if its value depends only on the number of input bits which are set to 1.) An embedded symmetric function is a Boolean function which depends only on some subset of its input variables and is a symmetric function on this subset, i.e., it is a symmetric function whose domain is "embedded" in a larger domain containing irrelevant variables.

In this paper we give a detailed PAC (Probably Approximately Correct) learning analysis of an interesting and natural family of embedded symmetric functions, which we call *embedded midbit functions*. An embedded midbit function is defined by a subset $i_1, \ldots, i_s$ of variables from $\{1, \ldots, n\}$. The value of this embedded midbit function on an input $x \in \{0, 1\}^n$ is the value of the middle bit in the binary representation of $x_{i_1} + x_{i_2} + \cdots + x_{i_s}$. As described below, we show that the class of embedded midbit functions has many interesting properties from a PAC learning perspective.

[☆] This research was performed while the author was at Harvard University and was supported by a National Science Foundation Mathematical Sciences Postdoctoral Research Fellowship and by NSF Grant CCR-98-77049.

* Tel.: +1 212 939 7065; fax: +1 212 666 0140.
E-mail address: rocco@cs.columbia.edu
URL: http://www.cs.columbia.edu/rocco.

0304-3975/$ - see front matter © 2005 Published by Elsevier B.V.
doi:10.1016/j.tcs.2005.10.013

1.1. Our results

We first give a hardness result (Theorem 4) for learning embedded midbit functions in the standard PAC model of learning from random examples drawn from an arbitrary probability distribution. Using Green et al.'s characterization of the complexity class *ACC* [9], we observe that if there is a PAC learning algorithm for the class of embedded midbit functions which runs in polynomial time (or even quasipolynomial time), then the class *ACC* of constant-depth, polynomial-size circuits of unbounded fanin AND/OR/MOD$_m$ gates can also be PAC learned in quasipolynomial time. This would be a major breakthrough since, as described in Section 3, the fastest PAC learning algorithms to date for even very restricted subclasses of *ACC* require much more than quasipolynomial time. This hardness result strengthens an earlier hardness result of Blum et al. for embedded symmetric functions, and establishes an interesting connection between learning the "semantic" class of embedded midbit functions and learning rich syntactic classes. (We emphasize that this hardness result is independent of the hypothesis representation which the learning algorithm uses.)

While Theorem 4 implies that it may be difficult to learn embedded midbit functions efficiently under an arbitrary distribution, this does not mean that PAC learning algorithms for embedded midbit functions must require exponential time. In Section 4, we give two different subexponential time PAC learning algorithms, each of which can learn embedded midbit functions over n variables in time $n^{O(\sqrt{n})}$.

Finally, by means of a careful analysis of the correlation of single variables and pairs of variables with embedded midbit functions, we show in Section 5 that embedded midbit functions can be learned in polynomial time under the uniform distribution. Embedded midbit functions thus give a simple and natural concept class which seems to exhibit a large gap between the complexity of learning in the uniform distribution PAC model and the general (arbitrary distribution) PAC model.

2. Preliminaries

Throughout this paper S denotes a subset of the variables $\{x_1, \ldots, x_n\}$ and s denotes $|S|$. All logarithms are base 2.

Definition 1. For $S \neq \emptyset$ the embedded midbit function $M_S : \{0,1\}^n \to \{0,1\}$ is defined as $M_S(x) =$ the value of the $\lfloor \log(s)/2 \rfloor$th bit in the binary representation of $\sum_S x_i$, where we consider the least significant bit to be the 0th bit. (We take $M_\emptyset(x)$ to be identically 0.) The class C_{mid} of embedded midbit functions is $C_{\text{mid}} = \{M_S\}_{S \subseteq \{x_1,\ldots,x_n\}}$.

We write C_{sym} to denote the class of all embedded symmetric functions on $\{0,1\}^n$ as described in Section 1; note that $C_{\text{mid}} \subset C_{\text{sym}}$.

Definition 2. Given an embedded midbit function $M_S(x)$, let $f_s : \{0, 1, \ldots, s\} \to \{0,1\}$ be the unique function such that $M_S(x) = f_s(\sum_S x_i)$ for all $x \in \{0,1\}^n$. We say that f_s is the *basis function* of $M_S(x)$ and we refer to the $(s+1)$-bit string $f_s(0) f_s(1) \cdots f_s(s)$ as the *pattern* of f_s.

If f_s is the basis function for M_S then the pattern for f_s is a concatenation of strings of the form $0^{k(s)}1^{k(s)}$, where $k(s) = 2^{\lfloor \log(s)/2 \rfloor}$ and the concatenation is truncated to be of length precisely $s+1$. It is easy to see that $\sqrt{s}/2 < k(s) \leqslant \sqrt{s}$.

A function f is *quasipolynomial* if $f(n) = 2^{(\log n)^{O(1)}}$. We write $[a \bmod b]$ to denote the unique real number $r \in [0, b)$ such that $a = kb + r$ for some integer k.

2.1. The learning model

We work in the standard PAC learning model [17] and the uniform distribution variant of the PAC model. Let C be a class of Boolean functions over $\{0,1\}^n$. In the PAC model, a learning algorithm has access to a random example oracle $EX(c, \mathcal{D})$ which when invoked provides, in one time step, a labeled example $\langle x, c(x) \rangle \in \{0,1\}^n \times \{0,1\}$ where x is drawn from the distribution $\mathcal{D}$ over $\{0,1\}^n$. An algorithm A is a PAC learning algorithm for class C if the following holds: for all $c \in C$ and all distributions $\mathcal{D}$ over $\{0,1\}^n$, if A is given as input $\varepsilon, \delta > 0$ and A is given access to $EX(c, \mathcal{D})$, then with probability at least $1-\delta$ the output of A is a hypothesis $h : \{0,1\}^n \to \{0,1\}$ such that $\Pr_{x \in \mathcal{D}}[c(x) \neq h(x)] \leqslant \varepsilon$.

(Strictly speaking, the output of A is some particular representation of h such as a Boolean circuit.) Algorithm A is said to run in time t if (i) the worst case running time of A (over all choices of $c \in C$ and all distributions $\mathcal{D}$) is at most t, and (ii) for every output h of A and all $x \in \{0, 1\}^n$, $h(x)$ can be evaluated in time t.

If A satisfies the above definition only for some fixed distribution $\mathcal{D}$ (such as the uniform distribution on $\{0, 1\}^n$), then we say that A is a PAC learning algorithm for C under distribution $\mathcal{D}$.

3. Hardness of learning embedded midbit functions

In this section we show that learning embedded midbit functions is almost as difficult as learning a rich syntactic class which contains decision trees, DNF formulas, and constant depth circuits.

3.1. Background: hardness of learning C_{sym}

We first describe a result of Blum et al. which gives some evidence that the broader class C_{sym} of embedded symmetric functions may be hard to PAC learn in polynomial time. Let $C_{\log}$ denote the class of Boolean functions on n bits which have at most $\log n$ relevant variables. Note that like C_{sym}, the class $C_{\log}$ has the property that learning is no more difficult than finding relevant variables—in either case, once the set of relevant variables has been identified, learning is simply a matter of observing and filling in at most n "table entries" which define the function (these entries are the bits of the pattern for a function from C_{sym}, and are the values of the function on all $2^{\log n}$ inputs for a function from $C_{\log}$).

Building on this intuition, Blum et al. gave a polynomial time prediction-preserving reduction from $C_{\log}$ to C_{sym}, thus showing that if C_{sym} can be PAC learned in polynomial time then $C_{\log}$ can also be PAC learned in polynomial time. Since no polynomial time learning algorithm is yet known for $C_{\log}$, this gives some evidence that C_{sym} may not be learnable in polynomial time.

3.2. Hardness of learning C_{mid}

The class *ACC* was introduced by Barrington [2] and since been studied by many researchers, e.g. [1,3,4,9,12,18,19]. *ACC* consists of languages recognized by a family of constant-depth polynomial-size circuits with NOT gates and unbounded fanin AND, OR and MOD_m gates, where m is fixed for each circuit family. In the context of learning theory, *ACC* is quite an expressive class, containing as it does polynomial size decision trees, polynomial size DNF formulas, and the well-studied class AC^0 of constant-depth polynomial-size AND/OR/NOT circuits.

Building on work of Beigel and Tarui [4], Green et al. [9] have given the following characterization of *ACC*:

Theorem 3. *For each $L \in ACC$ there is a depth-2 circuit which recognizes $L \cap \{0, 1\}^n$ and has the following structure: the top-level gate computes a midbit function of its inputs, and the bottom level consists of $2^{(\log n)^{O(1)}}$ AND gates each of fanin $(\log n)^{O(1)}$.*

Using this characterization we obtain the following hardness result for learning C_{mid}:

Theorem 4. *If C_{mid} can be PAC learned in polynomial (or even quasipolynomial) time, then ACC can be PAC learned in quasipolynomial time.*

Proof. Let $f : \{0, 1\}^n \to \{0, 1\}$ be the target *ACC* function. Let $q(n) = 2^{(\log n)^{O(1)}}$ be an upper bound on the number of AND gates on the bottom level of the Green et al. representation for f, and let $\ell(n) = (\log n)^{O(1)}$ be an upper bound on the fanin of each bottom level AND gate. Given an instance $x \in \{0, 1\}^n$ we generate a new instance $x' \in \{0, 1\}^m$ where $m = 2^{(\log n)^{O(1)}}$ by listing $q(n)$ copies of each AND of at most $\ell(n)$ variables from $x_1, \ldots, x_n$. Theorem 3 implies that there is an embedded midbit function f' on m bits such that $f(x) = f'(x')$ for all $x \in \{0, 1\}^n$. By assumption we can PAC learn this function f' in $2^{(\log m)^{O(1)}} = 2^{(\log n)^{O(1)}}$ time, so the theorem is proved. $\square$

We note that while our reduction only establishes quasipolynomial time learnability for *ACC* from learnability of C_{mid}, whereas the Blum reduction would establish polynomial time learnability of $C_{\log}$, the class *ACC* is likely to be much harder to learn than $C_{\log}$. While $C_{\log}$ can be PAC learned in $n^{\log n}$ time by doing an exhaustive search for the set of $\log n$ relevant variables, no learning algorithm for *ACC* is known which runs in subexponential time. In fact, no such algorithm is known even for the subclass of polynomial-size, depth 3 AND/OR/NOT circuits; to date the most expressive subclass of *ACC* which is known to be PAC learnable in subexponential time is the class of polynomial-size AND/OR/NOT circuits of depth 2, which has recently been shown by Klivans and Servedio [11] to be PAC learnable in time $2^{\tilde{O}(n^{1/3})}$.

4. Learning embedded midbit functions in $n^{O(\sqrt{n})}$ time

The results of Section 3 suggest that the class of embedded midbit functions may not be PAC learnable in quasipolynomial time. However, we will show that it is possible to learn this class substantially faster than a naive exponential time algorithm. In this section, we describe two different algorithms each of which PAC learns C_{mid} in time $n^{O(\sqrt{n})}$.

4.1. An algorithm based on learning linear threshold functions

Our first approach is a variant of an algorithm given by Blum et al. in [5, Section 5.2].

Definition 5. Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function and $p(x_1,\ldots,x_n)$ a real-valued polynomial. We say that $p(x)$ *sign-represents* $f(x)$ if for all $x \in \{0,1\}^n$, $p(x) \geqslant 0$ iff $f(x) = 1$.

Claim 1. *Let M_S be an embedded midbit function. Then there is a polynomial $p_S(x_1,\ldots,x_n)$ of degree $O(\sqrt{n})$ which sign-represents $M_S(x)$.*

Proof. Let f_s be the basis function for M_S. Since $k(s) = \Omega(\sqrt{s})$, the number of "flip" positions in the pattern of f_s where $f_s(i) \neq f_s(i+1)$ is $O(\sqrt{s})$. Since the pattern for f_s has $O(\sqrt{s})$ flips, there is some polynomial $P(X)$ of degree $O(\sqrt{s})$ which is nonnegative on precisely those $i \in \{0,1,\ldots,s\}$ which have $f_s(i) = 1$. This implies that $p_S(x_1,\ldots,x_n) = P(\sum_S x_i)$ sign-represents $M_S(x)$. Since the degree of p_S is $O(\sqrt{s})$ and $s \leqslant n$ the claim is proved. □

Consider the expanded feature space consisting of all monotone conjunctions of at most $O(\sqrt{n})$ variables. (Note that this feature space contains $\sum_{i=1}^{O(\sqrt{n})} n^i = n^{O(\sqrt{n})}$ features.) Claim 1 implies that $M_S(x)$ is equivalent to some linear threshold function over this space. Thus, we can use known polynomial time PAC learning algorithms for linear threshold functions [6] over this expanded feature space to learn embedded midbit functions in $n^{O(\sqrt{n})}$ time.

We note that one can show that the sign-representing polynomial $p_S(x_1,\ldots,x_n)$ described in Claim 1 can be taken without loss of generality to have integer coefficients of total magnitude $n^{O(\sqrt{n})}$. This implies that simple algorithms such as Winnow or Perceptron can be used to learn in $n^{O(\sqrt{n})}$ time (instead of the more sophisticated algorithm of [6] which is based on polynomial time linear programming). We also note that in [13] Minsky and Papert used a symmetrization technique to give a lower bound on the degree of any polynomial which sign-represents the parity function. The same technique can be used to show that the $O(\sqrt{n})$ degree bound of Claim 1 is optimal for embedded midbit functions.

4.2. An algorithm based on learning parities

We have seen that any embedded midbit function is equivalent to some linear threshold function over the feature space of all $O(\sqrt{n})$-size monotone conjunctions. We now show that any embedded midbit function is equivalent to some parity over this feature space as well.

Lemma 6. *Let $r, \ell \geqslant 0$. Then $\binom{r}{2^\ell}$ is even if and only if*

$$[r \bmod 2^{\ell+1}] \in \{0,1,\ldots,2^\ell - 1\}.$$

Proof. By induction on ℓ. The base case $\ell = 0$ is trivial; we suppose that the claim holds for $\ell = 0, \dots, i-1$ for some $i \geqslant 1$. For the induction step we use the fact [8, Exercise 5.61] that

$$\binom{r}{m} \equiv \binom{\lfloor r/p \rfloor}{\lfloor m/p \rfloor}\binom{[r \bmod p]}{[m \bmod p]} \pmod{p}$$

for all primes p and all $r, m \geqslant 0$. Taking $p = 2$ and $m = 2^i$, since $i \geqslant 1$ we have

$$\binom{r}{2^i} \equiv \binom{\lfloor r/2 \rfloor}{2^{i-1}}\binom{[r \bmod 2]}{0} \equiv \binom{\lfloor r/2 \rfloor}{2^{i-1}} \pmod{2}.$$

By the induction hypothesis we have that this is 0 if and only if $[\lfloor r/2 \rfloor \bmod 2^i] \in \{0, 1, \dots, 2^{i-1} - 1\}$, which holds if and only if $[r \bmod 2^{i+1}] \in \{0, 1, \dots, 2^i - 1\}$. □

Claim 2. *Let M_S be an embedded midbit function. Then $M_S(x)$ is equivalent to some parity of monotone conjunctions each of which contains at most $O(\sqrt{n})$ variables.*

Proof. Let $\oplus$ denote the parity function. We have

$$\begin{aligned} M_S(x) = 0 &\Longleftrightarrow \lfloor \log(s)/2 \rfloor \text{th bit of } \sum_S x_i \text{ is } 0 \\ &\Longleftrightarrow \left[\sum_S x_i \bmod 2^{\lfloor \log(s)/2 \rfloor + 1}\right] \in \{0, 1, \dots, 2^{\lfloor \log(s)/2 \rfloor} - 1\} \\ &\Longleftrightarrow \binom{\sum_S x_i}{2^{\lfloor \log(s)/2 \rfloor}} = \binom{\sum_S x_i}{k(s)} \text{ is even} \\ &\Longleftrightarrow \bigoplus_{A \subseteq S, |A| = k(s)} \left(\bigwedge_{i \in A} x_i\right) = 0. \end{aligned}$$

The third step is by Lemma 6 and the last step is because for any x exactly $\binom{\sum_S x_i}{k(s)}$ of the conjunctions $\{\bigwedge_{i \in A} x_i\}_{A \subseteq S, |A| = k(s)}$ take value 1. Since $k(s) = O(\sqrt{n})$ the claim is proved. □

As in the discussion following Claim 1, Claim 2 implies that we can use known PAC learning algorithms for parity [7,10] over an expanded feature space to learn embedded midbit functions in $n^{O(\sqrt{n})}$ time.

5. A polynomial time algorithm for learning embedded midbits under the uniform distribution

In [5] Blum et al. posed as an open problem the question of whether embedded symmetric concepts can be learned under the uniform distribution in polynomial time. In this section, we show that embedded midbit functions can be PAC learned under the uniform distribution in polynomial time. This is in strong contrast to the results of Section 3 which indicate that embedded midbit functions probably cannot be PAC learned (in even quasipolynomial time) under arbitrary probability distributions.

Throughout this section, we let $t(s)$ denote $\lfloor s/k(s) \rfloor$.

5.1. First approach: testing single variables

To learn M_S it is sufficient to identify the set $S \subseteq \{x_1, \dots, x_n\}$ of relevant variables. A natural first approach is to test the correlation of each individual variable with $M_S(x)$; clearly variables not in S will have zero correlation, and one might hope that variables in S will have nonzero correlation. However, this hope is incorrect as shown by Lemma 8 below.

For $1 \leqslant i \leqslant n$ define $p_i = \Pr[M_S(x) = 1 | x_i = 1] - \Pr[M_S(x) = 1]$. The following fact is easily verified:

Fact 3. *If $i \notin S$ then $p_i = 0$.*

Lemma 7. *If $i \in S$ then*

$$p_i = \frac{1}{2^s} \sum_{\ell=1}^{t(s)} (-1)^{\ell-1} \binom{s-1}{\ell k(s)-1}. \tag{1}$$

Proof. Since the distribution on examples is uniform over $\{0,1\}^n$, the probability that exactly ℓ of the s relevant variables are 1 is exactly $\binom{s}{\ell}/2^s$. Hence we have

$$p_i = \frac{1}{2^{s-1}} \sum_{\ell: f_s(\ell)=1} \binom{s-1}{\ell-1} - \frac{1}{2^s} \sum_{\ell: f_s(\ell)=1} \binom{s}{\ell}.$$

Using the identity

$$\binom{s}{\ell} = \binom{s-1}{\ell-1} + \binom{s-1}{\ell},$$

we find that

$$p_i = \frac{1}{2^s} \sum_{f_s(\ell)=1} \left(\binom{s-1}{\ell-1} - \binom{s-1}{\ell} \right).$$

Cancelling terms where possible we obtain (1). □

Lemma 8. *There are embedded midbit functions $M_S(x)$ with S a proper subset of $\{x_1, \ldots, x_n\}$ such that $p_i = 0$ for all $1 \leqslant i \leqslant n$.*

Proof. By Fact 3 for $i \notin S$ we have $p_i = 0$. Suppose that $t(s)$ is even and $t(s)k(s) - 1 = s - 1 - (k(s) - 1)$. Then the expression for p_i given in (1) is exactly 0 since the positive and negative binomial coefficients $\pm\binom{s-1}{\ell k(s)-1}$ and $\mp\binom{s-1}{(t(s)-\ell+1)k(s)-1}$ cancel each other out (e.g. take $s = 27$, $k(s) = 4$, $t(s) = 6$). □

Thus the correlation of individual variables with $M_S(x)$ need not provide information about membership in S. However, we will show that by testing correlations of *pairs* of variables with $M_S(x)$ we can efficiently determine whether or not a given variable belongs to S.

5.2. Second approach: testing pairs of variables

For $1 \leqslant i, j \leqslant n$, $i \neq j$ let $p_{i,j} = \Pr[M_S(x) = 1 | x_i = x_j = 1] - \Pr[M_S(x) = 1 | x_j = 1]$. Similar to Fact 3 we have

Fact 4. *If $i \notin S$ then $p_{i,j} = 0$.*

Lemma 9. *If $i \in S$ and $j \in S$ then*

$$p_{i,j} = \frac{1}{2^{s-1}} \sum_{\ell=1}^{t(s)} (-1)^{\ell-1} \binom{s-2}{\ell k(s)-2}. \tag{2}$$

Proof. We have

$$p_{i,j} = \frac{1}{2^{s-2}} \sum_{\ell: f_s(\ell)=1} \binom{s-2}{\ell-2} - \frac{1}{2^{s-1}} \sum_{\ell: f_s(\ell)=1} \binom{s-1}{\ell-1}.$$

Rearranging the sum as in Lemma 7 proves the lemma. □

It is easy to construct an example similar to that of Lemma 8 in which $p_{i,j} = 0$ even though $i \in S$ and $j \in S$. Thus, there are examples for which looking only at single variables fails, and there are examples for which looking only at pairs of variables fails. However, we show below that a strategy of looking *both* at single variables *and* at pairs

```
Input: variable $x_i \in \{x_1,\ldots,x_n\}$
Output: either "$x_i \in S$" or "$x_i \notin S$" correct with probability $1-\frac{\delta}{n}$
1. let $T$ be a sample of $m = O(n^2 \log\frac{n}{\delta})$ labeled examples $\langle x, M_S(x)\rangle$
2. let $\hat{p}_i$ be an empirical estimate of $p_i$ obtained from $T$
3. for all $j \in \{1,\ldots,n\} - \{i\}$
4.     let $\hat{p}_{i,j}$ be an empirical estimate of $p_{i,j}$ obtained from $T$
5. if $|\hat{p}_i| > \frac{1}{2000n}$ or $|\hat{p}_{i,j}| > \frac{1}{2000n}$ for some $j \in \{1,\ldots,n\} - \{i\}$
6.     then output "$i \in S$"
7.     else output "$i \in S$"
```

Fig. 1. An algorithm to determine whether x_i is relevant for $M_S(x)$.

of variables cannot fail. More precisely, our algorithm is based on the fact (Theorem 10 below) that quantities (1) and (2) cannot both be extremely close to 0:

Theorem 10. *Let k be even and $\sqrt{s}/2 < k \leqslant \sqrt{s}$. Let*

$$A = \frac{1}{2^s}\sum_{\ell}(-1)^{\ell-1}\binom{s-1}{\ell k-1} \quad \textit{and} \quad B = \frac{1}{2^{s-1}}\sum_{\ell}(-1)^{\ell-1}\binom{s-2}{\ell k-2}.$$

Then $\max\{|A|,|B|\} \geqslant 1/1000s$.

The proof of Theorem 10 is somewhat involved and is deferred to Section 5.3.

With Theorem 10 in hand we can prove our main positive learning result for C_{mid}.

Theorem 11. *The class of embedded midbit functions is learnable under the uniform distribution in polynomial time.*

Proof. Since there are fewer than n^3 midbit functions $M_S(x)$ which have $s \leqslant 3$ we can test each of these for consistency with a polynomial size random sample in polynomial time, and thus we can learn in polynomial time if $s \leqslant 3$. We, henceforth, assume that $s \geqslant 4$ and thus that $k(s) \geqslant 2$ is even.

We show that the algorithm in Fig. 1 correctly determines whether or not $x_i \in S$ with probability $1-(\delta/n)$. By running this algorithm n times on variables $x_1,\ldots,x_n$ we can identify the set S and thus learn M_S correctly with probability $1-\delta$.

Case 1: $x_i \notin S$. In this case by Facts 3 and 4 we have $p_i = p_{i,j} = 0$. Using standard Chernoff bounds it is easily verified that taking $m = \mathrm{O}(n^2\log(n/\delta))$, each of the n empirical estimates $\hat{p}_i, \hat{p}_{i,j}$ will satisfy $|\hat{p}_i| < 1/2000n$ and $|\hat{p}_{i,j}| < 1/2000n$ with probability $1-(\delta/n^2)$. Thus in this case the algorithm outputs "$x_i \notin S$" with probability at least $1-(\delta/n)$.

Case 2: $x_i \in S$. Since $s \geqslant 4$ there is some $x_j \neq x_i$ such that $x_j \in S$. Lemmas 7 and 9 and Theorem 10 imply that the true value of at least one of $|p_i|, |p_{i,j}|$ will be at least $1/1000s \geqslant 1/1000n$. As before, for m as above each of the n empirical estimates $\hat{p}_i, \hat{p}_{i,j}$ will differ from its true value by less than $1/2000n$ with probability $1-(\delta/n^2)$. Thus in this case the algorithm outputs "$x_i \in S$" with probability at least $1-(\delta/n)$. □

5.3. Proof of Theorem 10

The following lemma gives a useful expression for sums in the form of (1) and (2).

Lemma 12. *Let $r, j, k > 0$ with k even. Then*

$$\sum_{\ell}(-1)^{\ell-1}\binom{r}{\ell k-j} = \frac{-2}{k}\left(\sum_{\ell=1,3,5,\ldots,k-1}\left(2\cos\frac{\ell\pi}{2k}\right)^r\cos\left(\frac{(r+2j)\ell\pi}{2k}\right)\right). \tag{3}$$

Proof. We reexpress the left side as

$$\sum_{\ell}\binom{r}{\ell(2k)+(k-j)}-\sum_{\ell}\binom{r}{\ell(2k)-j}. \tag{4}$$

The following well-known identity (see e.g. [15,16]) is due to Ramus [14]:

$$\sum_{\ell}\binom{r}{\ell k-j}=\frac{1}{k}\sum_{\ell=1}^{k}\left(2\cos\frac{\ell\pi}{k}\right)^{r}\cos\left(\frac{(r+2j)\ell\pi}{k}\right).$$

Applying this identity to (4) we obtain

$$\frac{1}{2k}\left[\sum_{\ell=1}^{2k}\left(2\cos\frac{\ell\pi}{2k}\right)^{r}\cos\left(\frac{(r-2k+2j)\ell\pi}{2k}\right)-\sum_{\ell=1}^{2k}\left(2\cos\frac{\ell\pi}{2k}\right)^{r}\cos\left(\frac{(r+2j)\ell\pi}{2k}\right)\right].$$

Since even terms cancel out in the two sums above, we obtain

$$\frac{-1}{k}\left(\sum_{\ell=1,3,\ldots,2k-1}\left(2\cos\frac{\ell\pi}{2k}\right)^{r}\cos\left(\frac{(r+2j)\ell\pi}{2k}\right)\right). \tag{5}$$

Consider the term of this sum obtained when $\ell=2k-h$ for some odd value h

$$\begin{aligned}&\left(2\cos\frac{(2k-h)\pi}{2k}\right)^{r}\cos\left(\frac{(r+2j)(2k-h)\pi}{2k}\right)\\&\quad=(-1)^{r+(r+2j)}\left(2\cos\frac{-h\pi}{2k}\right)^{r}\cos\left(\frac{(r+2j)(-h)\pi}{2k}\right)\\&\quad=\left(2\cos\frac{h\pi}{2k}\right)^{r}\cos\left(\frac{(r+2j)h\pi}{2k}\right).\end{aligned}$$

This equals the term obtained when $\ell=h$. Since $k=2m$ is even we have that (5) equals the right side of (3). □

The following two technical lemmas will help us analyze the right-hand side of Eq. (3). No attempt has been made to optimize constants in the bounds.

Lemma 13. *Let r,k be such that $k\geqslant 4$ is even and $k^2-2\leqslant r<4k^2-1$. Then*
(i) *for $\ell=1,3,\ldots,k-3$ we have $0<(\cos((\ell+2)\pi/2k))^{r}<(\cos(\ell\pi/2k))^{r}/16$,*
(ii) $(\cos(\pi/2k))^{r}\geqslant 1/200$.

Proof. By considering the Taylor series of $\cos x$ one can show that $1-(x^2/2)\leqslant\cos x\leqslant 1-(x^2/3)$ for all $x\in[0,\pi/2]$.
Part (i): since $0<\ell\pi/2k<(\ell+2)\pi/2k<\pi/2$, we have

$$\cos\frac{(\ell+2)\pi}{2k}=\cos\frac{\ell\pi}{2k}\cos\frac{\pi}{k}-\sin\frac{\ell\pi}{2k}\sin\frac{\pi}{k}<\left(1-\frac{\pi^2}{3k^2}\right)\cos\frac{\ell\pi}{2k},$$

and hence

$$\begin{aligned}\left(\cos\frac{(\ell+2)\pi}{2k}\right)^{r}&\leqslant\left(1-\frac{\pi^2}{3k^2}\right)^{r}\left(\cos\frac{\ell\pi}{2k}\right)^{r}\\&\leqslant\left(1-\frac{\pi^2}{3k^2}\right)^{k^2-2}\left(\cos\frac{\ell\pi}{2k}\right)^{r}\\&\leqslant\frac{e^{-\pi^2/3}}{(1-\pi^2/(3k^2))^2}\cdot\left(\cos\frac{\ell\pi}{2k}\right)^{r}\\&\leqslant\frac{1}{16}\cdot\left(\cos\frac{\ell\pi}{2k}\right)^{r}.\end{aligned}$$

Here the third inequality uses $(1-(1/x))^{x}\leqslant e^{-1}$ and the fourth inequality uses $k\geqslant 4$.

Part (ii): we have

$$\left(\cos\frac{\pi}{2k}\right)^r > \left(\cos\frac{\pi}{2k}\right)^{4k^2} > \left(1-\frac{\pi^2}{8k^2}\right)^{4k^2}.$$

This is an increasing function of k so for $k \geqslant 4$ the value is at least $\left(1-(\pi^2/128)\right)^{64} \geqslant 1/200$. □

Lemma 14. *For all real x and all odd* $\ell \geqslant 3$, *we have* $|\cos(\ell x)| \leqslant \ell|\cos x|$.

Proof. Fix $\ell \geqslant 3$. Let $y = (\pi/2) - x$ so $\ell|\cos x| = \ell|\sin y|$ and

$$|\cos(\ell x)| = \left|\cos\frac{\ell\pi}{2}\cos(\ell y) - \sin\frac{\ell\pi}{2}\sin(\ell y)\right| = |\sin(\ell y)|$$

(note that we have used the fact that ℓ is odd). Thus, we must show that $|\sin(\ell y)| \leqslant \ell|\sin y|$. This is clearly true if $|\sin y| \geqslant 1/\ell$; otherwise we may suppose that $0 \leqslant y < \sin^{-1} 1/\ell$ (the other cases are entirely similar) so $0 \leqslant \ell y \leqslant \pi/2$. Now $\sin(\ell y) \leqslant \ell \sin y$ follows from the concavity of $\sin y$ on $[0, \pi/2]$ and the fact that the derivative of $\sin y$ is 1 at $y = 0$. □

Using these tools we can now prove Theorem 10.

Theorem 10. *Let k be even and* $\sqrt{s}/2 < k \leqslant \sqrt{s}$. *Let*

$$A = \frac{1}{2^s}\sum_\ell (-1)^{\ell-1}\binom{s-1}{\ell k-1} \quad \textit{and} \quad B = \frac{1}{2^{s-1}}\sum_\ell (-1)^{\ell-1}\binom{s-2}{\ell k-2}.$$

Then $\max\{|A|, |B|\} \geqslant 1/1000s$.

Proof. By Lemma 12 we have

$$A = \frac{-1}{k}\left(\sum_{\ell=1,3,\ldots,k-1}\left(\cos\frac{\ell\pi}{2k}\right)^{s-1}\cos\left(\frac{(s+1)\ell\pi}{2k}\right)\right), \tag{6}$$

and

$$B = \frac{-1}{k}\left(\sum_{\ell=1,3,\ldots,k-1}\left(\cos\frac{\ell\pi}{2k}\right)^{s-2}\cos\left(\frac{(s+2)\ell\pi}{2k}\right)\right). \tag{7}$$

First the easy case: if $k = 2$ then $4 \leqslant s \leqslant 15$ and $A = (-1/2)(\cos(\pi/4))^{s-1}\cos((s+1)\pi/4)$, $B = (-1/2)(\cos(\pi/4))^{s-2}\cos((s+2)\pi/4)$. Since either $|\cos((s+1)\pi/4)|$ or $|\cos((s+2)\pi/4)|$ must be $\sqrt{2}/2$ we have $\max\{|A|, |B|\} \geqslant 1/2^{(s/2+1}$ which is easily seen to be at least $1/1000s$ for $4 \leqslant s \leqslant 15$.

Now suppose $k \geqslant 4$. For $\ell = 3, \ldots, k-1$ we have

$$\begin{aligned}\left|\left(\cos\frac{\ell\pi}{2k}\right)^{s-1}\cos\left(\frac{(s+1)\ell\pi}{2k}\right)\right| &\leqslant \frac{(\cos(\pi/2k))^{s-1}}{4^{\ell-1}}\cdot\left|\cos\left(\frac{(s+1)\ell\pi}{2k}\right)\right| \\ &\leqslant \left|\frac{\ell}{4^{\ell-1}}\cdot\left(\cos\frac{\pi}{2k}\right)^{s-1}\cos\left(\frac{(s+1)\pi}{2k}\right)\right|,\end{aligned}$$

where the first inequality is by repeated application of part (i) of Lemma 13 and the second is by Lemma 14. We thus have

$$\begin{aligned}&\sum_{\ell=3,5,\ldots,k-1}\left|\left(\cos\frac{\ell\pi}{2k}\right)^{s-1}\cos\left(\frac{(s+1)\ell\pi}{2k}\right)\right| \\ &\quad\leqslant \sum_{\ell=3,5,\ldots,k-1}\left|\frac{\ell}{4^{\ell-1}}\cdot\left(\cos\frac{\pi}{2k}\right)^{s-1}\cos\left(\frac{(s+1)\pi}{2k}\right)\right|\end{aligned}$$

$$
\begin{aligned}
&< \left|\left(\cos\frac{\pi}{2k}\right)^{s-1}\cos\left(\frac{(s+1)\pi}{2k}\right)\right|\cdot\sum_{\ell=3}^{\infty}\frac{\ell}{4^{\ell-1}}\\
&= \frac{5}{18}\cdot\left|\left(\cos\frac{\pi}{2k}\right)^{s-1}\cos\left(\frac{(s+1)\pi}{2k}\right)\right|.
\end{aligned}
$$

Thus the $\ell = 1$ term in sum (6) dominates the sum and we have

$$
\begin{aligned}
|A| &\geqslant \frac{13}{18}\cdot\frac{1}{k}\left|\left(\cos\frac{\pi}{2k}\right)^{s-1}\cos\left(\frac{(s+1)\pi}{2k}\right)\right|\\
&\geqslant \frac{13}{3600k}\cdot\left|\cos\left(\frac{(s+1)\pi}{2k}\right)\right|
\end{aligned}
$$

by part (ii) of Lemma 13. An identical analysis for B shows that

$$
|B| \geqslant \frac{13}{3600k}\cdot\left|\cos\left(\frac{(s+2)\pi}{2k}\right)\right|
$$

as well.

We now observe that

$$
\max\left\{\left|\cos\frac{(s+1)\pi}{2k}\right|,\left|\cos\frac{(s+2)\pi}{2k}\right|\right\}\geqslant\cos\left(\frac{\pi}{2}-\frac{\pi}{4k}\right)=\sin\frac{\pi}{4k}.
$$

Using Taylor series this is easily seen to be at least $\pi/8k$. Hence we have

$$
\max\{|A|,|B|\}\geqslant\frac{13}{3600k}\cdot\frac{\pi}{8k}>\frac{1}{1000k^2}\geqslant\frac{1}{1000s}
$$

and the theorem is proved. □

6. Conclusion

Several interesting open problems suggest themselves for future work. One goal is to improve on the $n^{\sqrt{n}}$ running time bound for PAC learning embedded midbit functions under arbitrary distributions. Another goal is to extend the uniform distribution algorithm for learning embedded midbit functions to an algorithm which can succeed under any product distribution. Finally, a more ambitious question is whether the reduction of Section 3 can be exploited to provide nontrivial learning algorithms for *ACC*. More specifically, the reduction of Section 3 implicitly defines a specific nonuniform distribution which is such that if embedded midbit functions can be learned under this distribution in quasipolynomial time, then any function in *ACC* can be learned in quasipolynomial time under the uniform distribution. Does this approach offer any new insights for designing uniform distribution learning algorithms for *ACC*?

References

[1] E. Allender, U. Hertrampf, Depth reduction for circuits of unbounded fan-in, Inform. Comput. 112 (2) (1994) 217–238.
[2] D. Barrington, Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1, J. Comput. System Sci. 38 (1) (1989) 150–164.
[3] D. Barrington, D. Therien, Finite monoids and the fine structure of NC^1, J. Assoc. Comput. Math. 35 (4) (1988) 941–952.
[4] R. Beigel, J. Tarui, On *ACC*, Comput. Complexity 4 (1994) 350–366.
[5] A. Blum, P. Chalasani, J. Jackson, On learning embedded symmetric concepts, in: Proc. Sixth Annu. Conf. on Computational Learning Theory, 1993, pp. 337–346.
[6] A. Blumer, A. Ehrenfeucht, D. Haussler, M. Warmuth, Learnability and the Vapnik–Chervonenkis dimension, J. Assoc. Comput. Math. 36 (4) (1989) 929–965.
[7] P. Fischer, H.U. Simon, On learning ring-sum expansions, SIAM J. Comput. 21 (1) (1992) 181–192.
[8] R.L. Graham, D.E. Knuth, O. Patashnik, Concrete Mathematics, Addison-Wesley, Reading, MA, 1994.
[9] F. Green, J. Kobler, K. Regan, T. Schwentick, J. Toran, The power of the middle bit of a #P function, J. Comput. System Sci. 50 (3) (1998) 456–467.
[10] D. Helmbold, R. Sloan, M. Warmuth, Learning integer lattices, SIAM J. Comput. 21 (2) (1992) 240–266.
[11] A. Klivans, R. Servedio, Learning DNF in time $2^{\tilde{O}(n^{1/3})}$, in: Proc. Thirty-Third Annu. Symp. on Theory of Computing, 2001, pp. 258–265.

[12] P. McKenzie, D. Therien, Automata theory meets circuit complexity, in: Proc. Internat. Colloq. on Automata, Languages and Programming, 1989, pp. 589–602.
[13] M. Minsky, S. Papert, Perceptrons: An Introduction to Computational Geometry, MIT Press, Cambridge, MA, 1968.
[14] C. Ramus, Solution générale d'un problème d'analyse combinatoire, J. Reine Angew. Math. 11 (1834) 353–355.
[15] J. Riordan, An Introduction to Combinatorial Analysis, Wiley, New York, 1958.
[16] J. Riordan, Combinatorial Identities, Wiley, New York, 1968.
[17] L. Valiant, A theory of the learnable, Comm. ACM 27 (11) (1984) 1134–1142.
[18] A. Yao, Separating the polynomial time hierarchy by oracles, in: Proc. Twenty-Sixth Annu. Symp. on Foundations of Computer Science, 1985, pp. 1–10.
[19] A. Yao, On *ACC* and threshold circuits, in: Proc. Thirty-First Annu. Symp. on Foundations of Computer Science, 1990, pp. 619–627.