

Memory-Sample Lower Bounds for Learning Parity with Noise

Sumegha Garg ✉

Department of Computer Science, Harvard University, Cambridge, MA, USA

Pravesh K. Kothari ✉

Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA

Pengda Liu ✉

Department of Computer Science, Stanford University, CA, USA

Ran Raz ✉

Department of Computer Science, Princeton University, NJ, USA

Abstract

In this work, we show, for the well-studied problem of learning parity under noise, where a learner tries to learn $x = (x_1, \dots, x_n) \in \{0, 1\}^n$ from a stream of random linear equations over \mathbb{F}_2 that are correct with probability $\frac{1}{2} + \varepsilon$ and flipped with probability $\frac{1}{2} - \varepsilon$ ($0 < \varepsilon < \frac{1}{2}$), that any learning algorithm requires either a memory of size $\Omega(n^2/\varepsilon)$ or an exponential number of samples.

In fact, we study memory-sample lower bounds for a large class of learning problems, as characterized by [8], when the samples are noisy. A matrix $M : A \times X \rightarrow \{-1, 1\}$ corresponds to the following learning problem with error parameter ε : an unknown element $x \in X$ is chosen uniformly at random. A learner tries to learn x from a stream of samples, $(a_1, b_1), (a_2, b_2) \dots$, where for every i , $a_i \in A$ is chosen uniformly at random and $b_i = M(a_i, x)$ with probability $1/2 + \varepsilon$ and $b_i = -M(a_i, x)$ with probability $1/2 - \varepsilon$ ($0 < \varepsilon < \frac{1}{2}$). Assume that k, ℓ, r are such that any submatrix of M of at least $2^{-k} \cdot |A|$ rows and at least $2^{-\ell} \cdot |X|$ columns, has a bias of at most 2^{-r} . We show that any learning algorithm for the learning problem corresponding to M , with error parameter ε , requires either a memory of size at least $\Omega\left(\frac{k \cdot \ell}{\varepsilon}\right)$, or at least $2^{\Omega(r)}$ samples. The result holds even if the learner has an exponentially small success probability (of $2^{-\Omega(r)}$). In particular, this shows that for a large class of learning problems, same as those in [8], any learning algorithm requires either a memory of size at least $\Omega\left(\frac{(\log |X|) \cdot (\log |A|)}{\varepsilon}\right)$ or an exponential number of noisy samples.

Our proof is based on adapting the arguments in [21, 8] to the noisy case.

2012 ACM Subject Classification Theory of computation \rightarrow Machine learning theory

Keywords and phrases memory-sample tradeoffs, learning parity under noise, space lower bound, branching program

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2021.60

Category RANDOM

Funding *Sumegha Garg*: Research supported by Michael O. Rabin Postdoctoral Fellowship.


Pravesh K. Kothari: Research supported by NSF CAREER Award No. 2047933.

Ran Raz: Research supported by the Simons Collaboration on Algorithms and Geometry, by a Simons Investigator Award and by the National Science Foundation grants No. CCF-1714779, CCF-2007462.


Acknowledgements We would like to thank Avishay Tal and Greg Valiant for the helpful discussions.

1 Introduction

In this work, we study the number of samples needed for learning under noise and memory constraints. The study of the resources needed for learning, under memory constraints was initiated by Shamir [22] and Steinhart, Valiant and Wager [24], and has been studied

 © Sumegha Garg, Pravesh K. Kothari, Pengda Liu, and Ran Raz; licensed under Creative Commons License CC-BY 4.0
Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021).

Editors: Mary Wootters and Laura Sanità; Article No. 60; pp. 60:1–60:19

 Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

in the streaming setting. In addition to being a natural question in learning theory and complexity theory, lower bounds in this model also have direct applications to bounded storage cryptography [20, 26, 16, 25, 11, 13, 6, 12]. [24] conjectured that any algorithm for learning parities of size n (that is, learning $x \in \{0, 1\}^n$ from a stream of random linear equations in \mathbb{F}_2) requires either a memory of size $\Omega(n^2)$ or an exponential number of samples. This conjecture was proven in [20] and in follow up works, this was generalized to learning sparse parities in [16] and more general learning problems in [21, 17, 19, 8, 2, 5, 18, 23, 9, 4, 10].

In this work, we extend this line of work to *noisy* Boolean function learning problems. In particular, we consider the well-studied problem of learning parity under noise (LPN). In this problem, a learner wants to learn $x \in \{0, 1\}^n$ from independent and uniformly random linear equations in \mathbb{F}_2 where the right hand sides are obtained by independently flipping the evaluation of an unknown parity function with probability $\frac{1}{2} - \varepsilon$. Learning Parity with Noise (LPN) is a central problem in Learning and Coding Theory (often referred to as decoding random linear codes) and has been extensively studied. Even without memory constraints, coming up with algorithms for the problem has proven to be challenging and the current state-of-the-art for solving the problem is still the celebrated work of Blum, Kalai and Wasserman [3] that runs in time $2^{O(n/\log_2(n))}$. Over time, the hardness of LPN (and its generalization to non-binary finite fields) has been used as a starting point in several hardness results [14, 7] and constructing cryptographic primitives [1]. On the other hand, lower-bounds for the problem are known only in restricted models such as Statistical Query Learning¹ [15].

Learning under noise is at least as hard as learning without noise and thus, memory-sample lower bounds for parity learning [20] holds for learning parity under noise too. It is natural to ask – can we get better space lower bounds for learning parities under noise? In this work, we are able to strengthen the memory lower bound to $\Omega(n^2/\varepsilon)$ for parity learning with noise.

Our results actually extend to a broad class of learning problems – under noise. As in [21] and follow up works, we represent a learning problem using a matrix. Let X, A be two finite sets (where X represents the concept-class that we are trying to learn and A represents the set of possible samples). Let $M : A \times X \rightarrow \{-1, 1\}$ be a matrix. The matrix M represents the following learning problem with error parameter ε ($0 < \varepsilon < \frac{1}{2}$): An unknown element $x \in X$ was chosen uniformly at random. A learner tries to learn x from a stream of samples, $(a_1, b_1), (a_2, b_2) \dots$, where for every i , $a_i \in A$ is chosen uniformly at random and $b_i = M(a_i, x)$ with probability $\frac{1}{2} + \varepsilon$.

1.1 Our Results

We use extractor-based characterization of the matrix M to prove our lower bounds, as done in [8]. Our main result can be stated as follows (Corollary 19): Assume that k, ℓ, r are such that any submatrix of M of at least $2^{-k} \cdot |A|$ rows and at least $2^{-\ell} \cdot |X|$ columns, has a bias of at most 2^{-r} . Then, any learning algorithm for the learning problem corresponding to M with error parameter ε requires either a memory of size at least $\Omega(k \cdot \ell/\varepsilon)$, or at least $2^{\Omega(r)}$ samples. Thus, we get an extra factor of $\frac{1}{\varepsilon}$ in the space lower bound for all the bounds on learning problems that [8] imply, some of which are as follows (see [8] for details on why the corresponding matrices satisfy the extractor-based property):

¹ The SQ model does not seem to distinguish between noisy and noiseless variants of parity learning and yields the same lower bound in both cases.

1. **Parities with noise:** A learner tries to learn $x = (x_1, \dots, x_n) \in \{0, 1\}^n$, from (a stream of) random linear equations over \mathbb{F}_2 which are correct with probability $\frac{1}{2} + \varepsilon$ and flipped with probability $\frac{1}{2} - \varepsilon$. Any learning algorithm requires either a memory of size $\Omega(n^2/\varepsilon)$ or an exponential number of samples.
2. **Sparse parities with noise:** A learner tries to learn $x = (x_1, \dots, x_n) \in \{0, 1\}^n$ of sparsity ℓ , from (a stream of) random linear equations over \mathbb{F}_2 which are correct with probability $\frac{1}{2} + \varepsilon$ and flipped with probability $\frac{1}{2} - \varepsilon$. Any learning algorithm requires:
 - a. Assuming $\ell \leq n/2$: either a memory of size $\Omega(n \cdot \ell/\varepsilon)$ or $2^{\Omega(\ell)}$ samples.
 - b. Assuming $\ell \leq n^{0.9}$: either a memory of size $\Omega(n \cdot \ell^{0.99}/\varepsilon)$ or $\ell^{\Omega(\ell)}$ samples.
3. **Learning from noisy sparse linear equations:** A learner tries to learn $x = (x_1, \dots, x_n) \in \{0, 1\}^n$, from (a stream of) random sparse linear equations, of sparsity ℓ , over \mathbb{F}_2 , which are correct with probability $\frac{1}{2} + \varepsilon$ and flipped with probability $\frac{1}{2} - \varepsilon$. Any learning algorithm requires:
 - a. Assuming $\ell \leq n/2$: either a memory of size $\Omega(n \cdot \ell/\varepsilon)$ or $2^{\Omega(\ell)}$ samples.
 - b. Assuming $\ell \leq n^{0.9}$: either a memory of size $\Omega(n \cdot \ell^{0.99}/\varepsilon)$ or $\ell^{\Omega(\ell)}$ samples.
4. **Learning from noisy low-degree equations:** A learner tries to learn $x = (x_1, \dots, x_n) \in \{0, 1\}^n$, from (a stream of) random multilinear polynomial equations of degree at most d , over \mathbb{F}_2 , which are correct with probability $\frac{1}{2} + \varepsilon$ and flipped with probability $\frac{1}{2} - \varepsilon$. We prove that if $d \leq 0.99 \cdot n$, any learning algorithm requires either a memory of size $\Omega\left(\binom{n}{\leq d} \frac{n}{d \cdot \varepsilon}\right)$ or $2^{\Omega(n/d)}$ samples (where $\binom{n}{\leq d} = \binom{n}{0} + \binom{n}{1} + \dots + \binom{n}{d}$).
5. **Low-degree polynomials with noise:** A learner tries to learn an n' -variate multilinear polynomial p of degree at most d over \mathbb{F}_2 , from (a stream of) random evaluations of p over $\mathbb{F}_2^{n'}$, which are correct with probability $\frac{1}{2} + \varepsilon$ and flipped with probability $\frac{1}{2} - \varepsilon$. We prove that if $d \leq 0.99 \cdot n'$, any learning algorithm requires either a memory of size $\Omega\left(\binom{n'}{\leq d} \cdot \frac{n'}{d \cdot \varepsilon}\right)$ or $2^{\Omega(n'/d)}$ samples.

1.2 Techniques

Our proof follows the proof of [21, 8] very closely and builds on that proof. We extend the extractor-based result of [8] to the noisy case and a straightforward adaptation to its proof gives the stronger lower bound for the noisy case (which reflects on the strength of the current techniques). The main contribution of this paper is not a technical one but establishing stronger space lower bounds for a well-studied problem of learning parity with noise, using the current techniques.

1.3 Discussion and Open Problem

Let's look at a space upper bound for the problem of learning parity with noise, that is, a learner tries to learn $x \in \{0, 1\}^n$ from a stream of samples of the form (a, b) , where $a \in \{0, 1\}^n$ is chosen uniformly at random and $b = a \cdot x$ with probability $\frac{1}{2} + \varepsilon$ and $b = 1 - a \cdot x$ with probability $\frac{1}{2} - \varepsilon$ (here, $a \cdot x$ represents the inner product of a and x in \mathbb{F}_2 , that is, $a \cdot x = \sum_i a_i x_i \pmod{2}$).

Upper Bound

Consider the following algorithm A : Store the first $m = O(n/\varepsilon^2)$ samples. Check for every $x' \in \{0, 1\}^n$, if for at least $(\frac{1}{2} + \frac{\varepsilon}{2})$ fraction of the samples $(a_1, b_1), \dots, (a_m, b_m)$, $a_i \cdot x'$ agrees with b_i . Output the first x' that satisfies the check. In expectation, $a_i \cdot x$ would agree with b_i for $(\frac{1}{2} + \varepsilon)$ fraction of the samples, and otherwise for $x' \neq x$, in expectation, $a_i \cdot x'$ would

agree with b_i for half the samples. Therefore, for large enough m , using Chernoff bound and a union bound, with high probability $(1 - o(1))$ over the m samples, x' satisfies the check if and only if $x' = x$, and A outputs the correct answer under such an event. A uses $O(n/\varepsilon^2)$ samples and $O(n^2/\varepsilon^2)$ bits of space.

In this paper, we prove that any algorithm that learns parity with noise from a stream of samples (as defined above) requires $\Omega(n^2/\varepsilon)$ bits of space or exponential number of samples. Improving the lower bound to match the upper bound (or vice versa) is a fascinating open problem and we conjecture that the upper bound is tight. As each sample gives at most $O(\varepsilon^2)$ bits of information about x , we can at least show that a learning algorithm requires $O(n/\varepsilon^2)$ samples to learn x (which corresponds to using $O(n^2/\varepsilon^2)$ bits of space if each sample is stored).

► **Conjecture 1.** *Any learner that tries to learn $x \in \{0, 1\}^n$ from a stream of samples of the form (a, b) , where $a \in \{0, 1\}^n$ is chosen uniformly at random and $b = a \cdot x$ with probability $\frac{1}{2} + \varepsilon$ and $b = 1 - a \cdot x$ with probability $\frac{1}{2} - \varepsilon$, requires either $\Omega(n^2/\varepsilon^2)$ bits of memory or $2^{\Omega(n)}$ samples.*

The proof of the conjecture, if true, would lead to new technical insights (beyond extractor-based techniques) into proving time-space (or memory-sample) lower bounds for learning problems.

1.4 Outline of the Paper

In Section 2, we establish certain notations and definitions, which are borrowed from [21, 8]. We give a proof overview in Section 3 and prove the main theorem in Section 4.

2 Preliminaries

Denote by $\mathcal{U}_X : X \rightarrow \mathbb{R}^+$ the uniform distribution over X . Denote by \log the logarithm to base 2. For a random variable Z and an event E , we denote by \mathbb{P}_Z the distribution of the random variables Z , and we denote by $\mathbb{P}_{Z|E}$ the distribution of the random variable Z conditioned on the event E .

Viewing a Learning Problem, with error $\frac{1}{2} - \varepsilon$, as a Matrix

Let X, A be two finite sets of size larger than 1. Let $n = \log_2 |X|$ and $n' = \log_2 |A|$.

Let $M : A \times X \rightarrow \{-1, 1\}$ be a matrix. The matrix M corresponds to the following learning problem with error parameter ε ($0 < \varepsilon < \frac{1}{2}$). There is an unknown element $x \in X$ that was chosen uniformly at random. A learner tries to learn x from samples (a, b) , where $a \in A$ is chosen uniformly at random, and $b = M(a, x)$ with probability $\frac{1}{2} + \varepsilon$ and $b = -M(a, x)$ with probability $\frac{1}{2} - \varepsilon$. That is, the learning algorithm is given a stream of samples, $(a_1, b_1), (a_2, b_2) \dots$, where each a_t is uniformly distributed, and $b_t = M(a_t, x)$ with probability $\frac{1}{2} + \varepsilon$ and $b_t = -M(a_t, x)$ with probability $\frac{1}{2} - \varepsilon$.

Norms and Inner Products

Let $p \geq 1$. For a function $f : X \rightarrow \mathbb{R}$, denote by $\|f\|_p$ the ℓ_p norm of f , with respect to the uniform distribution over X , that is:

$$\|f\|_p = \left(\mathbf{E}_{x \in_R X} [|f(x)|^p] \right)^{1/p}.$$

For two functions $f, g : X \rightarrow \mathbb{R}$, define their inner product with respect to the uniform distribution over X as

$$\langle f, g \rangle = \mathbf{E}_{x \in_R X} [f(x) \cdot g(x)].$$

For a matrix $M : A \times X \rightarrow \mathbb{R}$ and a row $a \in A$, we denote by $M_a : X \rightarrow \mathbb{R}$ the function corresponding to the a -th row of M . Note that for a function $f : X \rightarrow \mathbb{R}$, we have $\langle M_a, f \rangle = \frac{(M \cdot f)_a}{|X|}$. Here, $M \cdot f$ represents the matrix multiplication of M with f .

L_2 -Extractors and L_∞ -Extractors

► **Definition 2** (L_2 -Extractor). *Let X, A be two finite sets. A matrix $M : A \times X \rightarrow \{-1, 1\}$ is a (k, ℓ) - L_2 -Extractor with error 2^{-r} , if for every non-negative $f : X \rightarrow \mathbb{R}$ with $\frac{\|f\|_2}{\|f\|_1} \leq 2^\ell$ there are at most $2^{-k} \cdot |A|$ rows a in A with*

$$\frac{|\langle M_a, f \rangle|}{\|f\|_1} \geq 2^{-r}.$$

Let Ω be a finite set. We denote a distribution over Ω as a function $f : \Omega \rightarrow \mathbb{R}^+$ such that $\sum_{x \in \Omega} f(x) = 1$. We say that a distribution $f : \Omega \rightarrow \mathbb{R}^+$ has min-entropy k if for all $x \in \Omega$, we have $f(x) \leq 2^{-k}$.

► **Definition 3** (L_∞ -Extractor). *Let X, A be two finite sets. A matrix $M : A \times X \rightarrow \{-1, 1\}$ is a $(k, \ell \sim r)$ - L_∞ -Extractor if for every distribution $p_x : X \rightarrow \mathbb{R}^+$ with min-entropy at least $(\log(|X|) - \ell)$ and every distribution $p_a : A \rightarrow \mathbb{R}^+$ with min-entropy at least $(\log(|A|) - k)$,*

$$\left| \sum_{a' \in A} \sum_{x' \in X} p_a(a') \cdot p_x(x') \cdot M(a', x') \right| \leq 2^{-r}.$$

Branching Program for a Learning Problem

In the following definition, we model the learner for the learning problem that corresponds to the matrix M , by a *branching program*, as done by previous papers starting with [20].

► **Definition 4. Branching Program for a Learning Problem:** *A branching program of length m and width d , for learning, is a directed (multi) graph with vertices arranged in $m + 1$ layers containing at most d vertices each. In the first layer, that we think of as layer 0, there is only one vertex, called the start vertex. A vertex of outdegree 0 is called a leaf. All vertices in the last layer are leaves (but there may be additional leaves). Every non-leaf vertex in the program has $2|A|$ outgoing edges, labeled by elements $(a, b) \in A \times \{-1, 1\}$, with exactly one edge labeled by each such (a, b) , and all these edges going into vertices in the next layer. Each leaf v in the program is labeled by an element $\tilde{x}(v) \in X$, that we think of as the output of the program on that leaf.*

Computation-Path: *The samples $(a_1, b_1), \dots, (a_m, b_m) \in A \times \{-1, 1\}$ that are given as input, define a computation-path in the branching program, by starting from the start vertex and following at step t the edge labeled by (a_t, b_t) , until reaching a leaf. The program outputs the label $\tilde{x}(v)$ of the leaf v reached by the computation-path.*

Success Probability: *The success probability of the program is the probability that $\tilde{x} = x$, where \tilde{x} is the element that the program outputs, and the probability is over $x, a_1, \dots, a_m, b_1, \dots, b_m$ (where x is uniformly distributed over X and a_1, \dots, a_m are uniformly distributed over A , and for every t , $b_t = M(a_t, x)$ with probability $\frac{1}{2} + \varepsilon$ and $-M(a_t, x)$ with probability $\frac{1}{2} - \varepsilon$).*

A learning algorithm, using m samples and a memory of s bits, can be modeled as a branching program² of length m and width $2^{O(s)}$. Thus, we will focus on proving width-length tradeoffs for any branching program that learns an extractor-based learning problem with noise, and such tradeoffs would translate into memory-sample tradeoffs for the learning algorithms.

3 Overview of the Proof

The proof adapts the extractor-based time-space lower bound of [8] to the *noisy* case, which in turn built on [21] that gave a general technique for proving memory-samples lower bounds. We recall the arguments in [21, 8] for convenience.

Assume that M is a (k', ℓ') - L_2 -extractor with error $2^{-r'}$, and let $r = \min\{k', \ell', r'\}$. Let B be a branching program for the noisy learning problem that corresponds to the matrix M . We want to prove that B has at least $2^{\Omega(r)}$ length or requires at least $2^{\Omega(\frac{k'\ell'}{\varepsilon})}$ width (that is, any learning algorithm solving the learning problem corresponding to the matrix M with error parameter ε , requires either $\Omega(\frac{k'\ell'}{\varepsilon})$ memory or exponential number of samples). Assume for a contradiction that B is of length $m = 2^{cr}$ and width $d = 2^{c\frac{k'\ell'}{\varepsilon}}$, where $c > 0$ is a small constant.

We define the *truncated-path*, \mathcal{T} , to be the same as the computation-path of B , except that it sometimes stops before reaching a leaf. Roughly speaking, \mathcal{T} stops before reaching a leaf if certain “bad” events occur. Nevertheless, we show that the probability that \mathcal{T} stops before reaching a leaf is negligible, so we can think of \mathcal{T} as almost identical to the computation-path.

For a vertex v of B , we denote by E_v the event that \mathcal{T} reaches the vertex v . We denote by $\Pr(v) = \Pr(E_v)$ the probability for E_v (where the probability is over $x, a_1, \dots, a_m, b_1, \dots, b_m$), and we denote by $\mathbb{P}_{x|v} = \mathbb{P}_{x|E_v}$ the distribution of the random variable x conditioned on the event E_v . Similarly, for an edge e of the branching program B , let E_e be the event that \mathcal{T} traverses the edge e . Denote, $\Pr(e) = \Pr(E_e)$, and $\mathbb{P}_{x|e} = \mathbb{P}_{x|E_e}$.

A vertex v of B is called *significant* if

$$\|\mathbb{P}_{x|v}\|_2 > 2^{\ell'} \cdot 2^{-n}.$$

Roughly speaking, this means that conditioning on the event that \mathcal{T} reaches the vertex v , a non-negligible amount of information is known about x . In order to guess x with a non-negligible success probability, \mathcal{T} must reach a significant vertex. Lemma 6 shows that the probability that \mathcal{T} reaches any significant vertex is negligible, and thus the main result follows.

To prove Lemma 6, we show that for every fixed significant vertex s , the probability that \mathcal{T} reaches s is at most $2^{-\Omega(k'\ell'/\varepsilon)}$ (which is smaller than one over the number of vertices in B). Hence, we can use a union bound to prove the lemma.

The proof that the probability that \mathcal{T} reaches s is extremely small is the main part of the proof. To that end, we use the following functions to measure the progress made by the branching program towards reaching s .

² The lower bound holds for randomized learning algorithms because a branching program is a non-uniform model of computation, and we can fix a *good* randomization for the computation without affecting the width.

Let L_i be the set of vertices v in layer- i of B , such that $\Pr(v) > 0$. Let Γ_i be the set of edges e from layer- $(i-1)$ of B to layer- i of B , such that $\Pr(e) > 0$. Let

$$\mathcal{Z}_i = \sum_{v \in L_i} \Pr(v) \cdot \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^{k'/2\varepsilon},$$

$$\mathcal{Z}'_i = \sum_{e \in \Gamma_i} \Pr(e) \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^{k'/2\varepsilon}.$$

We think of $\mathcal{Z}_i, \mathcal{Z}'_i$ as measuring the progress made by the branching program, towards reaching a state with distribution similar to $\mathbb{P}_{x|s}$.

We show that each \mathcal{Z}_i may only be negligibly larger than \mathcal{Z}_{i-1} . Hence, since it's easy to calculate that $\mathcal{Z}_0 = 2^{-\frac{2nk'}{2\varepsilon}}$, it follows that \mathcal{Z}_i is close to $2^{-\frac{2nk'}{2\varepsilon}}$, for every i . On the other hand, if s is in layer- i then \mathcal{Z}_i is at least $\Pr(s) \cdot \langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle^{\frac{k'}{2\varepsilon}}$. Thus, $\Pr(s) \cdot \langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle^{\frac{k'}{2\varepsilon}}$ cannot be much larger than $2^{-2n\frac{k'}{2\varepsilon}}$. Since s is significant, $\langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle^{\frac{k'}{2\varepsilon}} > 2^{(2\ell' - 2n)\frac{k'}{2\varepsilon}}$ and hence $\Pr(s)$ is at most $2^{-\Omega(\frac{k'\ell'}{\varepsilon})}$.

The proof that \mathcal{Z}_i may only be negligibly larger than \mathcal{Z}_{i-1} is done in two steps: Claim 17 shows by a simple convexity argument that $\mathcal{Z}_i \leq \mathcal{Z}'_i$. The hard part, that is done in Claim 15 and Claim 16, is to prove that \mathcal{Z}'_i may only be negligibly larger than \mathcal{Z}_{i-1} .

For this proof, we define for every vertex v , the set of edges $\Gamma_{out}(v)$ that are going out of v , such that $\Pr(e) > 0$. Claim 15 shows that for every vertex v ,

$$\sum_{e \in \Gamma_{out}(v)} \Pr(e) \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^{k'/2\varepsilon}$$

may only be negligibly higher than

$$\Pr(v) \cdot \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^{k'/2\varepsilon}.$$

For the proof of Claim 15, which is the hardest proof in the paper, we follow [21, 8] and consider the function $\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}$. We first show how to bound $\|\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}\|_2$. We then consider two cases: If $\|\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}\|_1$ is negligible, then $\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^{k'/2\varepsilon}$ is negligible and doesn't contribute much, and we show that for every $e \in \Gamma_{out}(v)$, $\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^{k'/2\varepsilon}$ is also negligible and doesn't contribute much. If $\|\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}\|_1$ is non-negligible, we use the bound on $\|\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}\|_2$ and the assumption that M is a (k', ℓ') - L_2 -extractor to show that for almost all edges $e \in \Gamma_{out}(v)$, we have that $\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^{k'/2\varepsilon}$ is very close to $\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^{k'/2\varepsilon}$. Only an exponentially small $(2^{-k'})$ fraction of edges are “bad” and give a significantly larger $\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^{k'/2\varepsilon}$. In the noiseless case, any “bad” edge can increase $\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle$ by a factor of 2 in the worst case, and hence [8] raised $\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle$ and $\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle$ to the power of k' , as it is the largest power for which the contribution of the “bad” edges is still small (as their fraction is $2^{-k'}$). But in the noisy case, any “bad” edge can increase $\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle$ by a factor of at most $(1 + 2\varepsilon)$ in the worst case, and thus, we can afford to raise $\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle$ and $\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle$ to the power of $k'/2\varepsilon$. *This is where our proof differs from that of [8].*

This outline oversimplifies many details. To make the argument work, we force \mathcal{T} to stop at significant vertices and whenever $\mathbb{P}_{x|v}(x)$ is large, that is, at significant values, as done in previous papers. And we force \mathcal{T} to stop before traversing some edges, that are so “bad” that their contribution to \mathcal{Z}'_i is huge and they cannot be ignored. We show that the total probability that \mathcal{T} stops before reaching a leaf is negligible.

4 Main Result

► **Theorem 5.** Let $\frac{1}{100} < c < \frac{\ln 2}{3}$. Fix γ to be such that $\frac{3c}{\ln 2} < \gamma^2 < 1$. Let X, A be two finite sets. Let $n = \log_2 |X|$. Let $M : A \times X \rightarrow \{-1, 1\}$ be a matrix which is a (k', ℓ') - L_2 -extractor with error $2^{-r'}$, for sufficiently large³ k', ℓ' and r' , where $\ell' \leq n$. Let

$$r := \min \left\{ \frac{r'}{2}, \frac{(1-\gamma)k'}{2}, \frac{(1-\gamma)\ell'}{2} - 1 \right\}. \quad (1)$$

Let B be a branching program, of length at most 2^r and width at most $2^{c \cdot k' \cdot \ell' / \varepsilon}$, for the learning problem that corresponds to the matrix M with error parameter ε . Then, the success probability of B is at most $O(2^{-r})$.

Proof. We recall the proof in [8, 21] and adapt it to the noisy case. Let

$$k := \frac{\gamma \ln 2}{2\varepsilon} k' \quad \text{and} \quad \ell := \gamma \ell' / 3. \quad (2)$$

Our proof differs from [8] starting with Claim 10, which allows us to set k to a larger value of $\frac{\gamma \ln 2}{2\varepsilon} k'$ instead of $\gamma(\ln 2)k'$ as set in [8]. Note that by the assumption that k', ℓ' and r' are sufficiently large, we get that k, ℓ and r are also sufficiently large. Since $\ell' \leq n$, we have $\ell + r \leq \frac{\gamma \ell'}{3} + \frac{(1-\gamma)\ell'}{2} < \frac{\ell'}{2} \leq \frac{n}{2}$. Thus,

$$r < n/2 - \ell. \quad (3)$$

Let B be a branching program of length $m = 2^r$ and width⁴ $d = 2^{c \cdot k' \cdot \ell' / \varepsilon}$ for the learning problem that corresponds to the matrix M with error parameter ε . We will show that the success probability of B is at most $O(2^{-r})$.

4.1 The Truncated-Path and Additional Definitions and Notation

We will define the **truncated-path**, \mathcal{T} , to be the same as the computation-path of B , except that it sometimes stops before reaching a leaf. Formally, we define \mathcal{T} , together with several other definitions and notations, by induction on the layers of the branching program B .

Assume that we already defined the truncated-path \mathcal{T} , until it reaches layer- i of B . For a vertex v in layer- i of B , let E_v be the event that \mathcal{T} reaches the vertex v . For simplicity, we denote by $\Pr(v) = \Pr(E_v)$ the probability for E_v (where the probability is over $x, a_1, \dots, a_m, b_1, \dots, b_m$), and we denote by $\mathbb{P}_{x|v} = \mathbb{P}_{x|E_v}$ the distribution of the random variable x conditioned on the event E_v .

There will be three cases in which the truncated-path \mathcal{T} stops on a non-leaf v :

1. If v is a, so called, significant vertex, where the ℓ_2 norm of $\mathbb{P}_{x|v}$ is non-negligible. (Intuitively, this means that conditioned on the event that \mathcal{T} reaches v , a non-negligible amount of information is known about x).
2. If $\mathbb{P}_{x|v}(x)$ is non-negligible. (Intuitively, this means that conditioned on the event that \mathcal{T} reaches v , the correct element x could have been guessed with a non-negligible probability).
3. If $(M \cdot \mathbb{P}_{x|v})(a_{i+1})$ is non-negligible. (Intuitively, this means that \mathcal{T} is about to traverse a “bad” edge, which is traversed with a non-negligibly higher or lower probability than probability of traversal under uniform distribution on x).

Next, we describe these three cases more formally.

³ By “sufficiently large” we mean that k', ℓ', r' are larger than some constant that depends on γ .

⁴ width lower bound is vacuous for $\varepsilon < 2^{-r/2}$ as regardless of the width, $\Omega(n/\varepsilon^2) > 2^r$ samples are needed to learn.

Significant Vertices

We say that a vertex v in layer- i of B is **significant** if

$$\|\mathbb{P}_{x|v}\|_2 > 2^\ell \cdot 2^{-n}.$$

Significant Values

Even if v is not significant, $\mathbb{P}_{x|v}$ may have relatively large values. For a vertex v in layer- i of B , denote by $\text{Sig}(v)$ the set of all $x' \in X$, such that,

$$\mathbb{P}_{x|v}(x') > 2^{2\ell+2r} \cdot 2^{-n}.$$

Bad Edges

For a vertex v in layer- i of B , denote by $\text{Bad}(v)$ the set of all $\alpha \in A$, such that,

$$|(M \cdot \mathbb{P}_{x|v})(\alpha)| \geq 2^{-r'}.$$

The Truncated-Path \mathcal{T}

We define \mathcal{T} by induction on the layers of the branching program B . Assume that we already defined \mathcal{T} until it reaches a vertex v in layer- i of B . The path \mathcal{T} stops on v if (at least) one of the following occurs:

1. v is significant.
2. $x \in \text{Sig}(v)$.
3. $a_{i+1} \in \text{Bad}(v)$.
4. v is a leaf.

Otherwise, \mathcal{T} proceeds by following the edge labeled by (a_{i+1}, b_{i+1}) (same as the computational-path).

4.2 Proof of Theorem 5

Since \mathcal{T} follows the computation-path of B , except that it sometimes stops before reaching a leaf, the success probability of B is bounded (from above) by the probability that \mathcal{T} stops before reaching a leaf, plus the probability that \mathcal{T} reaches a leaf v and $\tilde{x}(v) = x$.

The main lemma needed for the proof of Theorem 5 is Lemma 6 that shows that the probability that \mathcal{T} reaches a significant vertex is at most $O(2^{-r})$.

► **Lemma 6.** *The probability that \mathcal{T} reaches a significant vertex is at most $O(2^{-r})$.*

Lemma 6 is proved in Section 4.3. We will now show how the proof of Theorem 5 follows from that lemma.

Lemma 6 shows that the probability that \mathcal{T} stops on a non-leaf vertex, because of the first reason (i.e., that the vertex is significant), is small. The next two claims imply that the probabilities that \mathcal{T} stops on a non-leaf vertex, because of the second and third reasons, are also small. We defer the proofs to Appendix A (proved as in [8]).

▷ **Claim 7.** If v is a non-significant vertex of B then

$$\Pr_x[x \in \text{Sig}(v) \mid E_v] \leq 2^{-2r}.$$

60:10 Memory-Sample Lower Bounds for Learning Parity with Noise

▷ **Claim 8.** If v is a non-significant vertex of B then

$$\Pr_{a_{i+1}} [a_{i+1} \in \text{Bad}(v)] \leq 2^{-2r}.$$

We can now use Lemma 6, Claim 7 and Claim 8 to prove that the probability that \mathcal{T} stops before reaching a leaf is at most $O(2^{-r})$. Lemma 6 shows that the probability that \mathcal{T} reaches a significant vertex and hence stops because of the first reason, is at most $O(2^{-r})$. Assuming that \mathcal{T} doesn't reach any significant vertex (in which case it would have stopped because of the first reason), Claim 7 shows that in each step, the probability that \mathcal{T} stops because of the second reason, is at most 2^{-2r} . Taking a union bound over the $m = 2^r$ steps, the total probability that \mathcal{T} stops because of the second reason, is at most 2^{-r} . In the same way, assuming that \mathcal{T} doesn't reach any significant vertex (in which case it would have stopped because of the first reason), Claim 8 shows that in each step, the probability that \mathcal{T} stops because of the third reason, is at most 2^{-2r} . Again, taking a union bound over the 2^r steps, the total probability that \mathcal{T} stops because of the third reason, is at most 2^{-r} . Thus, the total probability that \mathcal{T} stops (for any reason) before reaching a leaf is at most $O(2^{-r})$.

Recall that if \mathcal{T} doesn't stop before reaching a leaf, it just follows the computation-path of B . Recall also that by Lemma 6, the probability that \mathcal{T} reaches a significant leaf is at most $O(2^{-r})$. Thus, to bound (from above) the success probability of B by $O(2^{-r})$, it remains to bound the probability that \mathcal{T} reaches a non-significant leaf v and $\tilde{x}(v) = x$. Claim 9 shows that for any non-significant leaf v , conditioned on the event that \mathcal{T} reaches v , the probability for $\tilde{x}(v) = x$ is at most 2^{-r} , which completes the proof of Theorem 5.

▷ **Claim 9.** If v is a non-significant leaf of B then

$$\Pr[\tilde{x}(v) = x \mid E_v] \leq 2^{-r}.$$

Refer to Appendix A for the proof (proved as in [8]). This completes the proof of Theorem 5. ◀

4.3 Proof of Lemma 6

Proof. We need to prove that the probability that \mathcal{T} reaches any significant vertex is at most $O(2^{-r})$. Let s be a significant vertex of B . We will bound from above the probability that \mathcal{T} reaches s , and then use a union bound over all significant vertices of B . Interestingly, the upper bound on the width of B is used only in the union bound.

The Distributions $\mathbb{P}_{x|v}$ and $\mathbb{P}_{x|e}$

Recall that for a vertex v of B , we denote by E_v the event that \mathcal{T} reaches the vertex v . For simplicity, we denote by $\Pr(v) = \Pr(E_v)$ the probability for E_v (where the probability is over $x, a_1, \dots, a_m, b_1, \dots, b_m$), and we denote by $\mathbb{P}_{x|v} = \mathbb{P}_{x|E_v}$ the distribution of the random variable x conditioned on the event E_v .

Similarly, for an edge e of the branching program B , let E_e be the event that \mathcal{T} traverses the edge e . Denote, $\Pr(e) = \Pr(E_e)$ (where the probability is over $x, a_1, \dots, a_m, b_1, \dots, b_m$), and $\mathbb{P}_{x|e} = \mathbb{P}_{x|E_e}$.

▷ **Claim 10.** For any edge $e = (v, u)$ of B , labeled by (a, b) , such that $\Pr(e) > 0$, for any $x' \in X$,

$$\mathbb{P}_{x|e}(x') = \begin{cases} 0 & \text{if } x' \in \text{Sig}(v) \\ \mathbb{P}_{x|v}(x')(1 + 2\varepsilon) \cdot c_e^{-1} & \text{if } x' \notin \text{Sig}(v) \text{ and } M(a, x') = b \\ \mathbb{P}_{x|v}(x')(1 - 2\varepsilon) \cdot c_e^{-1} & \text{if } x' \notin \text{Sig}(v) \text{ and } M(a, x') \neq b \end{cases}$$

where c_e is a normalization factor that satisfies,

$$c_e \geq 1 - 4 \cdot 2^{-2r}.$$

Proof. Let $e = (v, u)$ be an edge of B , labeled by (a, b) , and such that $\Pr(e) > 0$. Since $\Pr(e) > 0$, the vertex v is not significant (as otherwise \mathcal{T} always stops on v and hence $\Pr(e) = 0$). Also, since $\Pr(e) > 0$, we know that $a \notin \text{Bad}(v)$ (as otherwise \mathcal{T} never traverses e and hence $\Pr(e) = 0$).

If \mathcal{T} reaches v , it traverses the edge e if and only if: $x \notin \text{Sig}(v)$ (as otherwise \mathcal{T} stops on v) and $a_{i+1} = a$, $b_{i+1} = b$. Therefore, by Bayes' rule, for any $x' \in X$,

$$\mathbb{P}_{x|e}(x') = \begin{cases} 0 & \text{if } x' \in \text{Sig}(v) \\ \mathbb{P}_{x|v}(x')(1 + 2\varepsilon) \cdot c_e^{-1} & \text{if } x' \notin \text{Sig}(v) \text{ and } M(a, x') = b \\ \mathbb{P}_{x|v}(x')(1 - 2\varepsilon) \cdot c_e^{-1} & \text{if } x' \notin \text{Sig}(v) \text{ and } M(a, x') \neq b \end{cases}$$

where c_e is a normalization factor, given by

$$\begin{aligned} c_e &= \sum_{\{x' : x' \notin \text{Sig}(v) \wedge M(a, x') = b\}} \mathbb{P}_{x|v}(x')(1 + 2\varepsilon) \\ &\quad + \sum_{\{x' : x' \notin \text{Sig}(v) \wedge M(a, x') \neq b\}} \mathbb{P}_{x|v}(x')(1 - 2\varepsilon) \\ &= (1 + 2\varepsilon) \cdot \Pr_x[(x \notin \text{Sig}(v)) \wedge (M(a, x) = b) \mid E_v] \\ &\quad + (1 - 2\varepsilon) \cdot \Pr_x[(x \notin \text{Sig}(v)) \wedge (M(a, x) \neq b) \mid E_v]. \end{aligned}$$

Since v is not significant, by Claim 7,

$$\Pr_x[x \in \text{Sig}(v) \mid E_v] \leq 2^{-2r}.$$

Since $a \notin \text{Bad}(v)$,

$$\left| \Pr_x[M(a, x) = 1 \mid E_v] - \Pr_x[M(a, x) = -1 \mid E_v] \right| = |(M \cdot \mathbb{P}_{x|v})(a)| \leq 2^{-r'},$$

and hence for every $b' \in \{-1, 1\}$,

$$\Pr_x[M(a, x) = b' \mid E_v] \geq \frac{1}{2} - 2^{-r'}.$$

Hence, by the union bound,

$$c_e \geq (1 + 2\varepsilon) \cdot (\frac{1}{2} - 2^{-r'} - 2^{-2r}) + (1 - 2\varepsilon) \cdot (\frac{1}{2} - 2^{-r'} - 2^{-2r}) \geq 1 - 4 \cdot 2^{-2r}$$

(where the last inequality follows since $r \leq r'/2$, by Equation (1)). \triangleleft

Bounding the Norm of $\mathbb{P}_{x|s}$

We will show that $\|\mathbb{P}_{x|s}\|_2$ cannot be too large. Towards this, we will first prove that for every edge e of B that is traversed by \mathcal{T} with probability larger than zero, $\|\mathbb{P}_{x|e}\|_2$ cannot be too large. We defer the proofs of the following claims to Appendix A (proved as in [8]).

▷ **Claim 11.** For any edge e of B , such that $\Pr(e) > 0$,

$$\|\mathbb{P}_{x|e}\|_2 \leq 4 \cdot 2^\ell \cdot 2^{-n}.$$

▷ **Claim 12.**

$$\|\mathbb{P}_{x|s}\|_2 \leq 4 \cdot 2^\ell \cdot 2^{-n}.$$

60:12 Memory-Sample Lower Bounds for Learning Parity with Noise

Similarity to a Target Distribution

Recall that for two functions $f, g : X \rightarrow \mathbb{R}^+$, we defined

$$\langle f, g \rangle = \mathbf{E}_{z \in_R X} [f(z) \cdot g(z)].$$

We think of $\langle f, g \rangle$ as a measure for the similarity between a function f and a target function g . Typically f, g will be distributions.

▷ Claim 13.

$$\langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle > 2^{2\ell} \cdot 2^{-2n}.$$

Proof. Since s is significant,

$$\langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle = \|\mathbb{P}_{x|s}\|_2^2 > 2^{2\ell} \cdot 2^{-2n}. \quad \triangleleft$$

▷ Claim 14.

$$\langle \mathcal{U}_X, \mathbb{P}_{x|s} \rangle = 2^{-2n},$$

where \mathcal{U}_X is the uniform distribution over X .

Proof. Since $\mathbb{P}_{x|s}$ is a distribution,

$$\langle \mathcal{U}_X, \mathbb{P}_{x|s} \rangle = 2^{-2n} \cdot \sum_{z \in X} \mathbb{P}_{x|s}(z) = 2^{-2n}. \quad \triangleleft$$

Measuring the Progress

For $i \in \{0, \dots, m\}$, let L_i be the set of vertices v in layer- i of B , such that $\Pr(v) > 0$. For $i \in \{1, \dots, m\}$, let Γ_i be the set of edges e from layer- $(i-1)$ of B to layer- i of B , such that $\Pr(e) > 0$. Recall that $k = \frac{\gamma \ln 2}{2\varepsilon} k'$ (Equation (2)).

For $i \in \{0, \dots, m\}$, let

$$\mathcal{Z}_i = \sum_{v \in L_i} \Pr(v) \cdot \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k.$$

For $i \in \{1, \dots, m\}$, let

$$\mathcal{Z}'_i = \sum_{e \in \Gamma_i} \Pr(e) \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k.$$

We think of $\mathcal{Z}_i, \mathcal{Z}'_i$ as measuring the progress made by the branching program, towards reaching a state with distribution similar to $\mathbb{P}_{x|s}$.

For a vertex v of B , let $\Gamma_{out}(v)$ be the set of all edges e of B , that are going out of v , such that $\Pr(e) > 0$. Note that

$$\sum_{e \in \Gamma_{out}(v)} \Pr(e) \leq \Pr(v).$$

(We don't always have an equality here, since sometimes \mathcal{T} stops on v).

The next four claims show that the progress made by the branching program is slow.

▷ **Claim 15.** For every vertex v of B , such that $\Pr(v) > 0$,

$$\sum_{e \in \Gamma_{out}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k \leq \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k \cdot (1 + 2^{-r})^k + (2^{-2n+2})^k.$$

Proof. If v is significant or v is a leaf, then \mathcal{T} always stops on v and hence $\Gamma_{out}(v)$ is empty and thus the left hand side is equal to zero and the right hand side is positive, so the claim follows trivially. Thus, we can assume that v is not significant and is not a leaf.

Define $P : X \rightarrow \mathbb{R}^+$ as follows. For any $x' \in X$,

$$P(x') = \begin{cases} 0 & \text{if } x' \in \text{Sig}(v) \\ \mathbb{P}_{x|v}(x') & \text{if } x' \notin \text{Sig}(v) \end{cases}$$

Note that by the definition of $\text{Sig}(v)$, for any $x' \in X$,

$$P(x') \leq 2^{2\ell+2r} \cdot 2^{-n}. \quad (4)$$

Define $f : X \rightarrow \mathbb{R}^+$ as follows. For any $x' \in X$,

$$f(x') = P(x') \cdot \mathbb{P}_{x|s}(x').$$

By Claim 12 and Equation (4),

$$\|f\|_2 \leq 2^{2\ell+2r} \cdot 2^{-n} \cdot \|\mathbb{P}_{x|s}\|_2 \leq 2^{2\ell+2r} \cdot 2^{-n} \cdot 4 \cdot 2^\ell \cdot 2^{-n} = 2^{3\ell+2r+2} \cdot 2^{-2n}. \quad (5)$$

By Claim 10, for any edge $e \in \Gamma_{out}(v)$, labeled by (a, b) , for any $x' \in X$,

$$\mathbb{P}_{x|e}(x') = \begin{cases} 0 & \text{if } x' \in \text{Sig}(v) \\ \mathbb{P}_{x|v}(x')(1 + 2\varepsilon) \cdot c_e^{-1} & \text{if } x' \notin \text{Sig}(v) \text{ and } M(a, x') = b \\ \mathbb{P}_{x|v}(x')(1 - 2\varepsilon) \cdot c_e^{-1} & \text{if } x' \notin \text{Sig}(v) \text{ and } M(a, x') \neq b \end{cases}$$

where c_e is a normalization factor that satisfies,

$$c_e \geq 1 - 4 \cdot 2^{-2r}.$$

Therefore, for any edge $e \in \Gamma_{out}(v)$, labeled by (a, b) , for any $x' \in X$,

$$\mathbb{P}_{x|e}(x') \cdot \mathbb{P}_{x|s}(x') = f(x') \cdot (1 + 2\varepsilon \cdot b \cdot M(a, x')) \cdot c_e^{-1}$$

and hence, we have

$$\begin{aligned} \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle &= \mathbf{E}_{x' \in \mathcal{R}X} [\mathbb{P}_{x|e}(x') \cdot \mathbb{P}_{x|s}(x')] = \mathbf{E}_{x' \in \mathcal{R}X} [f(x') \cdot (1 + 2\varepsilon \cdot b \cdot M(a, x')) \cdot c_e^{-1}] \\ &= (\|f\|_1 + 2\varepsilon \cdot b \cdot \langle M_a, f \rangle) \cdot (c_e)^{-1} \\ &< (\|f\|_1 + 2\varepsilon |\langle M_a, f \rangle|) \cdot (1 + 2^{-2r+3}) \end{aligned} \quad (6)$$

(where the last inequality holds by the bound that we have on c_e , because we assume that k', ℓ', r' and thus r are sufficiently large).

We will now consider two cases:

Case I: $\|f\|_1 < 2^{-2n}$. In this case, we bound $|\langle M_a, f \rangle| \leq \|f\|_1$ (since f is non-negative and the entries of M are in $\{-1, 1\}$) and $(1 + 2^{-2r+3}) < 2$ (since we assume that k', ℓ', r' and thus r are sufficiently large) and obtain for any edge $e \in \Gamma_{out}(v)$,

$$\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle < 4 \cdot 2^{-2n}.$$

Since $\sum_{e \in \Gamma_{out}(v)} \frac{\Pr(e)}{\Pr(v)} \leq 1$, Claim 15 follows, as the left hand side of the claim is smaller than the second term on the right hand side.

60:14 Memory-Sample Lower Bounds for Learning Parity with Noise

Case II: $\|f\|_1 \geq 2^{-2n}$. For every $a \in A$, define

$$t(a) = \frac{|\langle M_a, f \rangle|}{\|f\|_1}.$$

By Equation (6),

$$\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k < \|f\|_1^k \cdot (1 + 2\varepsilon \cdot t(a))^k \cdot (1 + 2^{-2r+3})^k. \quad (7)$$

Note that by the definitions of P and f ,

$$\|f\|_1 = \mathbf{E}_{x' \in_{RA} X} [f(x')] = \langle P, \mathbb{P}_{x|s} \rangle \leq \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle.$$

Note also that for every $a \in A$, there is at most one edge $e_{(a,1)} \in \Gamma_{out}(v)$, labeled by $(a, 1)$, and at most one edge $e_{(a,-1)} \in \Gamma_{out}(v)$, labeled by $(a, -1)$, and we have

$$\frac{\Pr(e_{(a,1)})}{\Pr(v)} + \frac{\Pr(e_{(a,-1)})}{\Pr(v)} \leq \frac{1}{|A|},$$

since $\frac{1}{|A|}$ is the probability that the next sample read by the program is a . Thus, summing over all $e \in \Gamma_{out}(v)$, by Equation (7),

$$\sum_{e \in \Gamma_{out}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k < \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k \cdot \mathbf{E}_{a \in_{RA} A} \left[(1 + 2\varepsilon \cdot t(a))^k \right] \cdot (1 + 2^{-2r+3})^k. \quad (8)$$

It remains to bound

$$\mathbf{E}_{a \in_{RA} A} \left[(1 + 2\varepsilon \cdot t(a))^k \right], \quad (9)$$

using the properties of the matrix M and the bounds on the ℓ_2 versus ℓ_1 norms of f .

By Equation (5), the assumption that $\|f\|_1 \geq 2^{-2n}$, Equation (1) and Equation (2), we get

$$\frac{\|f\|_2}{\|f\|_1} \leq 2^{3\ell+2r+2} \leq 2^{\ell'}.$$

Since M is a (k', ℓ') - L_2 -extractor with error $2^{-r'}$, there are at most $2^{-k'} \cdot |A|$ rows $a \in A$ with $t(a) = \frac{|\langle M_a, f \rangle|}{\|f\|_1} \geq 2^{-r'}$. We bound the expectation in Equation (9), by splitting the expectation into two sums

$$\mathbf{E}_{a \in_{RA} A} \left[(1 + 2\varepsilon \cdot t(a))^k \right] = \frac{1}{|A|} \cdot \sum_{a : t(a) \leq 2^{-r'}} (1 + 2\varepsilon \cdot t(a))^k + \frac{1}{|A|} \cdot \sum_{a : t(a) > 2^{-r'}} (1 + 2\varepsilon \cdot t(a))^k. \quad (10)$$

We bound the first sum in Equation (10) by $(1 + 2\varepsilon \cdot 2^{-r'})^k$. As for the second sum in Equation (10), we know that it is a sum of at most $2^{-k'} \cdot |A|$ elements, and since for every $a \in A$, we have $t(a) \leq 1$, we have

$$\frac{1}{|A|} \cdot \sum_{a : t(a) > 2^{-r'}} (1 + 2\varepsilon \cdot t(a))^k \leq 2^{-k'} \cdot (1 + 2\varepsilon)^k \leq 2^{-k'} e^{2\varepsilon k} \leq 2^{-2r}$$

(where in the last inequality we used Equations (1) and (2)). Overall, using Equation (1) again, we get

$$\mathbf{E}_{a \in_{RA} A} \left[(1 + 2\varepsilon \cdot t(a))^k \right] \leq (1 + 2\varepsilon \cdot 2^{-r'})^k + 2^{-2r} \leq (1 + 2^{-2r})^{k+1}. \quad (11)$$

Substituting Equation (11) into Equation (8), we obtain

$$\begin{aligned} \sum_{e \in \Gamma_{out}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k &< \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k \cdot (1 + 2^{-2r})^{k+1} \cdot (1 + 2^{-2r+3})^k \\ &< \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k \cdot (1 + 2^{-r})^k \end{aligned}$$

(where the last inequality uses the assumption that r is sufficiently large). This completes the proof of Claim 15. \triangleleft

The following three claims use Claim 15 to quantify the progress over the layers and we defer the proofs to Appendix A (proved as in [8]).

▷ **Claim 16.** For every $i \in \{1, \dots, m\}$,

$$\mathcal{Z}'_i \leq \mathcal{Z}_{i-1} \cdot (1 + 2^{-r})^k + (2^{-2n+2})^k.$$

▷ **Claim 17.** For every $i \in \{1, \dots, m\}$,

$$\mathcal{Z}_i \leq \mathcal{Z}'_i.$$

▷ **Claim 18.** For every $i \in \{1, \dots, m\}$,

$$\mathcal{Z}_i \leq 2^{4k+2r} \cdot 2^{-2k \cdot n}.$$

Proof of Lemma 6

We can now complete the proof of Lemma 6. Assume that s is in layer- i of B . By Claim 13,

$$\mathcal{Z}_i \geq \Pr(s) \cdot \langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle^k > \Pr(s) \cdot (2^{2\ell} \cdot 2^{-2n})^k = \Pr(s) \cdot 2^{2\ell \cdot k} \cdot 2^{-2k \cdot n}.$$

On the other hand, by Claim 18,

$$\mathcal{Z}_i \leq 2^{4k+2r} \cdot 2^{-2k \cdot n}.$$

Thus, using Equation (1) and Equation (2), we get

$$\Pr(s) \leq 2^{4k+2r} \cdot 2^{-2\ell \cdot k} \leq 2^{\frac{2k'}{\varepsilon}} \cdot 2^{-\frac{\gamma^2 \ln 2}{3\varepsilon} (k' \ell')}.$$

Recall that we assumed that the width of B is at most $2^{ck' \ell' / \varepsilon}$ for some constant $c < \ln 2/3$, and that the length of B is at most 2^r . Recall that we fixed γ such that $\gamma^2(\ln 2)/3 > c$. Taking a union bound over at most $2^r \cdot 2^{ck' \ell' / \varepsilon} \leq 2^{k'} \cdot 2^{ck' \ell' / \varepsilon}$ significant vertices of B , we conclude that the probability that \mathcal{T} reaches any significant vertex is at most $2^{-\Omega(k' \ell' / \varepsilon)}$. Since we assume that k' and ℓ' are sufficiently large, $2^{-\Omega(k' \ell' / \varepsilon)}$ is certainly at most $2^{-k'}$, which is at most 2^{-r} . \triangleleft

► **Corollary 19.** *Let X, A be two finite sets. Let $M : A \times X \rightarrow \{-1, 1\}$ be a matrix. Assume that $k, \ell, r \in \mathbb{N}$ are large enough and such that any submatrix of M of at least $2^{-k} \cdot |A|$ rows and at least $2^{-\ell} \cdot |X|$ columns, has a bias of at most 2^{-r} .*

Then, any learning algorithm for the learning problem corresponding to M with error parameter ε , requires either a memory of size at least $\Omega\left(\frac{k \cdot \ell}{\varepsilon}\right)$, or at least $2^{\Omega(r)}$ samples. The result holds even if the learner has an exponentially small success probability (of $2^{-\Omega(r)}$).

Corollary follows from the equivalence between L_2 -Extractors and L_∞ -Extractors (up to constant factors) observed in [8].

References

- 1 Michael Alekhnovich. More on average case vs approximation complexity. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 298–307. IEEE Computer Society, 2003. doi:10.1109/SFCS.2003.1238204.
- 2 Paul Beame, Shayan Oveis Gharan, and Xin Yang. Time-space tradeoffs for learning finite functions from random evaluations, with applications to polynomials. In *Conference On Learning Theory*, pages 843–856, 2018.
- 3 Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, 50(4):506–519, 2003. doi:10.1145/792538.792543.
- 4 Yuval Dagan, Gil Kur, and Ohad Shamir. Space lower bounds for linear prediction in the streaming model. In *Conference on Learning Theory*, pages 929–954. PMLR, 2019.
- 5 Yuval Dagan and Ohad Shamir. Detecting correlations with little memory and communication. In *Conference On Learning Theory*, pages 1145–1198, 2018.
- 6 Wei Dai, Stefano Tessaro, and Xihu Zhang. Super-linear time-memory trade-offs for symmetric encryption. Cryptology ePrint Archive, Report 2020/663, 2020. URL: <https://eprint.iacr.org/2020/663>.
- 7 Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. On agnostic learning of parities, monomials, and halfspaces. *SIAM J. Comput.*, 39(2):606–645, 2009. doi:10.1137/070684914.
- 8 Sumegha Garg, Ran Raz, and Avishay Tal. Extractor-based time-space lower bounds for learning. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 990–1002. ACM, 2018.
- 9 Sumegha Garg, Ran Raz, and Avishay Tal. Time-space lower bounds for two-pass learning. In *34th Computational Complexity Conference (CCC 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.
- 10 Uma Girish, Ran Raz, and Wei Zhan. Quantum logspace algorithm for powering matrices with bounded norm. *arXiv preprint arXiv:2006.04880*, 2020.
- 11 Jiaxin Guan and Mark Zhandary. Simple schemes in the bounded storage model. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 500–524. Springer, 2019.
- 12 Jiaxin Guan and Mark Zhandry. Disappearing cryptography in the bounded storage model. *IACR Cryptol. ePrint Arch.*, 2021:406, 2021.
- 13 Joseph Jaeger and Stefano Tessaro. Tight time-memory trade-offs for symmetric encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 467–497. Springer, 2019.
- 14 Adam Tauman Kalai, Adam R. Klivans, Yishay Mansour, and Rocco A. Servedio. Agnostically learning halfspaces. *SIAM J. Comput.*, 37(6):1777–1805, 2008. doi:10.1137/060649057.
- 15 Michael J. Kearns. Efficient noise-tolerant learning from statistical queries. *J. ACM*, 45(6):983–1006, 1998. doi:10.1145/293347.293351.
- 16 Gillat Kol, Ran Raz, and Avishay Tal. Time-space hardness of learning sparse parities. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1067–1080. ACM, 2017.
- 17 Dana Moshkovitz and Michal Moshkovitz. Mixing implies lower bounds for space bounded learning. In *Conference on Learning Theory*, pages 1516–1566. PMLR, 2017.
- 18 Dana Moshkovitz and Michal Moshkovitz. Entropy samplers and strong generic lower bounds for space bounded learning. In *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.
- 19 Michal Moshkovitz and Naftali Tishby. Mixing complexity and its applications to neural networks. *arXiv preprint arXiv:1703.00729*, 2017.
- 20 Ran Raz. Fast learning requires good memory: A time-space lower bound for parity learning. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 266–275. IEEE, 2016.

- 21 Ran Raz. A time-space lower bound for a large class of learning problems. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 732–742, 2017.
- 22 Ohad Shamir. Fundamental limits of online and distributed algorithms for statistical learning and estimation. *Advances in Neural Information Processing Systems*, 27:163–171, 2014.
- 23 Vatsal Sharan, Aaron Sidford, and Gregory Valiant. Memory-sample tradeoffs for linear regression with small error. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 890–901, 2019.
- 24 Jacob Steinhardt, Gregory Valiant, and Stefan Wager. Memory, communication, and statistical queries. In *Conference on Learning Theory*, pages 1490–1516. PMLR, 2016.
- 25 Stefano Tessaro and Aishwarya Thiruvengadam. Provable time-memory trade-offs: symmetric cryptography against memory-bounded adversaries. In *Theory of Cryptography Conference*, pages 3–32. Springer, 2018.
- 26 Gregory Valiant and Paul Valiant. Information theoretically secure databases. *arXiv preprint arXiv:1605.02646*, 2016.

A Omitted Proofs from Section 4

Proof of Claim 7. Since v is not significant,

$$\mathbf{E}_{x' \sim \mathbb{P}_{x|v}} [\mathbb{P}_{x|v}(x')] = \sum_{x' \in X} [\mathbb{P}_{x|v}(x')]^2 = 2^n \cdot \mathbf{E}_{x' \in \mathbb{R}X} [\mathbb{P}_{x|v}(x')] \leq 2^{2\ell} \cdot 2^{-n}.$$

Hence, by Markov's inequality,

$$\Pr_{x' \sim \mathbb{P}_{x|v}} [\mathbb{P}_{x|v}(x') > 2^{2r} \cdot 2^{2\ell} \cdot 2^{-n}] \leq 2^{-2r}.$$

Since conditioned on E_v , the distribution of x is $\mathbb{P}_{x|v}$, we obtain

$$\Pr_x [x \in \text{Sig}(v) \mid E_v] = \Pr_x [(\mathbb{P}_{x|v}(x) > 2^{2r} \cdot 2^{2\ell} \cdot 2^{-n}) \mid E_v] \leq 2^{-2r}. \quad \blacktriangleleft$$

Proof of Claim 8. Since v is not significant, $\|\mathbb{P}_{x|v}\|_2 \leq 2^\ell \cdot 2^{-n}$. Since $\mathbb{P}_{x|v}$ is a distribution, $\|\mathbb{P}_{x|v}\|_1 = 2^{-n}$. Thus,

$$\frac{\|\mathbb{P}_{x|v}\|_2}{\|\mathbb{P}_{x|v}\|_1} \leq 2^\ell \leq 2^{\ell'}.$$

Since M is a (k', ℓ') - L_2 -extractor with error $2^{-r'}$, there are at most $2^{-k'} \cdot |A|$ elements $\alpha \in A$ with

$$|\langle M_\alpha, \mathbb{P}_{x|v} \rangle| \geq 2^{-r'} \cdot \|\mathbb{P}_{x|v}\|_1 = 2^{-r'} \cdot 2^{-n}$$

The claim follows since a_{i+1} is uniformly distributed over A and since $k' \geq 2r$ (Equation (1)). \blacktriangleleft

Proof of Claim 9. Since v is not significant,

$$\mathbf{E}_{x' \in \mathbb{R}X} [\mathbb{P}_{x|v}(x')] \leq 2^{2\ell} \cdot 2^{-2n}.$$

Hence, for every $x' \in X$,

$$\Pr[x = x' \mid E_v] = \mathbb{P}_{x|v}(x') \leq 2^\ell \cdot 2^{-n/2} \leq 2^{-r}$$

since $r \leq n/2 - \ell$ (Equation (3)). In particular, $\Pr[\tilde{x}(v) = x \mid E_v] \leq 2^{-r}$. \blacktriangleleft

60:18 Memory-Sample Lower Bounds for Learning Parity with Noise

Proof of Claim 11. Let $e = (v, u)$ be an edge of B , labeled by (a, b) , and such that $\Pr(e) > 0$. Since $\Pr(e) > 0$, the vertex v is not significant (as otherwise \mathcal{T} always stops on v and hence $\Pr(e) = 0$). Thus,

$$\|\mathbb{P}_{x|v}\|_2 \leq 2^\ell \cdot 2^{-n}.$$

By Claim 10, for any $x' \in X$,

$$\mathbb{P}_{x|e}(x') = \begin{cases} 0 & \text{if } x' \in \text{Sig}(v) \\ \mathbb{P}_{x|v}(x')(1 + 2\varepsilon) \cdot c_e^{-1} & \text{if } x' \notin \text{Sig}(v) \text{ and } M(a, x') = b \\ \mathbb{P}_{x|v}(x')(1 - 2\varepsilon) \cdot c_e^{-1} & \text{if } x' \notin \text{Sig}(v) \text{ and } M(a, x') \neq b \end{cases}$$

where c_e is a normalization factor that satisfies,

$$c_e \geq 1 - 4 \cdot 2^{-2r} > \frac{1}{2}.$$

(where the last inequality holds because we assume that k', ℓ', r' and thus r are sufficiently large.) Thus, $\|\mathbb{P}_{x|e}\|_2 \leq c_e^{-1} \cdot (1 + 2\varepsilon) \|\mathbb{P}_{x|v}\|_2 \leq 4 \cdot 2^\ell \cdot 2^{-n}$. ◀

Proof of Claim 12. Let $\Gamma_{in}(s)$ be the set of all edges e of B , that are going into s , such that $\Pr(e) > 0$. Note that

$$\sum_{e \in \Gamma_{in}(s)} \Pr(e) = \Pr(s).$$

By the law of total probability, for every $x' \in X$,

$$\mathbb{P}_{x|s}(x') = \sum_{e \in \Gamma_{in}(s)} \frac{\Pr(e)}{\Pr(s)} \cdot \mathbb{P}_{x|e}(x'),$$

and hence by Jensen's inequality,

$$\mathbb{P}_{x|s}(x')^2 \leq \sum_{e \in \Gamma_{in}(s)} \frac{\Pr(e)}{\Pr(s)} \cdot \mathbb{P}_{x|e}(x')^2.$$

Summing over $x' \in X$, we obtain,

$$\|\mathbb{P}_{x|s}\|_2^2 \leq \sum_{e \in \Gamma_{in}(s)} \frac{\Pr(e)}{\Pr(s)} \cdot \|\mathbb{P}_{x|e}\|_2^2.$$

By Claim 11, for any $e \in \Gamma_{in}(s)$,

$$\|\mathbb{P}_{x|e}\|_2^2 \leq (4 \cdot 2^\ell \cdot 2^{-n})^2.$$

Hence, $\|\mathbb{P}_{x|s}\|_2^2 \leq (4 \cdot 2^\ell \cdot 2^{-n})^2$. ◀

Proof of Claim 16. By Claim 15,

$$\begin{aligned} \mathcal{Z}'_i &= \sum_{e \in \Gamma_i} \Pr(e) \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k = \sum_{v \in L_{i-1}} \Pr(v) \cdot \sum_{e \in \Gamma_{out}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k \\ &\leq \sum_{v \in L_{i-1}} \Pr(v) \cdot \left(\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k \cdot (1 + 2^{-r})^k + (2^{-2n+2})^k \right) \\ &= \mathcal{Z}_{i-1} \cdot (1 + 2^{-r})^k + \sum_{v \in L_{i-1}} \Pr(v) \cdot (2^{-2n+2})^k \\ &\leq \mathcal{Z}_{i-1} \cdot (1 + 2^{-r})^k + (2^{-2n+2})^k \end{aligned}$$

◀

Proof of Claim 17. For any $v \in L_i$, let $\Gamma_{in}(v)$ be the set of all edges $e \in \Gamma_i$, that are going into v . Note that

$$\sum_{e \in \Gamma_{in}(v)} \Pr(e) = \Pr(v).$$

By the law of total probability, for every $v \in L_i$ and every $x' \in X$,

$$\mathbb{P}_{x|v}(x') = \sum_{e \in \Gamma_{in}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \mathbb{P}_{x|e}(x'),$$

and hence

$$\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle = \sum_{e \in \Gamma_{in}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle.$$

Thus, by Jensen's inequality,

$$\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k \leq \sum_{e \in \Gamma_{in}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k.$$

Summing over all $v \in L_i$, we get

$$\begin{aligned} \mathcal{Z}_i &= \sum_{v \in L_i} \Pr(v) \cdot \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k \\ &\leq \sum_{v \in L_i} \Pr(v) \cdot \sum_{e \in \Gamma_{in}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k \\ &= \sum_{e \in \Gamma_i} \Pr(e) \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k \\ &= \mathcal{Z}'_i. \end{aligned} \quad \blacktriangleleft$$

Proof of Claim 18. By Claim 14, $\mathcal{Z}_0 = (2^{-2n})^k$. By Claim 16 and Claim 17, for every $i \in \{1, \dots, m\}$,

$$\mathcal{Z}_i \leq \mathcal{Z}_{i-1} \cdot (1 + 2^{-r})^k + (2^{-2n+2})^k.$$

Hence, for every $i \in \{1, \dots, m\}$,

$$\mathcal{Z}_i \leq (2^{-2n+2})^k \cdot (m+1) \cdot (1 + 2^{-r})^{km}.$$

Since $m = 2^r$,

$$\mathcal{Z}_i \leq 2^{-2k \cdot n} \cdot 2^{2k} \cdot (2^r + 1) \cdot e^k \leq 2^{-2k \cdot n} \cdot 2^{4k+2r}. \quad \blacktriangleleft$$