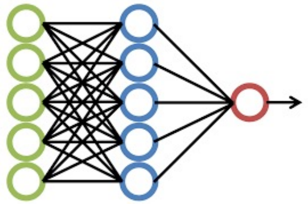
# Extractor-Based Approach to Memory-Sample Tradeoffs

Sumegha Garg

(Harvard)

Joint Works with
Pravesh Kothari (CMU),
Ran Raz (Princeton) and
Avishay Tal (UC Berkeley)

## Learning under Memory Constraints?



- Increasing scale of learning problems
- Many learning algorithms try to learn a concept/hypothesis by modeling it as a neural network. The algorithm keeps in the memory some neural network and updates the weights when new samples arrive. Memory used is the size of the network (low if network size is small).

# Brief Description for the Learning Model

A learner tries to learn $f: A \rightarrow \{0,1\}$, $f \in H$ (hypothesis class)

from samples of the form $(a_1, f(a_1)), (a_2, f(a_2)), \ldots$

that arrive one by one

## [Shamir'14], [SVW'15]

Can one prove unconditional strong lower bounds on the number of samples needed for learning under memory constraints?

***(when samples are viewed one by one)***

## Outline of the Talk

- The first example of such memory-sample tradeoff: Parity learning [Raz'16]
- Spark in the field: Subsequent results
- Extractor-based approach to lower bounds [G-Raz-Tal'18]
- Shallow dive into the proofs
- Implications for refutation of CSPs [G-Kothari-Raz'20]

## Example: Parity Learning

## Parity Learning

$f \in_R \{0,1\}^n$ is unknown

A learner tries to learn $f = (f^1, f^2, \dots, f^n)$ from

$(a_1, b_1), (a_2, b_2), \dots, (a_m, b_m)$, where $\forall\, t$,

$a_t \in_R \{0,1\}^n$ and $b_t = < a_t, f > = \sum_i a_t^i \cdot f^i \ mod\ 2$ (inner product mod 2)

## Parity Learning

A learner tries to learn $f = (f^1, f^2, \dots, f^n)$ from

$(a_1, b_1), (a_2, b_2), \dots, (a_m, b_m)$, where $\forall\, t$,

$a_t \in_R \{0,1\}^n$ and $b_t = \sum_i a_t^i \cdot f^i \ mod\ 2$

In other words, learner gets random binary linear equations in $f^1, f^2, \dots, f^n$, one by one, and need to solve them

## Let's Try Learning (n=5)

$$f_1 + f_3 + f_4 = 1$$

## Let’s Try Learning (n=5)

$$f_2 + f_4 + f_5 = 0$$

## Let’s Try Learning (n=5)

$$f_1 + f_2 + f_4 = 1$$

## Let’s Try Learning (n=5)

$$f_2 + f_3 = 0$$

## Let’s Try Learning (n=5)

$$f_1 + f_2 + f_4 + f_5 = 0$$

## Let’s Try Learning (n=5)

$$f_1 + f_3 = 1$$

## Let’s Try Learning (n=5)

$$f_2 + f_3 + f_4 + f_5 = 1$$

## Let's Try Learning (n=5)

$$f_1 + f_2 + f_5 = 0$$

## Let’s Try Learning (n=5)

$f_1 + f_2 + f_5 = 0$

$f_1 + f_3 + f_4 = 1$

$f_2 + f_3 + f_4 + f_5 = 1$

$f = (0,1,1,0,1)$

$f_2 + f_4 + f_5 = 0$

$f_1 + f_2 + f_4 = 1$

$f_1 + f_3 = 1$

$f_1 + f_2 + f_4 + f_5 = 0$

$f_2 + f_3 = 0$

## Parity Learners

- Solve independent linear equations (Gaussian Elimination)

***$O(n)$ samples and $O(n^2)$ memory***

- Try all possibilities of $f$

***$O(n)$ memory but exponential number of samples***

## Raz's Breakthrough '16

Any algorithm for parity learning of size $n$ requires either memory of $n^2/25$ bits or exponential number of samples to learn

***Memory-Sample Tradeoff***

# Motivations from Other Fields

Bounded Storage Cryptography: [Raz'16, GZ'19, VV'16, KRT'16, TT'18, GZ'19, JT'19, DTZ'20, GZ'21]

Key's length: $n$ Encryption/Decryption time: $n$

***Unconditional security, if attacker's memory size*** $< \frac{n^2}{25}$

Complexity Theory: Time-Space Lower Bounds have been studied in many models [BJS'98, Ajt'99, BSSV'00, For'97, FLvMV'05, Wil'06,...]

## Subsequent Results

## Subsequent Generalizations

- [KRT‘17]: Generalization to learning sparse parities
- [Raz’17, MM’17, MT’17, MM’18, DKS’19, GKLR’21]: Generalization to larger class of learning problems

## [Garg-Raz-Tal'18]

Extractor-based approach to prove memory-sample tradeoffs for a large class of learning problems

Implied all the previous results + new lower bounds

Closely follows the proof technique of [Raz'17]

[Beame-Oveis-Gharan-Yang'18] independently proved related lower bounds

## Subsequent Generalizations

[Sharan-Sidford-Valiant'19]: Any algorithm performing linear regression over a stream of $d$-dimensional examples, uses a quadratic amount of memory or exhibits a slower rate of convergence than can be achieved without memory constraint

***First-order methods for learning may have provably slower rate of convergence***

# More Related Work

- [GRT’19] Learning under multiple passes over the stream
- [DS’18, AMN’18, DGKR’19, GKR’20] Distinguishing, testing under memory or communication constraints
- [MT’19, GLM’20] Towards characterization of memory-bounded learning
- [GRZ’20] Comments on quantum memory-bounded learning

## Extractor-Based Approach to Lower Bounds

[G-Raz-Tal'18]

# Learning Problem as a Matrix

$A, F$ : finite sets, $M: A \times F \to \{-1,1\}$ : a matrix

$F$ : concept class $= \{0,1\}^n$

$A$ : possible samples $= \{0,1\}^{n'}$

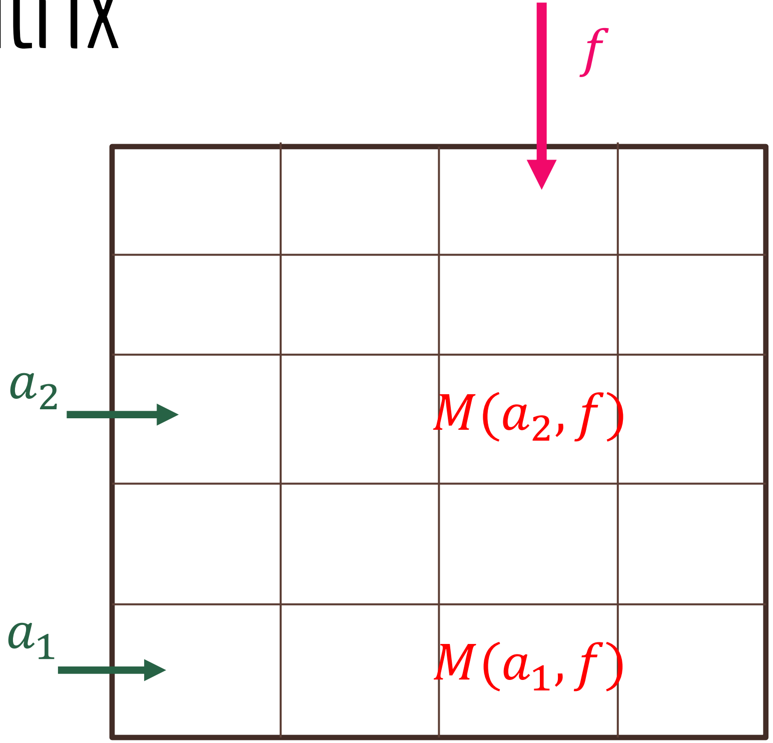
# Learning Problem as a Matrix

$M: A \times F \rightarrow \{-1,1\}$ : a matrix

$f \in_R F$ is unknown. A learner tries to learn $f$ from a stream

$(a_1, b_1), (a_2, b_2), \ldots, (a_m, b_m)$, where $\forall t$ :

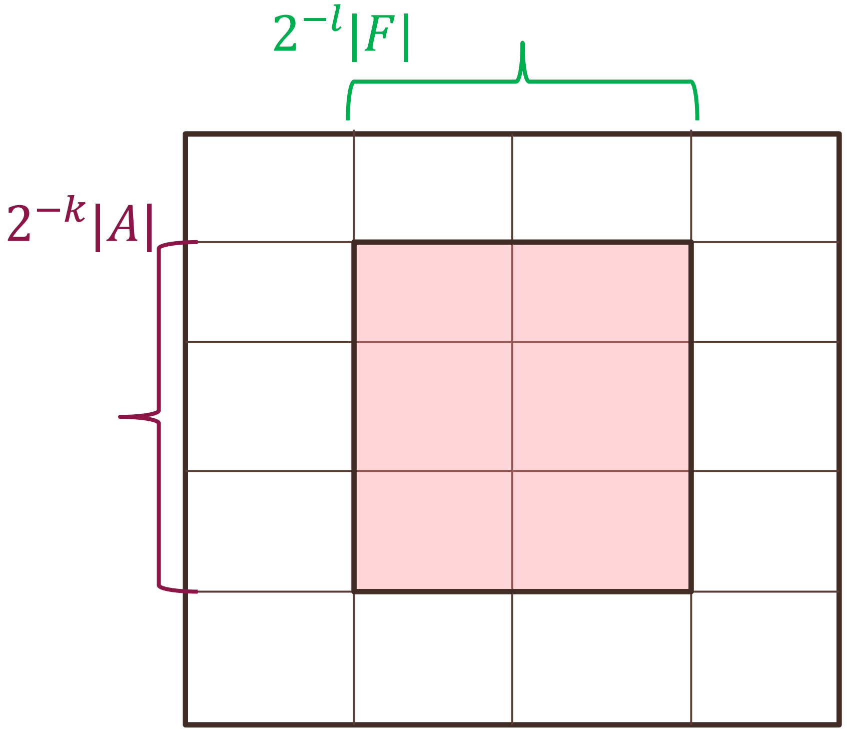
$a_t \in_R A$ and $b_t = M(a_t, f)$



## Main Theorem

Assume that any submatrix of $M$ of at least $2^{-k}|A|$ rows and at least $2^{-\ell}|F|$ columns, has a bias of at most $2^{-r}$. Then,

**Any learning algorithm requires either $\boldsymbol{\Omega(kl)}$ memory bits or $\boldsymbol{2^{\Omega(r)}}$ samples**



# Applications

Low-degree polynomials: A learner tries to learn an $n$-variate multilinear polynomial $p$ of degree at most $d$ over $\mathbb{F}_2$, from random evaluations of $p$ over $\mathbb{F}_2^n$

$\boldsymbol{\Omega(n^{d+1})}$ ***memory or*** $\boldsymbol{2^{\Omega(n)}}$ ***samples***

# Applications

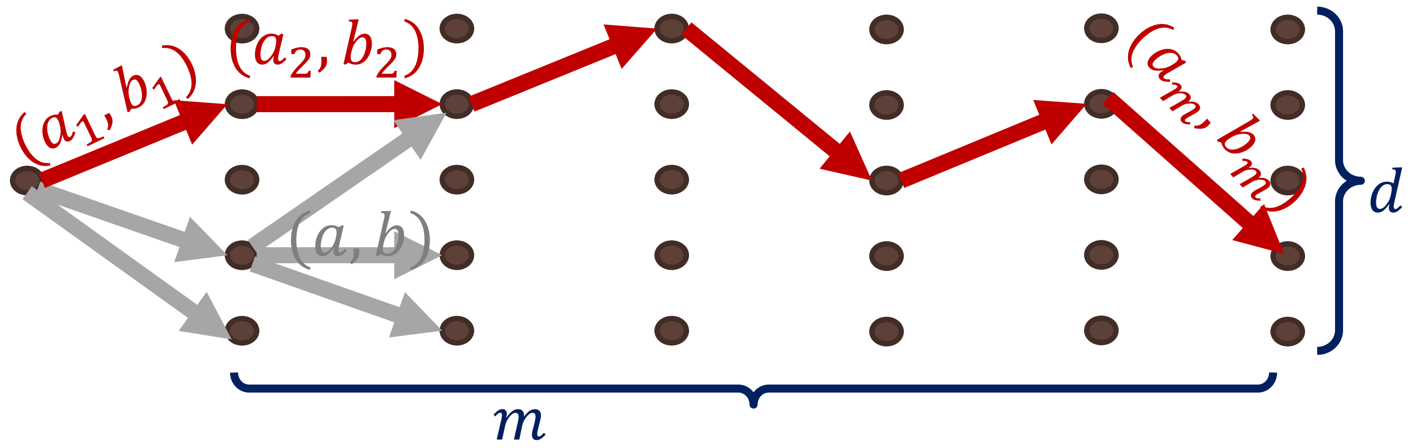
Learning from sparse equations: A learner tries to learn $f = (f_1, \ldots, f_n) \in \{0,1\}^n$ , from random sparse linear equations, of sparsity $l$, over $\mathbb{F}_2$ (each equation depends on $l$ variables)

***$\Omega(n \cdot l)$ memory or $2^{\Omega(l)}$ samples***

We will use this result for proving sample lower bounds for refuting CSPs under memory constraints

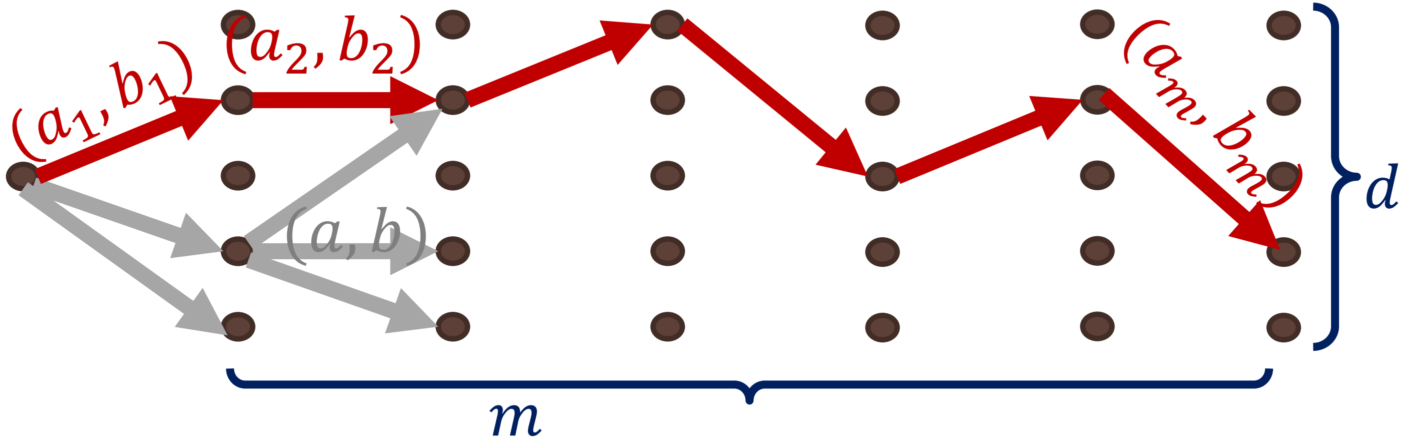
## Proof Overview

# Branching Program (length $m$, width $d$)



Each layer represents a time step. Each vertex represents a memory state of the learner ($d = 2^{memory}$). Each non-leaf vertex has $2^{n'+1}$ outgoing edges, one for each $(a, b) \in \{0,1\}^{n'} \times \{-1,1\}$

## Branching Program (length $m$, width $d$)



The samples $(a_1, b_1), \ldots, (a_m, b_m)$ define a computation-path. Each vertex $v$ in the last layer is labeled by $\hat{f}_v \in \{0,1\}^n$. The output is the label $\hat{f}_v$ of the vertex reached by the path

## Proof Overview

$\mathrm{P}_{f|v}$ = distribution of $f$ conditioned on the event that the computation-path reaches $v$

Significant vertices: $v$ s.t. $||\mathrm{P}_{f|v}||_2 \geq 2^{l} \cdot 2^{-n}$

$Pr(v)$ = probability that the path reaches $v$

We prove that if $v$ is significant, $Pr(v) \leq 2^{-\Omega(k \cdot l)}$

Hence, if there are less than $2^{\mathrm{O}(k \cdot l)}$ vertices, with high probability, we reach a non-significant vertex

# Proof Overview

Progress Function [Raz'17]: For layer $L_i$,

$$Z_i = \sum_{v \in L_i} Pr(v) \cdot \langle \mathrm{P}_{f|v}, \mathrm{P}_{f|s} \rangle^k$$

- $Z_0 = 2^{-2n \cdot k}$
- $Z_i$ is very slowly growing: $Z_0 \approx Z_m$
- If $s \in L_m$, then $Z_m \geq Pr(s) \cdot 2^{2l \cdot k} \cdot 2^{-2n \cdot k}$

Hence: If $s$ is significant, $Pr(s) \leq 2^{-\Omega(k \cdot l)}$

## Hardness of Refuting CSPs

[G-Kothari-Raz'20]

## Refuting CSPs under Streaming Model

- Fix a predicate $P: \{0,1\}^k \to \{0,1\}$
- With bounded memory, distinguish between CSPs (with predicate $P$) where the constraints are randomly generated vs ones where the constraints are generated using a satisfying assignment
- Constraints arrive one by one in a **stream**

## Refuting CSPs under Streaming Model

- Fix a predicate $P: \{0,1\}^k \rightarrow \{0,1\}$
- With bounded memory, distinguish between CSPs (with predicate $P$) where the constraints are randomly generated vs ones where the constraints are generated using a satisfying assignment
- Constraints arrive one

Refutation is at least as hard as distinguishing between Random CSPs and satisfiable ones

## Refuting CSPs under Streaming Model

- $x$ is chosen uniformly from $\{0,1\}^n$
- At each step $i$, $S_i \subseteq [n]$ is a randomly chosen (ordered) subset of size $k$.
- Distinguisher sees a stream of either $\left(S_i, P\left(x_{S_i}\right)\right)$ or $(S_i, b_i)$ where $b_i$ is chosen uniformly from $\{0,1\}$

$x_{S_i}$: projection of $x$ to coordinates of $S_i$

# Reduction from Learning from Sparse Equations

Any algorithm that distinguishes random $k$ –*XOR* (random CSPs with *XOR* predicate) on $n$ variables from

$k$ –*XOR* CSPs with a satisfying assignment (under the streaming model),

**requires either $\boldsymbol{\Omega(n \cdot k)}$ memory or $\boldsymbol{2^{\Omega(k)}}$ constraints**

# Reduction from Learning from Sparse Equations

Any algorithm that distinguishes random $k$ –*XOR* (random CSPs with *XOR* predicate) on $n$ variables from

$k$ –*XOR* CSPs with a

streaming model),

For $\boldsymbol{k \gg \log n}$, refutation is hard for algorithms using even super-linear memory, when the number of constraints is $< \boldsymbol{2^{\Omega(k)}}$

**requires either $\boldsymbol{\Omega(n \cdot k)}$ memory or $\boldsymbol{2^{\Omega(k)}}$ constraints**

## Better Sample Bounds for Sub-Linear Memory

- Stronger lower bounds on the number of constraints needed to refute
- Even for CSPs with $t$-resilient predicates $P$ [OW'14, AL'16, KMOW'17]
- $P: \{0,1\}^k \rightarrow \{0,1\}$ is $t$-resilient if $P$ has zero correlation with every parity function on at most $t-1$ bits

## Better Sample Bounds for Sub-Linear Memory

For every $0 < \epsilon < 1$, any algorithm that distinguishes

random CSPs (with a $t$-resilient predicate $P$) on $n$ variables

from satisfiable ones (when the constraints arrive in a streaming fashion),

**requires either $n^{\epsilon}$ memory or $\left(\frac{n}{t}\right)^{\Omega(t(1-\epsilon))}$ constraints**

Thank You!