Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 85 (2023)

Distributional PAC-Learning from Nisan's Natural Proofs

Ari Karchmer*
Boston University
arika@bu.edu

July 27, 2023

Abstract

Do strong natural proofs imply efficient learning algorithms? Carmosino et al. (2016) demonstrated that strong natural proofs of circuit lower bounds for Λ imply efficient algorithms for learning Λ-circuits, but only over *the uniform distribution*, with *membership queries*, and provided $\mathsf{AC}^0[p] \in \Lambda$. We consider whether this implication can be generalized to $\Lambda \not\supseteq \mathsf{AC}^0[p]$, and to learning algorithms that utilize only random examples and learn over arbitrary example distributions. We give results of both positive and negative flavor.

On the negative side, we observe that if, for *every* circuit class Λ, the implication from natural proofs for Λ to learning Λ-circuits in Valiant's PAC model holds, then there is a polynomial time solution to the $O(n^{1.5})$-uSVP (unique Shortest Vector Problem), and polynomial time quantum solutions to $\tilde{O}(n^{1.5})$-SVP and $\tilde{O}(n^{1.5})$-SIVP. This indicates that whether natural proofs for Λ imply efficient learning algorithms for Λ in Valiant's PAC model may depend on Λ.

On the positive side, our main result is that *specific* natural proofs arising from a type of communication complexity argument (e.g., Nisan (1993), for depth-2 majority circuits) imply PAC-learning algorithms in a new *distributional* variant of Valiant's model. Our distributional PAC model is stronger than the average-case prediction model of Blum et al (1993) and the heuristic PAC model of Nanashima (2021), and has several important properties which make it of independent interest, such as being *boosting-friendly.* The main applications of our result are new distributional PAC-learning algorithms for depth-2 majority circuits, polytopes and DNFs over natural target distributions, as well as the nonexistence of encoded-input weak PRFs that may be evaluated by depth-2 majority circuits.

*Part of this work was completed while the author was visiting the Simons Institute for the theory of computing.

ISSN 1433-8092

1 Introduction

Razborov and Rudich [RR97] introduced the concept of *natural proofs* of circuit lower bounds. Informally, a natural proof of a lower bound for a circuit class Λ encodes an efficient algorithm that can be used to distinguish between the truth tables of *simple* Boolean functions (those with "small" Λ-circuit complexity), and *random* Boolean functions. Razborov and Rudich essentially showed that natural proofs for a circuit class Λ rule out the existence of a cryptographic pseudorandom function (PRF) computable by Λ.

Carmosino et al. [CIKK16] strengthened the result of [RR97] by demonstrating that, provided $\mathsf{AC}^0[p] \subseteq \Lambda$, natural proofs of circuit lower bounds for Λ-circuits of size up to $u(n)$ imply algorithms for *learning* $\mathrm{poly}(n)$ size Λ-circuits with *membership queries* over the *uniform distribution*, in time exponential in $u^{-1}(\mathrm{poly}(n))$. As a corollary, [CIKK16] obtained a state-of-the-art *quasipolynomial* time learning algorithm for $\mathsf{AC}^0[p]$-circuits (with membership queries, over the uniform distribution), using the natural proofs for subexponential size $\mathsf{AC}^0[p]$-circuits, for any prime p, of Razborov and Smolensky [Raz87, Smo87].

Since the result of [CIKK16], whether or not there exists a fully general implication from natural proofs to learning algorithms in Valiant's *original* PAC model [Val84], even for $\Lambda \not\supseteq \mathsf{AC}^0[p]$, has remained open (see e.g. [GK23]). In Valiant's original model, learning algorithms are forced to utilize *random examples*, and learn over *unknown example distributions*.

> **Question 1.** Let Λ be any circuit class. Do natural circuit lower bounds for size $u(n)$ Λ-circuits imply $\exp(u^{-1}(\mathrm{poly}(n)))$ time learning algorithms for $\mathrm{poly}(n)$ size Λ-circuits in Valiant's PAC model?

Aside theoretical interest in complexity and learning theory, Question 1 is motivated by the prospect of implicitly extending the nonexistence of PRFs in low circuit classes (derived from [Raz87, Smo87, RR97, CIKK16]) to the nonexistence of *weak* PRFs. A weak PRF is a PRF that is only required to be secure if the adversary can inspect uniformly random points, as opposed to to chosen points (see Section 2.2 for a formal definition). Weak PRFs suffice for a variety of important cryptographic applications such as symmetric-key encryption (see e.g. [BCG+21] for more commentary). Therefore, understanding the minimum complexity needed to evaluate weak PRFs is of significant practical importance.

1.1 Our Contributions

We begin by observing that if the answer to Question 1 is essentially "*yes, for every* Λ," then this implies algorithmic breakthroughs for several important and well-studied computational problems. These breakthroughs include a classical polynomial time solution to the unique Shortest Vector Problem (uSVP), and quantum polynomial time algorithms for the Shortest Vector Problem (SVP) and Shortest Independent Vector Problem (SIVP) on lattices.[1] More specifically, we observe that majority-of-threshold circuits ($\mathsf{MAJ} \circ \mathsf{THR}$) cannot realize the implication from natural proofs to polynomial time PAC-learning in Valiant's model, assuming polynomial time hardness of each of those problems.

Theorem 1.1. *Suppose that a natural proof against* $\mathsf{MAJ} \circ \mathsf{THR}$*-circuits of size* $u(n)$ *implies that the class of* $\mathsf{MAJ} \circ \mathsf{THR}$*-circuits of size* $\mathrm{poly}(n)$ *is PAC-learnable in Valiant's model, in time* $\exp(u^{-1}(\mathrm{poly}(n)))$. *Then, there is a polynomial time classical solution to* $\tilde{O}(n^{1.5})$-uSVP*, and polynomial time quantum solutions to* $\tilde{O}(n^{1.5})$-SVP *and* $\tilde{O}(n^{1.5})$-SIVP.

To argue Theorem 1.1, we combine two observations. First, natural circuit lower bounds for $\exp(\Omega(n))$ size $\mathsf{MAJ} \circ \mathsf{THR}$-circuits were proved by Nisan [Nis93]. Second, Klivans and Sherstov [KS09] showed hardness of polynomial time PAC-learning in Valiant's model for $\mathsf{MAJ} \circ \mathsf{THR}$, assuming

[1]We will not try to discuss the huge literature on lattice problems (and lattice based cryptography). See Section 2 for a short description of uSVP, SVP, and SIVP, and refer to [Reg09a, Reg09b] for more information on complexity of lattice problems.

classical hardness of uSVP and quantum hardness SVP and SIVP. Taken together, we have both natural proofs against exponential-size $\mathsf{MAJ} \circ \mathsf{THR}$, and hardness of Valiant's PAC-learning for each. Therefore, we have a natural circuit class that resist an implication between natural lower bounds and Valiant's PAC-learning. A formal argument is presented in Section 4; to the best of our knowledge, the natural property underlying Nisan's circuit lower bounds has never been explicitly formalized, though it was acknowledged briefly by Raz [Raz00] and considered implicitly by Viola [Vio15].

Theorem 1.1 indicates a barrier to a general implication from a natural proof *for any* Λ to a PAC-learning algorithm for Λ in Valiant's model. Essentially, the natural proofs of [Nis93] confound the hardness result of [KS09]. In light of this, we shift our focus to the following more specific question.

Question 2. What learning algorithms *are* implied by Nisan's natural circuit lower bounds?

Answering Question 2 is important if we want to gain understanding of a potential general implication between natural proofs for any class Λ, and some kind of learning algorithms for Λ.

Towards an answer to Question 2, we will focus specifically on the possibility of learning algorithms that utilize only random examples, learn over unknown example distributions, and run in polynomial time. We briefly summarize our contributions towards an answer to Question 2, before digging into the specifics.

- In Section 1.1.1, we present a new learning model called *distributional* PAC learning, which relaxes Valiant's model, in order to try to sidestep Theorem 1.1. The new learning model is like Valiant's except it essentially removes the requirement of guarantees for the worst-case concept in the class. In Section 3, we illustrate that distributional PAC-learning is independently motivated for both technically and practically oriented reasons.

- In Section 4, we prove Theorem 1.1. To do so, we give the first (to the best of our knowledge) explicit formalization of Nisan's natural property for exponential size $\mathsf{MAJ} \circ \mathsf{THR}$-circuits.

- In Section 5, we prove our main theorem, which discovers a relationship between the computational complexity of distributional PAC-learning and the *communication complexity* of a simple communication game that is associated with a given concept class. This theorem serves as a "technical centerpiece" for extracting learning algorithms from Nisan's natural proofs.

- In Section 6 and 7, using the main theorem and Nisan's natural proofs, we obtain new algorithms as applications, including distributional PAC-learning algorithms for $\mathsf{MAJ} \circ \mathsf{THR}$-circuits, polytopes, and DNFs, and attacks on any weak PRFs evaluated by $\mathsf{MAJ} \circ \mathsf{THR}$-circuits, even when allowed an encoding of the inputs.

We note that, because the result of [CIKK16] only applies to circuit classes that contain $\mathsf{AC}^0[p]$, prior to this work there were no known learning algorithms following directly from Nisan's natural proofs, in any nontrivial learning model.

1.1.1 Distributional PAC-Learning

With the goal of obtaining learning algorithms from Nisan's natural proofs in mind, this paper introduces the distributional PAC-learning model (distPAC-learning). However, the distPAC-learning model is also independently motivated as a relaxation of Valiant's PAC-learning, which we discuss after defining the model next.

The starting point for the distPAC-learning model is the heuristic PAC-learning (heurPAC-learning) model of Nanashima [Nan21]. Following Nanashima, we define a Boolean concept class by a corresponding *evaluation rule* $\phi = \{\phi_n : \{0,1\}^* \times \{0,1\}^n \to \{-1,1\}\}_{n\in\mathbb{N}}$. The first input to the evaluation rule is a *binary representation* π_f of a concept f, and the second is an *input* to the concept x. The evaluation rule is defined so that for every $n \in \mathbb{N}$, $\phi_n(\pi_f, x) = f(x)$. An evaluation rule ϕ induces a Boolean concept class $\mathfrak{C} = \{\mathfrak{C}_n\}_{n\in\mathbb{N}}$ defined by

$$\mathfrak{C}_n = \{f(x) := \phi_n(\pi_f, x) : \pi_f \in \{0,1\}^*\}$$

We refer to $\mathfrak{C}$ as the ϕ-induced concept class.

For a function $s : \mathbb{N} \to \mathbb{N}$, we say that the ϕ-induced concept class $\mathfrak{C}$ is *$s(n)$-represented* if, for every $n \in \mathbb{N}$, under the evaluation rule ϕ_n, every $f \in \mathfrak{C}_n$ has a binary representation of length at most $s(n)$. Considering evaluation rules helps for formalizing learning using a distribution over a concept class. We let μ denote a *target distribution* over concepts $f \in \mathfrak{C}_n$, or equivalently, over binary representations $\pi_f \in \{0,1\}^{s(n)}$.

As for what access to the concept algorithms in the distPAC-learning model are allowed, we continue following the heurPAC-learning model (indeed, Valiant's too), and grant only random examples sampled from an *unknown* example distribution ρ over $\{0,1\}^n$. We denote by $\mathrm{Ex}(f,\rho)$ the *example oracle* that returns labelled examples $\langle x, f(x)\rangle$ for $x \sim \rho$.

A distPAC-learning algorithm takes three confidence parameters as input. The accuracy parameter ε, the failure parameter δ, and the heuristic parameter η. Essentially, the distPAC-learning model requires that, for a fixed evaluation rule ϕ, there exists some large probability mass of the ϕ-induced concept class $\mathfrak{C}$, as determined by μ and η, that is learnable in Valiant's model.

Definition 1.1 (Distributional PAC-learning)**.** *Let ϕ be an evaluation rule, and let the ϕ-induced concept class $\mathfrak{C}$ be $s(n)$-represented. The pair $(\mathfrak{C},\mu)$ is distributionally PAC-learnable if there exists an algorithm A such that, for any $n \in \mathbb{N}, \varepsilon, \delta, \eta > 0$,*

$$\Pr_{f\sim\mu}\left[\Pr_{A}\left[\forall\rho : \Pr_{x\sim\rho}\left[h(x) \neq f(x) : h \leftarrow A^{\mathrm{Ex}(f,\rho)}(n,\varepsilon,\delta,\eta)\right] \leq \varepsilon\right] \geq 1-\delta\right] \geq 1-\eta \qquad (1)$$

When A runs in time $\mathrm{poly}(n, s(n), \varepsilon^{-1}, \delta^{-1}, \eta^{-1})$, we say that $(\mathfrak{C},\mu)$ is efficiently distPAC-learnable.

Distributional PAC-learning is a clear relaxation of Valiant's PAC-learning, since it no longer requires good learning guarantees for "worst-case" concepts.

The essential difference between distPAC-learning and heurPAC-learning is the requirement that there exists a *single* large probability mass of concepts that is learnable with respect to *any* example distribution (see the location of the universal quantification over ρ in (1)). In heurPAC-learning, the order of quantifiers is different: it is only required that for each example distribution ρ, a large *but possibly different* probability mass of the concept class is learnable. This independently motivates our model for technical and practical reasons: for example, distPAC-learning allows the use of *boosting algorithms*. In other words, the equivalence of weak[2] and strong learning is preserved in our model [Sch90, DW+00] (see Section 3 for a formal statement on this). This is not true in heurPAC-learning.

In a nutshell, distPAC-learning is stronger than both heurPAC-learning and the Blum et al. average-case prediction model. Therefore distPAC-learning also inherits the well-founded motivation behind the theory of heuristic PAC-learning (see [Nan21]). See Section 1.3.1 for a continued discussion. Towards further motivating the distPAC-learning model, in Section 3 we give formal statements on useful properties of the distPAC-learning model, including the equivalence between weak and strong learning, an equivalence between hardness of distribution-specific variant of distPAC-learning and the existence of one-way functions, and finally on using the classic technique of Kearns and Valiant [KV94] for proving hardness of distPAC-learning with respect to *specific* target distributions.

1.1.2 DistPAC-Learning Algorithms from Nisan's Natural Proofs

We design new learning algorithms in the distributional PAC model. These learning algorithms arise from Nisan's natural proof technique, which we now describe in more detail.

Nisan's technique. Nisan [Nis93] used the following communication complexity argument for proving circuit lower bounds against circuits with threshold functions as gates. First, identify a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, which requires *high* 2-party communication complexity in some model (e.g. randomized, determinstic, distributional, etc.). Then, identify a circuit class $\mathcal{C}$ such that for every $g \in \mathcal{C}$, g is computable by a *low-cost* 2-party communication protocol in that model.

[2]A weak learning algorithm is only required to output a hypothesis that has a predictive advantage only slightly better than a coin toss.

Finally, conclude that f requires large $\mathcal{C}$-circuits (see Section 2 for essential definitions of 2-party communication complexity in various models, and [KN96] for further reference). To provide an example, let $f(x, y) = \mathsf{IP2}(x, y) = \sum_{i=1}^{n} x_i y_i \mod 2$ be the inner product mod 2 function. It is known that $\mathsf{IP2}$ requires $\Omega(n)$ bits to be transmitted in any randomized communication complexity protocol; therefore, as shown by [Nis93], since $\mathsf{MAJ} \circ \mathsf{THR}$ circuits are computed by randomized communication complexity protocols with cost $O(\log n)$, $\mathsf{IP2}$ must require $\mathsf{MAJ} \circ \mathsf{THR}$ circuits of exponential size. This lower bound remains one of the strongest known — as of now it is still not ruled out that $\mathsf{NEXP} \subseteq \mathsf{THR} \circ \mathsf{THR}$. In fact, proving NEXP is not contained in $\mathsf{THR} \circ \mathsf{THR}$ is considered a "major frontier" in complexity theory [Che18].

Main Theorem. We now introduce our main theorem, which is used as a "technical centerpiece" for obtaining distPAC-learning algorithms from Nisan's natural lower bounds for $\mathsf{MAJ} \circ \mathsf{THR}$-circuits, which are presented after. Roughly speaking, the main theorem presents a relationship between the computational complexity of distPAC-learning, and the communication complexity of a simple *communication game* associated with a given evaluation rule.

For any evaluation rule ϕ and ϕ-induced $s(n)$-represented concept class $\mathfrak{C}$, we define the associated *communication game* $\mathfrak{G}$ over the product distribution (μ, ρ), played as follows. A binary representation π_f of a function $f \in \mathfrak{C}_n$ is sampled according to μ, and an input x is sampled from ρ. Player one is given the binary representation π_f of f, and player two is given the input x. The two parties communicate until they are ready to output a value b, and win the game if $b = \phi_n(\pi_f, x) = f(x)$. We say that ϕ is *evaluated* by a 2-party distributional communication protocol with cost $c(n)$ and bias $\gamma(n)$ over (μ, ρ), if for every $n \in \mathbb{N}$, the two parties can communicate at most $c(n)$ bits before winning $\mathfrak{G}$ with probability at least $1/2 + \gamma(n)$ over the random inputs drawn from (μ, ρ).

Now we are ready to state the main theorem.

Theorem 1.2. *Let ϕ be an evaluation rule. Suppose that, for any product distribution (μ, ρ), ϕ is evaluated by a 2-party distributional communication protocol with cost $c := c(n)$ and bias $\gamma := \gamma(n)$ over (μ, ρ). Then, for the ϕ-induced $s(n)$-represented concept class $\mathfrak{C}$, and a time $t(n)$-samplable distribution μ, the pair $(\mathfrak{C}, \mu)$ is distributionally PAC-learnable. The learning algorithm runs in time polynomial in $n, s(n), t(n), \varepsilon^{-1}, \delta^{-1}, \eta^{-1}, \gamma^{-1}$ and 2^c.*

We give an overview of the proof of this theorem in Section 1.2. We remark that the exponential dependency of c is likely necessary, since the theorem does not restrict $\mathfrak{C}$ (e.g., it can be $\mathsf{P/poly}$), and no matter what $c \leq n$.

DistPAC-learning from Nisan's natural proofs. Nisan's technique clearly encodes a randomized communication complexity *upper bound*, which, by an averaging argument, can be converted to a distributional protocol over any distribution (without increased cost or decreased bias). Hence, Theorem 1.3 directly follows from a combination of Nisan's lower bounds and Theorem 1.2 since, as indicated in Nisan's lower bounds, every function in $\mathsf{MAJ} \circ \mathsf{THR}$ has a randomized communication protocol of cost $O(\log n)$ and large bias.

Theorem 1.3. *Let $\phi \in \mathsf{MAJ} \circ \mathsf{THR}$ be any evaluation rule, and let μ be any polynomial time samplable target distribution. For the ϕ-induced $s(n)$-represented concept class $\mathfrak{C}$, the pair $(\mathfrak{C}, \mu)$ is efficiently distPAC-learnable.*

Theorem 1.3 considers concept classes by the complexity of their evaluation rule. This is weaker than the learning-theoretic standard of considering the complexity of concepts directly. Any $s(n)$-represented concept class $\mathfrak{C}$ ϕ-induced by the rule $\phi \in \mathcal{C}$ must satisfy $\mathfrak{C} \subseteq \mathcal{C}$ (assuming $\mathcal{C}$-circuits can be poly($s(n)$)-size). For $\mathcal{C}$ containing a universal function (e.g. $\mathsf{P/poly}$, or NC^1), $\mathcal{C} = \mathfrak{C}$, but not necessarily for lower circuit classes.

By non-black-box inspection of the result of [KS09], which proves hardness of learning $\mathsf{MAJ} \circ \mathsf{THR}$ in Valiant's PAC model, we find that it actually provides a polynomial time samplable *distribution* over $\mathsf{MAJ} \circ \mathsf{THR}$-circuits that is hard to learn, even weakly (rather than just worst-case hardness).

In other words, it shows a target distribution μ^* such that hardness of $\tilde{O}(n^{1.5})$-SVP and its variants implies $(\mathsf{MAJ} \circ \mathsf{THR}, \mu^*)$ is not efficiently distPAC-learnable! In Section 3.4, we actually show that the *entire family* of proof techniques for showing hardness of Valiant's PAC-learning (used here by [KS09], and due originally to [KV94]), can be used to prove hardness of distPAC-learning with respect to *specific* polynomial time samplable target distributions.

Therefore, in Theorem 1.3, considering the complexity of the evaluation rule $\phi \in \mathsf{MAJ} \circ \mathsf{THR}$ and distPAC-learning of the ϕ-induced concept class $\mathfrak{C}$ is likely a needed relaxation, since the target distribution in the theorem can be *any* polynomial time samplable distribution. Hence, in order to get a distPAC-learning algorithm for $\mathsf{MAJ} \circ \mathsf{THR}$-circuits, we need to restrict the target distribution somehow.

Natural Target Distributions. In light of this, we show that $\mathsf{MAJ} \circ \mathsf{THR}$-circuits are distPAC-learnable, with respect to the following more natural families of target distributions. By more natural, we mean that the target distribution is not designed by a cryptographer (as opposed to μ^*). Indeed, this highlights another feature of the distPAC-learning model: efficient learnability of $(\mathfrak{C}, \mu)$ for "organic" target distributions μ can coexist with hardness for "inorganic" target distributions like μ^*.

Let $L = (T_1, \cdots T_m)$ be a list of $m := \text{poly}(n)$ linear threshold functions, and let μ be a $\text{poly}(n)$ time samplable distribution over $\{0,1\}^m$. We define the distribution μ_L over $\mathsf{MAJ} \circ \mathsf{THR}$-circuits is sampled as follows. First, sample $\theta \sim \mu$. Then, output the $\mathsf{MAJ} \circ \mathsf{THR}$-circuit that is the majority vote over each $T_i \in L$ such that $\theta_i = 1$.

Theorem 1.4. *Let $L = (T_1, \cdots T_m)$ be any list of $m :=$ poly(n) linear threshold functions, and let μ be any* poly(n) *time samplable distribution over $\{0,1\}^m$. The pair $(\mathsf{MAJ} \circ \mathsf{THR}, \mu_L)$ is efficiently distPAC-learnable.*

Previously, no polynomial time distPAC-learning algorithms were known, for any reasonable type of target distributions over $\mathsf{MAJ} \circ \mathsf{THR}$-circuits. An interesting feature of our distPAC-learning algorithm is that it does not need to know μ or L to work (see Section 6 for details).

We additionally consider slightly modified target distributions, that correspond to interesting and natural distributions over subclasses of $\mathsf{MAJ} \circ \mathsf{THR}$-circuits: polytopes (that is, and-of-thresholds circuits $(\mathsf{AND} \circ \mathsf{THR})$) and DNFs. Note that, in distributional PAC-learning, subclasses are not necessarily distPAC-learnable if their superclass is, since it is possible that the subclass is hard-core, and consisting of functions that are hard for the superclass distPAC-learning algorithm.

For the polytope distribution, let $L = (T_1, \cdots T_m)$ be a list of $m := \text{poly}(n)$ linear threshold functions, and let μ be a $\text{poly}(n)$ time samplable distribution over $\{0,1\}^m$. The distribution $\mu_L^{\wedge}$ over polytopes is sampled as follows. First, sample $\theta \sim \mu$. Then, output the polytope that is the conjunction of all $T_i \in L$ such that $\theta_i = 1$.

Theorem 1.5. *Let $L = (T_1, \cdots T_m)$ be any list of $m :=$ poly(n) linear threshold functions, and let μ be any* poly(n) *time samplable distribution over $\{0,1\}^m$. The pair $(\mathsf{AND} \circ \mathsf{THR}, \mu_L^{\wedge})$ is efficiently distPAC-learnable.*

For the DNF distribution, let $L = (T_1, \cdots T_m)$ be a list of $m := \text{poly}(n)$ disjunctions on n-bit inputs, and let μ be a $\text{poly}(n)$ time samplable distribution over $\{0,1\}^m$. The distribution $\mu_L^{\wedge\vee}$ over DNFs is sampled as follows. First, sample $\theta \sim \mu$. Then, output the DNF that is a conjunction of all disjunctions $T_i \in L$ such that $\theta_i = 1$.

Theorem 1.6. *Let $L = (T_1, \cdots T_m)$ be any list of $m :=$ poly(n) disjunctions on n-bit inputs, and let μ be any* poly(n) *time samplable distribution over $\{0,1\}^m$. The pair $(\text{DNF}, \mu_L^{\wedge\vee})$ is efficiently distPAC-learnable.*

Even though distPAC-learning is stronger that heurPAC-learning, Theorem 1.4, 1.5 and 1.6 are formally incomparable to the heurPAC-learning algorithm for $O(\log n)$-juntas due to [Nan21]. This is for the following reasons. On one hand, Theorem 1.4, 1.5 and 1.6 are stronger because $\mathsf{MAJ} \circ \mathsf{THR}$-circuits, polytopes, and DNFs are strictly more powerful than $O(\log n)$-juntas, and we handle learning over arbitrary example distributions, while [Nan21] only handles the uniform example distribution. However, the confounding variable is that the heurPAC algorithm works with respect to the uniform distribution over $O(\log n)$-juntas, while for any L, μ, the target distributions that we learn are not uniform over their support. We thus cannot show that our algorithm is stronger than Nanashima's in a formal sense.

1.1.3 Impossibility of Encoded-Input Weak PRFs

Although we do not obtain any PAC-learning algorithm in Valiant's model from Nisan's natural proofs, we show that distributional PAC-learning is still enough to rule out weak PRFs (which was one of the initial motivations of studying Question 1). In fact, we show that this is true even when the weak PRF is allowed an arbitrary input encoding.

Theorem 1.7. *There exists no encoded-input weak PRF that is evaluated by a* $\mathsf{MAJ} \circ \mathsf{THR}$*-circuit.*

Our notion of encoded-input weak PRF is the natural weak analogue of the encoded-input PRF introduced by [BIP+18]. Loosely speaking, an encoded-input weak PRF is a PRF that is only required to be secure when the adversary sees random points, where the inputs are taken uniformly at random from a predefined multi-subset of the input space. We refer to Section 7 for details.

1.2 Proof Overview of Theorem 1.2

We now overview the ideas behind the proof of Theorem 1.2. The most important tool we use is the 2-party norm of a function, $R_2(f)$, which is defined to be the expected product of a function computed on a list of correlated inputs.

Definition 1.2 (2-party norm). *For* $f : (\{0,1\}^n)^2 \to \{-1,1\}$*, the* 2*-party norm of* f *is defined as*

$$R_2(f) := \mathop{\mathbb{E}}_{x_1^0, x_2^0, x_1^1, x_2^1 \sim \{0,1\}^n} \left[\prod_{\varepsilon_1, \varepsilon_2 \in \{0,1\}} f(x_1^{\varepsilon_1}, x_2^{\varepsilon_2}) \right] \tag{2}$$

The 2-party norm is a special case of the k-party norm (sometimes called the cube-measure), which was introduced by [BNS92] for obtaining lower bounds in k-party Number-on-Forehead communication complexity.

The crucial property about $R_2(f)$ is that, up to parameters, it upper bounds the correlation of f with functions computable by deterministic 2-party communication protocols. We denote by $\Pi[2, c]$ the set of all $f : (\{0,1\}^n)^2 \to \{-1,1\}$ that have deterministic 2-party communication protocols with cost at most c. For a definition of the deterministic 2-party communication model, see Section 2.1.

Implicit in all three of [CT93, Raz00, VW07] (who showed a related theorem in the more general k-party case), is the following bound:

Theorem 1.8 (*The* correlation bound — [CT93, Raz00, VW07])**.** *For every function* $f : (\{0,1\}^n)^2 \to \{-1,1\}$,

$$\mathrm{Cor}(f, \Pi[2,c]) = \max_{\pi \in \Pi[2,c]} \left| \mathop{\mathbb{E}}_{x} [f(x) \cdot \pi(x)] \right| \leq 2^c \cdot R_2(f)^{1/4} \tag{3}$$

for x *uniformly distributed over* $(\{0,1\}^n)^2$.

Equation (3) implies that $(2^{-c} \cdot \mathrm{Cor}(f, \Pi[2,c]))^4 \leq R_2(f)$.

The construction of the learning algorithm of Theorem 1.2 uses the lower bound on $R_2(f)$ to distinguish structure from randomness. In other words, hypothetically consider functions $f : (\{0,1\}^n)^2 \to \{-1,1\}$ such that the quantity $(2^{-c} \cdot \mathrm{Cor}(f, \Pi[2,c]))^4$ is relatively *large* (greater than $1/\mathrm{poly}(n)$, say). Such functions can be distinguished from uniformly random functions, by taking a random sample from the distribution over the value inside the expectation in (2). This follows from the fact that $R_2(\psi)$ for a uniformly random function $\psi : (\{0,1\}^n)^2 \to \{-1,1\}$ is bounded from above by a negligible function of n.

Using this idea, we have the following proof outline. First, we can try to prove a "distinguisher-to-predictor" lemma, in the style of [Yao82], in order to obtain a weak randomized predictor for f (a weak predictor requires accuracy of a prediction for an unseen example to be only slightly more accurate than a coin toss). Second, we could apply standard averaging arguments to construct a weak PAC-learning algorithm. Finally, we could apply celebrated boosting results from learning theory [Sch90, DW$^+$00] to produce a full-blown PAC-learning algorithm.

However, this proof outline remains incomplete. First, the 2-party norm of the function is the expectation of a product of *correlated* inputs, so we have not given any way of using independent random examples. Second, we have said nothing of how to handle arbitrary example distributions (the inputs to f on the right hand side of (3) should be uniformly random). We handle both of these problems simultaneously, roughly by thinking of f as the *evaluation rule*, and not the concept itself.

First, let us describe how f should be viewed in more detail. There are two inputs to f, x_1 and x_2. Without loss of generality, identify x_1 as a random string for sampling the target distribution μ, and identify x_2 as a random string for sampling the example distribution ρ, with $|x_1| = |x_2| = m$. We abuse the notation and let $z = \rho(x_2)$ to denote a point z sampled according to ρ with the random bits x_2. Similarly, we let g be the function represented by $\pi_g = \mu(x_1)$. Next, fix the evaluation rule ϕ, which is the map that takes as input the concept representation π_g, plus the input z, and outputs $\phi(\pi_g, z) = g(z) = y$. As a function of x_1, x_2, we can thus write the process of generating a labelled example as $\langle \rho(x_2), \phi(\mu(x_1), \rho(x_2))\rangle = \langle z, g(z)\rangle = \langle z, y\rangle$. We let $f(x_1, x_2) = \phi(\mu(x_1), \rho(x_2))$. This allows us to write:

$$R_2(f) = \mathop{\mathbb{E}}_{x_1^0, x_2^0, x_1^1, x_2^1}\left[v(x_1^0, x_2^0, x_1^1, x_2^1)\right] \text{ for } v(x_1^0, x_2^0, x_1^1, x_2^1) := \prod_{\varepsilon_1, \varepsilon_2 \in \{0,1\}} \phi(\mu(x_1^{\varepsilon_1}), \rho(x_2^{\varepsilon_2}))$$

Now, we describe how we construct a weak randomized predictor which only uses random examples from an arbitrary ρ. At the core, we will use the example oracle to sample a single instance of $v(x_1^0, x_2^0, x_1^1, x_2^1)$, over uniformly random $x_1^0, x_2^0, x_1^1, x_2^1 \in \{0,1\}^m$. To see the significance of this, observe that by definition $v(x_1^0, x_2^0, x_1^1, x_2^1)$ has expected value $R_2(f)$. Hence, the process of sampling this value distinguishes examples labelled by uniformly random functions from examples labelled by concepts sampled according to μ — as long as μ samples representations of concept that are evaluated by ϕ. This claim is justified because whenever it is possible to win the communication game $\mathfrak{G}$ associated with ϕ with high bias and low communication, Theorem 1.8 implies that $R_2(f)$ is large. In other words, $R_2(f)$ is guaranteed to be large whenever it is possible to efficiently (probabilistically) communicate the evaluation rule ϕ (because this implies winning $\mathfrak{G}$ with good bias). At this point, we use a simple hybrid argument to construct a randomized prediction algorithm for examples sampled according to ρ.

It remains to verify that the randomized prediction algorithm can actually sample $v(x_1^0, x_2^0, x_1^1, x_2^1)$, using only access to $\mathrm{Ex}(g, \rho)$, where g is the concept sampled according to the target distribution μ. To see this, observe that the distribution over $v(x_1^0, x_2^0, x_1^1, x_2^1)$ is identical to the distribution over $g(z_1)g(z_2)h(z_1)h(z_2)$, for $\langle z_1, g(z_1)\rangle, \langle z_2, g(z_2)\rangle \sim \mathrm{Ex}(g, \rho)$, and $h \sim \mu$. The value $h(z_1)h(z_2)$ can be computed because $h(z_1)$ and $h(z_2)$ can be *queried*, since h is sampled *locally* by the algorithm. Therefore, we only need $\mathrm{Ex}(g, \rho)$.

We also need to verify that ρ need not be efficiently samplable. To argue this this, we observe that communicating parties participating in $\mathfrak{G}$ have unbounded computational power. This means that, even if ρ is an arbitrary distribution, there is no effect on the communication cost of $\mathfrak{G}$. Indeed, the process of sampling ρ can be viewed as a local pre-processing step in the protocol for party two. Therefore, $R_2(f)$ does not decrease when ρ is arbitrary.

1.3 Discussion

1.3.1 Distributional PAC-Learning vs. Related Models

As mentioned previously, distPAC-learning strengthens heurPAC-learning. This is due to the stronger quantification over example distributions. The main benefit of this is that it facilitates boosting of weak learning algorithms, which needs worst-case guarantees over the example distribution (see Section 3.1 for a formal statement). We encourage the reader to visit Section 1.2 of [Nan21], as their points regarding the motivations of heurPAC-learning as a relaxation of Valiant's PAC model, largely apply to distPAC-learning as well. Additionally, see Section 1.2 of [Nan21] for commentary of the differences with previous "implicit" definitions of average-case learning, such as in [JS05, JLSW11, Sel09] also apply to distPAC-learning.

In comparison to the seminal work of [BFKL93], distPAC-learning also differs on the order of quantifiers. In the definition of the average-case prediction considered by [BFKL93], both the target distribution μ and the example distribution ρ are fixed. This means that there can be a different prediction algorithm, for each pair of μ and ρ. This model is weaker than both the heurPAC-learning and distPAC-learning models.

See Section 3 for formal statements on other useful properties of the distPAC-learning model, such as on the relationship between cryptography and the hardness (or lack thereof) of distPAC-learning.

1.3.2 Other Related Work

Many other relationships between learning theory and communication complexity have been studied. Some notable examples include [KNR99, LS09, FX14, KLMY19] (also see the references therein). All of these works study relationships between communication complexity and notions of learning complexity, such as sample complexity [KNR99, KLMY19], differentially private sample complexity [FX14], margin complexity [LS09], VC dimension [KNR99, FX14] and Littlestone dimension [FX14]. These works are all incomparable to ours, as they do not directly study relationships between communication complexity and the *computational* complexity of learning.

Learning intersections of halfspaces (i.e., ands of linear threshold functions) was considered by [KOS04]. Using Fourier-analytic techniques, [KOS04] showed a polynomial time learning algorithm for any function of a constant number of halfspaces with respect to the uniform distribution over examples. Additionally, [KOS04] gave a quasi-polynomial time algorithm for learning any Boolean function of a *polylogarithmic* number of bounded-weights linear threshold functions, under *any* distribution over examples. Our learning results (Theorem 1.3, and Theorem 1.4) are at the moment similar but incomparable; we get polynomial time *distributional* PAC-learning of concepts *evaluated* by majorities of linear threshold functions over *any example distribution.*

1.3.3 Additional Remarks and Future Work

In this work, we began by observing that if Question 1 resolves to "yes, for every Λ," then cryptographic assumptions such as quantum polynomial time hardness of $\tilde{O}(n^{1.5})$-SVP (and its variants) do not hold. To continue our study, we shifted the focus to understanding what kind of learning algorithms *are* implied, specifically by Nisan's natural proofs. To this end, we introduced the distPAC-learning model as a relaxation of Valiant's PAC model. Using Nisan's natural proofs, we at least obtained novel learning algorithms in the distributional PAC model for $\mathsf{MAJ} \circ \mathsf{THR}$, over natural target distributions, and more generally, any concept class induced by evaluation rules contained in $\mathsf{MAJ} \circ \mathsf{THR}$.

Towards this result, we exploited the specific aspects of Nisan's lower bound method. Therefore, it remains open whether or not *other* natural proofs imply efficient distPAC-learning algorithms for *other* concept classes, such as $\mathsf{AC}^0[p]$. In particular, the natural proofs for $\mathsf{AC}^0[p]$ of [Raz87, Smo87] are not affected by Theorem 1.1, so presumably they could even imply algorithms in Valiant's model. At present, the difficulty in proving a similar barrier to Theorem 1.1 for $\mathsf{AC}^0[p]$ is that, while we are able to prove hardness of PAC-learning in Valiant's model, we do not have any exponentially strong natural proofs. We regard this question as the primary direction for future research. We note that works such as [BCG$^+$21] have introduced conjectured weak PRF candidates that can be evaluated by $\mathsf{AC}^0[2]$, with considerable evidence to support *subexponential* security of the candidates. A weak PRF evaluated by $\mathsf{AC}^0[2]$ with subexponential security would preclude any (even quasipolynomial time) distPAC-learning algorithm for concept classes induced by a $\mathsf{AC}^0[2]$ evaluation rule. That being said, we hope that our results shed light on what aspects of natural proofs are useful for learning algorithms that cannot query the concept.

Finally, an interesting direction is to obtain distPAC-learning algorithms for other natural distributions over $\mathsf{MAJ} \circ \mathsf{THR}$-circuits, polytopes, and DNFs.

2 Preliminaries and Definitions

2.1 2-Party Communication Complexity and Norms

In the following, we discuss Boolean functions that output -1 or 1.

The 2-party communication model is the following. There are 2 parties, each having unbounded computational power, who try to collectively compute a function. The input to the function is separated into 2 segments, and the i^{th} party sees the i^{th} segment. The parties can send each other direct messages.

Each party may transmit messages according to a fixed protocol. The protocol determines, for every sequence of bits transmitted up to that point (the transcript), whether the protocol is finished (as a function of the transcript), or if, and which, party writes next (as a function of the transcript) and what that party transmits (as a function of the transcript and the input of that party). Finally, the last bit transmitted is the output of the protocol, which is a value in $\{-1, 1\}$. The complexity measure of the protocol is the total number of bits transmitted by the parties.

Definition 2.1 ($\Pi[2, c]$ class)**.** $\Pi[2, c]$ *is defined to be the class of functions* $f : (\{0, 1\}^n)^2 \to \{-1, 1\}$ *that can be computed by a* 2*-party deterministic communication protocol with complexity* c.

Another communication model is *randomized* communications.

Definition 2.2 (Randomized $\Pi[2, c]$)**.** *The randomized* 2*-party communication model allows the protocol to depend on random bits. Therefore, we allow the protocol to err in its output. The probability of error of a randomized protocol is* ε *if for every input to the function* f*, the protocol errs in outputs with probability at most* ε*. We denote by* $\mathrm{r}\Pi[2, c, \gamma]$ *the class of* 2*-party randomized protocols that transmit at most* c *bits and err with probability at most* $1/2 - \gamma$.

For the sake of simplicity, this paper uses only the public coin version of randomized communication complexity. Namely the parties all share a string of random bits.

A model more relaxed than randomized communication is *distributional* communication.

Definition 2.3 (Distributional $\Pi[2, c]$)**.** *The distributional* 2*-party communication model allows the protocol to err on certain inputs. Fix a distribution* ρ *over* $(\{0, 1\}^n)^2$*. A function* $f : (\{0, 1\}^n)^2 \to \{-1, 1\}$ *is in* $\mathrm{d}\Pi[2, c, \rho, \gamma]$ *if there exists a communication protocol* $p \in \Pi[2, c]$ *such that*

$$\mathop{\mathbb{E}}_{(x_1,x_2)\sim\rho} [p(x_1, x_2) \cdot f(x_1, x_2)] \geq 2\gamma$$

Distributional communication complexity can be thought of as correlation.

Definition 2.4 (Boolean function correlation)**.** *Define* $\mathrm{Cor}(f, \Lambda) := \max_{h\in\Lambda} |\mathbb{E}[f(x) \cdot h(x)]|$*, where* x *is sampled uniformly at random from the domain.*

When we want to measure correlation between two function classes, we have it defined as follows:

Definition 2.5 (Boolean function correlation)**.** *Define* $\mathrm{Cor}(\mathfrak{C}, \Lambda) := \min_{f\in\mathfrak{C}} \max_{h\in\Lambda} |\mathbb{E}[f(x) \cdot h(x)]|$*, where* x *is sampled uniformly at random from the domain.*

When ρ is the uniform distribution, $f \in \mathrm{d}\Pi[2, c, \rho, \gamma]$ is equivalent to $\mathrm{Cor}(f, \Pi[2, c]) \geq 2\gamma$.

A simple fact is that for any distribution ρ, $f \in \mathrm{r}\Pi[2, c, \gamma]$ implies that $f \in \mathrm{d}\Pi[2, c, \rho, \gamma]$. Therefore, $f \in \mathrm{r}\Pi[2, c, \gamma]$ implies that $\mathrm{Cor}(f, \Pi[2, c]) \geq 2\gamma$.

Definition 2.6 (2-party norm)**.** *For $f : (\{0,1\}^n)^2 \to \{-1,1\}$, the 2-party norm of f is defined as*

$$R_2(f) := \mathop{\mathbb{E}}_{x_1^0, x_2^0, x_1^1, x_2^1 \sim \{0,1\}^n} \left[\prod_{\varepsilon_1, \varepsilon_2 \in \{0,1\}} f(x_1^{\varepsilon_1}, x_2^{\varepsilon_2}) \right] \tag{4}$$

The 2-party norm is a special case of the k-party norm (sometimes called the cube-measure), which was introduced by [BNS92] for obtaining lower bounds in k-party Number-on-Forehead communication complexity.

The crucial property about $R_2(f)$ is that, up to parameters, it upper bounds the correlation of f with functions computable by 2-party communication protocols. Implicit in all three of [CT93, Raz00, VW07] (who showed a related theorem in the more general k-party case), is the following bound:

Theorem 2.1 (*The* correlation bound — [CT93, Raz00, VW07])**.** *For every function $f : (\{0,1\}^n)^2 \to \{-1,1\}$,*

$$\mathrm{Cor}(f, \Pi[2,c]) \le 2^c \cdot R_2(f)^{1/4} \tag{5}$$

An immediate corollary of this bound is:

Theorem 2.2. *For every function $f : (\{0,1\}^n)^2 \to \{-1,1\}$, such that $f \in \mathrm{r}\Pi[2,c,\gamma]$,*

$$\gamma \le 2^c \cdot R_2(f)^{1/4} \tag{6}$$

2.2 (Weak) Pseudorandom Functions

For clarity, we define weak and strong pseudorandom functions. See Section 7 for the definition of encoded-input weak PRFs.

Definition 2.7 (Weak and Strong PRFs)**.** *Let λ be a security parameter, and $n = n(\lambda), \kappa = \kappa(\lambda)$ for polynomially bounded functions n, κ. Consider a pair of algorithms $\mathsf{f} : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^1, \mathsf{gen} : \{1\}^\lambda \to \{0,1\}^\kappa$.*

- gen *is a polynomial time sampling algorithm that given input parameter λ in unary and access to random coins $z \in \{0,1\}^{\mathrm{poly}(\lambda)}$ outputs a key $k \in \{0,1\}^\kappa$.*
- f *is a polynomial time algorithm that given a key k and the input $x \in \{0,1\}^n$, outputs a value $\mathsf{f}(k,x) = v \in \{0,1\}^1$.*

For $t = t(\lambda), \varepsilon = \varepsilon(\lambda)$, we say that $(\mathsf{f}, \mathsf{gen})$ is a (t,ε)-weak PRF if, for every size t oracle circuit C,

$$\left| \Pr_{k \sim \mathsf{gen}(1^\lambda)} \left[C^{\mathrm{Ex}(\mathsf{f}(k,\cdot), U_n)} = 1 \right] - \Pr_r \left[C^{\mathrm{Ex}(r, U_n)} = 1 \right] \right| \le \varepsilon(\lambda)$$

where $r : \{0,1\}^n \to \{0,1\}^1$ is a uniformly random function.

Additionally, we say that $(\mathsf{f}, \mathsf{gen})$ is a (t,ε)-PRF if, for every size t oracle circuit C,

$$\left| \Pr_{k \sim \mathsf{gen}(1^\lambda)} \left[C^{\mathsf{f}(k,\cdot)} = 1 \right] - \Pr_r \left[C^r = 1 \right] \right| \le \varepsilon(\lambda)$$

where $r : \{0,1\}^n \to \{0,1\}^1$ is a uniformly random function.

When $t, \varepsilon = 2^{\log^c(n)}$ for some constant $c > 1$, we say that $(\mathsf{f}, \mathsf{gen})$ has *quasipolynomial* security. When $t, \varepsilon = 2^{\lambda^\delta}$ for some constant $\delta \in (0,1)$, we say that $(\mathsf{f}, \mathsf{gen})$ has *subexponential* security.

2.3 Circuit Classes and Other Computational Classes

We will consider various circuit classes with different bases (all being defined previously in the literature). AC^0 is the class of constant depth, polynomial size, unbounded fan-in $\mathsf{AND}/\mathsf{OR}/\mathsf{NOT}$ circuits. $\mathsf{AC}^0[p]$ is the class of constant epth, polynomial size, unbounded fan-in $\mathsf{AND}/\mathsf{OR}/\mathsf{NOT}/\mathsf{MOD}p$ circuits, where $p \in \mathbb{N}$ is a prime number. TC^0 is the class of constant-depth, polynomial size, unbounded fan-in circuits of THR gates, where a THR gate is a linear threshold function $t(x_1, \cdots x_m) := \sum_{i=1}^{m} w_i x_i \geq^? \theta$, which outputs 1 if and only if the sum of the inputs weighted by $w_1, \cdots w_m$ exceeds a threshold θ. When the weights are fixed to be 1 and $\theta = m/2$, we call it a MAJ gate. An XOR gate takes the sum modulo 2 of its inputs.

Many circuit classes considered are of the form $\mathcal{C} \circ \mathcal{C}'$ for circuit classes $\mathcal{C}, \mathcal{C}'$. The composed class $\mathcal{C} \circ \mathcal{C}'$ denotes the class of $\mathcal{C}$-circuits with inputs as the outputs of functions from $\mathcal{C}'$. For example, the class $\mathsf{THR} \circ \mathsf{THR}$ (a.k.a. depth-2 TC^0). Alternatively, $\mathsf{MAJ} \circ \mathsf{THR}$ is the class of circuits consisting of a MAJ gate composed with a bottom layer of THR gates.

2.4 Lattice Problems

An n-dimensional lattice L is the set of all integer linear combinations of a given real basis $v_1 \cdots v_n \in \mathbb{R}$. Mathematically,

$$L = \left\{ \sum_{i \in [n]} c_i v_i : c_1, \cdots c_n \in \mathbb{Z} \right\}$$

In the unique shortest vector problem, $f(n)$-uSVP, one must find the shortest non-zero vector in the lattice (if there are many, then any is valid), with the promise that the shortest one is shorter, by at least an $f(n)$ factor, than any other non-parallel vector in the lattice. In the Shortest Vector Problem (SVP), $f(n)$-SVP, one must approximate the length of the shortest non-zero vector in the lattice within a factor of $f(n)$. Lastly, in the Shortest Independent Vector Problem ($f(n)$-SIVP), the object is to find a collection of n linearly independent lattice vectors of length at most $f(n) \cdot a$, where a is the minimum length of the longest vector present in some subset of n vectors in the lattice that is linearly independent.

In general, the hardness of uSVP, SVP, and SIVP increases as the parameter $f(n) \leq \mathrm{poly}(n)$ decreases. This work uses, as a black box, hardness results for PAC-learning $\mathsf{MAJ} \circ \mathsf{THR}$-circuits due to Klivans and Sherstov [KOS04]. Klivans and Sherstov in turn rely in a black box way on public key cryptosystems of Regev [Reg04] to prove their hardness results. The cryptosystems of Regev rely on polynomial time hardness of uSVP, and quantum polynomial time hardness of SVP and SIVP.

3 The Distributional PAC-Learning Model

In this section, we give some useful properties of distPAC-learning, which motivate the definition independently. First, we cover the equivalance between weak and strong versions of distPAC-learning. Then, we show that distPAC-learning implies heurPAC-learning. Finally, we consider the realationship of distPAC-learning with cryptography, by demonstrating that hardness of distPAC-learning for polynomial size circuits in a distribution-specific setting is equivalent to the existence of (infinitely-often) one-way functions. Finally, we show that the famous technique of [KV94] for proving hardness of Valiant's PAC-learning using public key encryption schemes generates "hard target distributions" for distPAC-learning.

3.1 Equivalence Between Weak and Strong DistPAC-Learning

Celebrated results of computational learning theory, indicate that efficient weak and strong PAC-learning are equivalent [Sch90] (in the "filtering" setting, this is shown by e.g. [DW$^+$00]).

Recall that $\mathfrak{C} = \{\mathfrak{C}_n\}_{n \in \mathbb{N}}$ is a Boolean concept class that is $s(n)$-represented and induced by the evaluation rule $\phi = \{\phi_n\}_{n \in \mathbb{N}}$. We define a weak version of distPAC-learning. The only difference with Definition 1.1 is that the accuracy of the hypothesis only needs to achieve error below $1/2 - \varepsilon$ with high probability.

Definition 3.1 (Weak distPAC-learning)**.** *Let ϕ be an evaluation rule, and let the ϕ-induced concept class $\mathfrak{C}$ be $s(n)$-represented. The pair $(\mathfrak{C}, \mu)$ is weakly distributionally PAC-learnable if there exists an algorithm A such that, for any $n \in \mathbb{N}, \delta, \eta > 0$,*

$$\Pr_{f \sim \mu}\left[\Pr_{A}\left[\forall \rho : \Pr_{x \sim \rho}\left[h(x) \neq f(x) : h \leftarrow A^{\mathrm{Ex}(f,\rho)}(n, \delta, \eta)\right] \leq 1/2 - 1/\mathrm{poly}(n)\right] \geq 1 - \delta\right] \geq 1 - \eta \quad (7)$$

When A runs in time $\mathrm{poly}(n, s(n), \delta^{-1}, \eta^{-1})$, we say that $(\mathfrak{C}, \mu)$ is efficiently distPAC-learnable.

We now demonstrate that the boosting results from Valiant's PAC model carry over to the distPAC-learning model.

Theorem 3.1 (distPAC Boosting)**.** *Let ϕ be an evaluation rule, and let $\mathfrak{C}$ be the ϕ-induced $s(n)$-represented concept class. For any target distribution μ, $(\mathfrak{C}, \mu)$ is efficiently weakly distributionally PAC-learnable if and only if $(\mathfrak{C}, \mu)$ is efficiently distributionally PAC-learnable.*

Proof. We invoke the equivalence of weak and strong PAC-learning [Sch90, DW+00] to conclude the desired expression. Note that, importantly, our weak distributional PAC learner works for all ρ after taking the probability over $f \sim \mu$ and the randomness of the learning algorithm A. If the quantifiers were in another order, then we could not guarantee boosting since there would be no guarantee that the same set of functions of $\mathfrak{C}$, understood as a subset of $\{0,1\}^{s(n)}$, could be learned. ■

3.2 DistPAC-Learning vs HeurPAC-Learning

Theorem 3.2. *Let ϕ be an evaluation rule, and let $\mathfrak{C}$ be the ϕ-induced $s(n)$-represented concept class. Let U be the uniform distribution over $\{0,1\}^{s(n)}$. If $(\mathfrak{C}, U)$ is distPAC-learnable in time t, then $\mathfrak{C}$ is heurPAC-learnable in time t.*

Proof. By definition, we can take the distPAC learner A for $(\mathfrak{C}, U)$ and apply it as a heurPAC learner for $\mathfrak{C}$. To see this, observe that A satisfies the heurPAC learning guarantee because it already obtains PAC-learning (i.e., learning with respect to *any* example distribution) for a $1 - \eta$ fraction of $\mathfrak{C}$. For heurPAC-learning, we in fact only need that for each example distribution, a $1 - \eta$ fraction of $\mathfrak{C}$ is learnable. ■

3.3 Distribution-Specific Learning

To obtain cryptographic primitives from the hardness of learning, the literature often considers hardness of *distribution-specific* learning (as in [BFKL93, Nan21]). In distribution-specific distPAC-learning, we intentionally fix the example distribution ρ. This limits the robustness of distPAC-learning, and entirely collapses it into Nanashima's notion of distribution-specific heurPAC-learning.

Definition 3.2 (Distribution-specific distPAC-learning)**.** *Let ϕ be an evaluation rule, and let the ϕ-induced concept class $\mathfrak{C}$ be $s(n)$-represented. The pair $(\mathfrak{C}, \mu)$ is distributionally PAC-learnable over the example distribution ρ if there exists an algorithm A such that, for any $n \in \mathbb{N}, \varepsilon, \delta, \eta > 0$,*

$$\Pr_{f \sim \mu}\left[\Pr_{A}\left[\Pr_{x \sim \rho}\left[h(x) \neq f(x) : h \leftarrow A^{\mathrm{Ex}(f,\rho)}(n, \varepsilon, \delta, \eta)\right] \leq \varepsilon\right] \geq 1 - \delta\right] \geq 1 - \eta \quad (8)$$

When A runs in time $\mathrm{poly}(n, s(n), \varepsilon^{-1}, \delta^{-1}, \eta^{-1})$, we say that $(\mathfrak{C}, \mu)$ is efficiently distPAC-learnable over ρ.

3.3.1 Cryptography from Hardness of Distribution-Specific DistPAC-Learning

The collapse of distPAC-learning and heurPAC-learning in the distribution-specific setting is a feature of distPAC-learning: using results of [Nan21] on one-way functions from hardness of distribution-specific heurPAC-learning, we obtain the same in distribution-specific distPAC-learning.

Theorem 3.3 (OWFs from hardness of ρ-specific distPAC-learning)**.** *Suppose that* $\mathsf{P/poly}$ *is not efficiently distPAC-learnable with respect to the uniform distribution over concept representations, on a polynomial time samplable example distribution ρ. Then, there exists an (infinitely-often) one-way function.*

Proof. Let U be the uniform distribution over $\{0,1\}^{\mathrm{poly}(n)}$. Observe that $(\mathsf{P/poly}, U)$ is efficiently distPAC-learnable on ρ if and only if $\mathsf{P/poly}$ is efficiently heurPAC-learnable on ρ. Now, the claim follows immediately from Corollary 9 of [Nan21]. ■

3.4 Hardness of DistPAC-Learning from Cryptography

In the other direction, we can use the classic results [GGM86, HILL99] to easily see that the existence of a one-way function implies that there exists a polynomial time samplable μ such that $(\mathsf{P/poly}, \mu)$ is not efficiently distPAC-learnable on any polynomial time samplable ρ.

Additionally, we now demonstrate that the technique for proving hardness of PAC-learning in Valiant's model (by Kearns and Valiant [KV94]), also works for distPAC-learning. The technique gives concrete "hard target distributions" for distPAC-learners. Note, it does not imply that distPAC-learning is hard *for every* target distribution.

We sketch the proof technique of Kearns and Valiant, but direct the reader to [KV94] for the technicalities and also [KS09] for a more in-depth overview. The proof technique is built around the idea that learning the decryption function of a public key cryptosystem should be hard. Indeed, this is true by the following reasoning. For any given public key cryptosystem, an attacker can simulate an example oracle for the decryption function, by sampling a public and private key pair, and then generating encryptions of ones and zeros (with equal probability), using the randomized encryption function. Then, if there is a PAC-learning algorithm (in Valiant's model) for the decryption function, the attacker can apply it to the dataset collected as described, and then predict the messages of new ciphertexts, which breaks security.

We show that it suffices to have a distPAC-learning algorithm, rather than a PAC-learning algorithm in Valiant's model, to break security of a public key cryptosystem in this way.

Theorem 3.4. *Suppose there exists a secure public key encryption scheme* $(\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$ *that generates n-bit encryptions of single bit messages, where decryption error is $\varepsilon := \varepsilon(n) \leq 1/n^{\omega(1)}$. Let μ be the distribution over decryption functions* $\mathsf{dec}(\mathsf{sk}, \cdot)$ *with* $\mathsf{sk}, \mathsf{pk} \sim \mathsf{gen}$ *and hard-wired. If for every* sk, $\mathsf{dec}(\mathsf{sk}, \cdot) \in \Lambda$*, then (Λ, μ) is not efficiently distPAC-learnable.*

Proof. Suppose towards a contradiction that (Λ, μ) is efficiently distPAC-learnable using an algorithm A. We will show that there is a probabilistic polynomial time algorithm D that breaks the security of $(\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$, by proving the following distinguishing equation:

$$\mathsf{adv}(D) := |\Pr[D(\mathsf{pk}, \mathsf{enc}(\mathsf{pk}, 1)) = 1] - \Pr[D(\mathsf{pk}, \mathsf{enc}(\mathsf{pk}, 0)) = 1]| \geq 1/\mathrm{poly}(n) \tag{9}$$

Here, the probabilities are taken over the internal randomness of D and the encryption function, as well as $\mathsf{sk}, \mathsf{pk} \sim \mathsf{gen}$. $D(\mathsf{pk}, z)$ works as follows.

1. Prepare a simulated oracle $\mathrm{Ex}(\mathsf{dec}(\mathsf{sk}, \cdot), \rho)$, where each example is created by first choosing $y \in \{0,1\}$ uniformly randomly, and then encrypting $x = \mathsf{enc}(\mathsf{pk}, y)$, and taking $\langle x, y \rangle$ as the example. ρ denotes the marginal distribution over examples x.

2. Execute $h \leftarrow A^{\mathrm{Ex}(\mathsf{dec}(\mathsf{sk},\cdot),\rho)}(n, \frac{1}{10}, \frac{1}{10}, \frac{1}{10})$.

3. Output $h(z)$.

We prove (9) holds. Since A is an efficient distPAC-learner, it only requires at most $\mathrm{poly}(n)$ examples. Therefore, since decryption error is $\varepsilon := 1/n^{\omega(1)}$, a union bound indicates that with probability $1 - 1/n^{\omega(1)}$, no decryptions occur. Assume this is the case. By the assumption that A is an efficient distPAC-learning algorithm for (Λ, μ), we have that, over the randomness of $h \leftarrow A^{\mathrm{Ex}(\mathsf{dec}(\mathsf{sk},\cdot),\rho)}(n, \frac{1}{10}, \frac{1}{10}, \frac{1}{10})$, and the randomness of $\mathsf{dec}(\mathsf{sk}, \cdot) \sim \mu$ and $x \sim \rho$,

$$\Pr\left[h(x) = \mathsf{dec}(\mathsf{sk}, x)\right] \geq \frac{1}{2} + \frac{1}{\mathrm{poly}(n)} \tag{10}$$

Therefore,

$$\begin{aligned}
\mathsf{adv}(D) &= \left| \Pr_{D,\mathsf{gen},\mathsf{enc}}[h(\mathsf{enc}(\mathsf{pk}, 1)) = 1] - \Pr_{D,\mathsf{gen},\mathsf{enc}}[h(\mathsf{enc}(\mathsf{pk}, 0)) = 1] \right| = 2 \Pr_{\substack{D,\mathsf{gen},\mathsf{enc},\\ y\sim\{0,1\}}}[h(\mathsf{enc}(\mathsf{pk}, y)) = y] - 1 \\
&= 2\left(\Pr_{\substack{D,\mathsf{gen},\mathsf{enc},\\ y\sim\{0,1\}}}[h(\mathsf{enc}(\mathsf{pk}, y)) = \mathsf{dec}(\mathsf{sk}, (\mathsf{enc}(\mathsf{pk}, y)))] - \Pr_{\substack{D,\mathsf{gen},\mathsf{enc},\\ y\sim\{0,1\}}}[\mathsf{dec}(\mathsf{sk}, (\mathsf{enc}(\mathsf{pk}, y))) \neq y] \right) - 1 \\
&= 2\left(\Pr_{\substack{D,\mathsf{gen},\mathsf{enc},\\ x\sim\rho}}[h(x) = \mathsf{dec}(\mathsf{sk}, x)] - \varepsilon \right) - 1 \\
&\geq \frac{1}{\mathrm{poly}(n)}
\end{aligned}$$

Since we assumed that no decryption error occurred, we need to consider the case that it does. We already know that probability of any decryption error is $1/n^{\omega(1)}$. Hence, by another union bound, we get that $\mathsf{adv}(D) \geq 1/\mathrm{poly}(n) - 1/n^{\omega(1)} \geq 1/\mathrm{poly}(n)$. ■

4 No General Implication from Natural Proofs to PAC-Learning

In this section, we show that for circuit classes that can compute $\mathsf{MAJ} \circ \mathsf{THR}$-circuits, there can be no general implication from natural proofs to PAC-learning algorithms in Valiant's model, unless $\tilde{O}(n^{1.5})$-uSVP has a polynomial time solution, and $\tilde{O}(n^{1.5})$-SVP and $\tilde{O}(n^{1.5})$-SIVP have polynomial time quantum solutions. We sketch an analogous fact for natural proofs for DNFs, and a natural assumption on the polynomial time hardness of refuting random k-SAT instances. Additionally, we will explain that there exists a special target distribution such that there cannot even be an implication from natural proofs to distPAC-learning algorithms for that distribution, under the same lattice assumptions.

4.1 Nisan's Natural Proof

We will formally define the natural property that arises from Nisan's natural proofs. We begin by defining what is a natural property. Let F_n be the set of all Boolean functions on n inputs. Typically, the Boolean functions in the context of circuit complexity are $f : \{0,1\}^n \to \{0,1\}$. We will continue to use $f : \{0,1\}^n \to \{-1,1\}$ to maintain continuity with the previous sections.

Definition 4.1 (Natural Property [RR97])**.** *A Natural Property is a sequence of subsets $Q = \{Q_n\}_{n\in\mathbb{N}}$ of $F = \{F_n\}_{n\in\mathbb{N}}$, if it satisfies the following conditions.*

1. ***Constructivity.*** *The predicate "is f contained in Q_n" can be computed in polynomial time.*
2. ***Largeness.*** *$|Q_n| \geq \delta_n \cdot |F_n|$.*
3. ***Usefulness.*** *For any sequence of functions $f_n \in F_n$ (for $n \in \mathbb{N}$), if $f_n \in \Lambda$ then $f_n \notin Q_n$ almost everywhere.*

When Q_n satisfies these conditions we say that it is a natural property for Λ with density δ_n. In general, we fix $\delta_n = 1/2$.[3]

Now we will define the property that arises from Nisan's lower bounds for $\mathsf{MAJ} \circ \mathsf{THR}$-circuits [Nis93], and then demonstrate that it is natural. To the best of our knowledge, the natural property has never been explicitly formalized prior to this paper, but the fact that it is natural is essentially credited to the sequence of works [Nis93, CT93, Raz00, VW07]; [Nis93] provided the lower bound, and [CT93, Raz00, VW07] all implicitly proved it was natural. The property is defined by the following algorithm that classifies whether or not a truth table is in the property.

Definition 4.2 (Nisan's Natural Property)**.**

1. ***Input.*** *$T_{f_n} \in \{0,1\}^{2^n}$ a truth table of a function $f_n \in F_n$.*
2. *Choose any partition of the n inputs of f_n into two sets A, B with $|A| = \lceil n/2 \rceil$ and $|B| = \lfloor n/2 \rfloor$. Let $x_1||x_2$ denote the partitioned input according to A, B.*
3. *Given T_{f_n}, compute*

$$\alpha = 2^n \cdot R_2(f_n) = \sum_{\substack{x_1^0, x_1^1 \in \{0,1\}^{\lceil n/2 \rceil} \\ x_2^0, x_2^1 \in \{0,1\}^{\lfloor n/2 \rfloor}}} \prod_{b_1, b_2 \in \{0,1\}} f_n(x_1^{b_1} || x_2^{b_2})$$

4. ***Output.*** *If $|\alpha| \leq 2^{2n/3}$, print 1, otherwise print 0.*

Let $\mathsf{MAJ}_s \circ \mathsf{THR}$ denote the class of majority-of-thresholds circuits where the top majority gate has fan-in s.

Theorem 4.1 ([Nis93, CT93, Raz00, VW07])**.** *There is a constant c such that there is a natural property for $\mathsf{MAJ}_{2^{n/c}} \circ \mathsf{THR}$ with density 1/2.*

To prove this theorem, one uses the result of Nisan [Nis93]. The notation $\mathrm{r}\Pi[2, c, \gamma]$ denotes the set of functions $f : \{0,1\}^n \to \{-1, 1\}$ that have 2-party randomized communication complexity, with cost at most c and bias at least γ, for every partition of the inputs to the function. See Section 2 for formal definitions of communication complexity classes and protocols in this work.

Theorem 4.2 ([Nis93])**.** $\mathsf{MAJ}_s \circ \mathsf{THR} \subseteq \mathrm{r}\Pi[2, O(\log(s)), O(1/s)]$

Sketch. The statement follows from combining a few arguments in [Nis93]. First, Nisan proves that a single THR gate is contained in $\mathrm{r}\Pi[2, O(\log(s)), 1/2 - O(1/s)]$ (Theorem 1a.). Second, Nisan shows that the majority vote of any s functions contained in $\mathrm{r}\Pi[2, O(\log(s)), 1/2 - O(1/s)]$ is contained in $\mathrm{r}\Pi[2, O(\log(s)), O(1/s)]$ (Lemma 5). Therefore we get that $\mathsf{MAJ}_s \circ \mathsf{THR} \in \mathrm{r}\Pi[2, O(\log(s)), O(1/s)]$. ■

Now we prove Theorem 4.1.

Proof of Theorem 4.1. We prove each of the three necessary properties individually. First, observe that Definition 4.2 defines a polynomial time algorithm, in the size of the truth table (this proves the Constructivity property).

For the Largeness property, we will analyze the probability that the algorithm outputs 1 given a random truth table as input.

Consider $\alpha/2^{-n} = R_2(f_n)$. We have that

$$\alpha/2^{-n} = \mathop{\mathbb{E}}_{x_1^0, x_1^1, x_2^0, x_2^1}\left[\prod_{b_1, b_2 \in \{0,1\}} f_n(x_1^{b_1}, x_2^{b_2})\right]$$

[3]See Lemma 2.7 of [CIKK16] for an explanation for why this is reasonable.

This expected value over the uniformly random choice of $x_1^0, x_1^1, x_2^0, x_2^1$ is 0 except for when either $x_1^0 = x_2^0$ or $x_1^1 = x_2^1$. By a union bound, we have then that $\alpha/2^{-n} \leq 2^{1-(n+1)/2}$. Therefore, $\alpha \leq 2^{n/2+1}$, and the probability over a random truth table T_{f_n} that $\alpha \leq 2^{2n/3}$ is at least $1/2$. This proves density is at least $1/2$.

Finally, we prove Usefulness. By Theorem 2.1 and Theorem 4.2, we know that when T_{f_n} is the truth table of $f_n \in \mathsf{MAJ}_s \circ \mathsf{THR}$, then for $s \leq 2^{n/c}$ and some sufficiently large constant c, we have that $\alpha \geq 2^n/\text{poly}(2^{n/c}) \geq 2^{2n/3} + 1$. Thus, whenever $f_n \in \mathsf{MAJ}_{2^{n/c}} \circ \mathsf{THR}$, the algorithm defining the natural property outputs 0, as desired.

■

4.2 Proof of Theorem 1.1

In the last section, we demonstrated that $\mathsf{MAJ}_s \circ \mathsf{THR}$ has a dense natural property for s up to $2^{n/c}$ for some constant c. Now we will use this fact, together with hardness of learning $\mathsf{MAJ} \circ \mathsf{THR}$ in Valiant's PAC model [KS09] to prove Theorem 1.1.

Theorem 4.3 (Theorem 1.3 in [KS09])**.** *Assume that* $\mathsf{MAJ} \circ \mathsf{THR}$ *is PAC-learnable in time* $\text{poly}(n)$*. Then there is a polynomial time solution to* $\tilde{O}(n^{1.5})$-uSVP*, and polynomial time quantum solutions to* $\tilde{O}(n^{1.5})$-SVP *and* $\tilde{O}(n^{1.5})$-SIVP*.*

Theorem 4.4 (Theorem 1.1 restated)**.** *Let* $u : \mathbb{N} \to \mathbb{N}$*. Suppose that a natural proof against* $\mathsf{MAJ}_{u(n)} \circ \mathsf{THR}$*-circuits (and density 1/2) implies that* $\text{poly}(n)$ *size* $\mathsf{MAJ} \circ \mathsf{THR}$*-circuits are PAC-learnable in Valiant's model, in time* $\exp(u^{-1}(\text{poly}(n)))$*. Then, then there is a polynomial time solution to* $\tilde{O}(n^{1.5})$-uSVP*, and polynomial time quantum solutions to* $\tilde{O}(n^{1.5})$-SVP *and* $\tilde{O}(n^{1.5})$-SIVP*.*

Proof. By Theorem 4.1, there is a constant c such that there is a natural property for $\mathsf{MAJ}_{2^{n/c}} \circ \mathsf{THR}$ with density $1/2$. Thus, by the condition of the current theorem, we conclude that the class of $\mathsf{MAJ} \circ \mathsf{THR}$-circuits is PAC-learnable in Valiant's model. By Theorem 4.3, we then obtain a polynomial time solution to $\tilde{O}(n^{1.5})$-uSVP, and polynomial time quantum solutions to $\tilde{O}(n^{1.5})$-SVP and $\tilde{O}(n^{1.5})$-SIVP. ■

4.2.1 Hardness of DistPAC-Learning for Majority of Threshold Circuits for all Target Distributions

Corollary 4.1. *There exists a polynomial time samplable distribution* μ *over polynomial size* $\mathsf{MAJ} \circ \mathsf{THR}$*-circuits, such that if* $(\mathsf{MAJ} \circ \mathsf{THR}, \mu)$ *is efficiently distPAC-learnable, then there is a polynomial time solution to* $\tilde{O}(n^{1.5})$-uSVP*, and polynomial time quantum solutions to* $\tilde{O}(n^{1.5})$-SVP *and* $\tilde{O}(n^{1.5})$-SIVP*.*

Proof. Implicitly from Theorem 4.3, we know that the decryption function of Regev's public key cryptosystem can be implemented by a $\mathsf{MAJ} \circ \mathsf{THR}$-circuit, with negligible decryption error. Therefore, by Theorem 3.4, the statement follows. ■

4.2.2 A Similar Barrier via Hardness of Learning DNFs

A similar statement to Theorem 4.4 can be observed for DNFs using a different hardness of PAC-learning result of [DSS16]. Specifically, [DSS16] prove that PAC-learning DNFs in Valiant's model implies the ability to refute random k-SAT instances. Feige [Fei02] first introduce hardness assumptions regarding refuting k-SAT instances.

An algorithm refutes random k-SAT instances with $c(n) \geq \Omega(n)$ clauses if on $1 - o(1)$ fraction of the k-SAT formulas with $c(n)$ constraints, it outputs "unsatisfiable", while whenever it encounters a satisfiable k-SAT formula with $c(n)$ constraints, it outputs "satisfiable." A random K-SAT instance $I = \{C_1, \cdots, C_{c(n)}\}$ is sampled in the way that each clause/constraint C_i is chosen uniformly at random from the set of k-SAT clauses/constraints with n variables.

Thus, it follows directly from the assumption regarding hardness of refuting random k-SAT formulas from [DSS16] that there is no implication from a Natural Proof for exponential size DNFs, to PAC-learning for DNFs. This is because Nisan's natural proof clearly works for exponential size DNFs, since the set is clearly contained by the set of exponential size majority-of-threshold circuits.

5 Main Theorem

In this section, we will prove Theorem 1.2, which obtains distPAC-learning algorithms for concept classes that have low-cost associated communication games. To do so, we will first define the communication game, and then obtain a weak distPAC-learning algorithm (see Definition 3.1). Finally, we will conclude Theorem 1.2 using the equivalence between weak and strong distPAC-learning (see Theorem 3.1).

In the following sections, we will use Theorem 1.2 to derive distPAC-learning algorithms for natural distributions over $\mathsf{MAJ} \circ \mathsf{THR}$-circuits, polytopes, and DNFs, and impossibility results for weak PRFs that can be evaluated with $\mathsf{MAJ} \circ \mathsf{THR}$-circuits with an arbitrary input encoding.

5.1 Communication Games

Recall that $\mathfrak{C} = \{\mathfrak{C}_n\}_{n\in\mathbb{N}}$ is the $s(n)$-represented Boolean concept class that is induced by the evaluation function $\phi = \{\phi_n\}_{n\in\mathbb{N}}$. We define the communication game associated with $\mathfrak{C}$:

Definition 5.1 (2-party distributional communication game)**.** *With respect to a evaluation rule ϕ and product distribution (μ, ρ), the 2-party communication game $\mathfrak{G}[\phi, n, (\mu, \rho)]$ is the following:*

- *Setup: $\pi_f \sim \mu, x \sim \rho$.*
- *Player 1 gets as input $\pi_f \in \{0,1\}^{s(n)}$ for concept $f \in \mathfrak{C}_n$.*
- *Player 2 gets as input a string $x \in \{0,1\}^n$.*
- *The object of the game is for the parties to output the value $\phi_n(\pi_f, x) = f(x)$, using as few bits of communication as possible.*

We say that $\mathfrak{G}[\phi, n, (\mu, \rho)]$ is (c, γ)-evaluated if the parties can communicate at most c bits, and win the game with probability $1/2 + \gamma$ (over the random sample of inputs according to (μ, ρ)).

Other definitions. We direct the reader to Section 2 for the necessary definitions of communication complexity.

5.2 Weak Learning

Towards Theorem 1.2, we will start by first obtaining a weak learning algorithm, which only requires prediction accuracy marginally better than a coin toss.

Notation. In the following, we discuss boolean functions $f : I^n \to \{-1, 1\}$, and denote by U_n the uniform distribution over I^n. For shorthand, we will write $c := c(n), \gamma := \gamma(n)$, to denote number of bits of communication and protocol bias, which are dependent on n, the input length of a concept.

Also, in the rest of the paper we will streamline notation by eliding the subscripts on distributions coming from ensembles indexed by $n \in \mathbb{N}$.

Theorem 5.1. *Let ϕ be an evaluation rule. Suppose that, for every $n \in \mathbb{N}$, and product distribution (μ, ρ), $\mathfrak{G}[\phi, n, (\mu, \rho)]$ is (c, γ)-evaluated. Then there exists an algorithm A such that, for any $n \in \mathbb{N}, \delta, \eta > 0$,*

$$\Pr_{f\sim\mu}\left[\Pr_{A}\left[\forall\rho : \Pr_{x\sim\rho}\left[h(x) \neq f(x) : h \leftarrow A^{\mathrm{Ex}(f,\rho)}(n, \delta, \eta)\right] \leq \frac{1}{2} - \mathrm{poly}(\gamma \cdot 2^{-c})\right] \geq 1 - \delta\right] \geq 1 - \eta$$

For μ samplable in time $t(n)$, A runs in time $\mathrm{poly}(n, t(n), s(n), \gamma^{-1}, \delta^{-1}, 2^c)$.

Proof. To construct A, we will follow three steps:

1. Construct a weak randomized predictor L.
2. Argue that many good non-uniform but deterministic predictors exist, by fixing coins and samples for L.
3. Construct a deterministic predictor by sampling and then testing enough non-uniform predictors.

Steps 2 and 3 follow from standard techniques (i.e., "constructive averaging").

Claim 5.1 (Weak randomized predictor). *Let ϕ be an evaluation rule. Under the conditions of Theorem 5.1, there exists a randomized algorithm L, running in time* $\mathrm{poly}(n, t(n), s(n), \gamma^{-1}, \delta^{-1}, \eta^{-1}, 2^c)$, *such that for any $n \in \mathbb{N}, \delta, \eta > 0$, the following equation is satisfied:*

$$\Pr_{f\sim\mu}\left[\Pr_{L}\left[\forall\rho : \Pr_{z\sim\rho}\left[L^{\mathrm{Ex}(f,\rho)}(z,n,\delta,\eta) \neq f(z)\right] \leq \frac{1}{2} - \mathrm{poly}(\gamma\cdot 2^{-c})\right] \geq 1-\delta\right] \geq 1-\eta \qquad (11)$$

Proof of Claim 5.1. We will abuse notation and write $\pi_g \sim \mu$ to denote the binary representation of a concept g, distributed appropriately according to the target distribution μ over the ϕ-induced $s(n)$-represented concept class $\mathfrak{C}$. See the randomized predictor L in Figure 1.

Consider the distribution $\mathcal{M}$ over 2×2 matrices

$$C = \begin{bmatrix} \phi(\pi_f, z) & \phi(\pi_f, w) \\ \phi(\pi_g, z) & \phi(\pi_g, w) \end{bmatrix}$$

where $\pi_f, \pi_g \sim \mu$ and $z, w \sim \rho$. We now claim that, under the conditions of Theorem 5.1, this distribution is efficiently *distinguishable* from the distribution $\mathcal{R}$ over random 2×2 matrices,

$$R = \begin{bmatrix} r_{00} & r_{01} \\ r_{10} & r_{11} \end{bmatrix}$$

To see this, observe that the distribution over C is identical to the following distribution over 2×2 matrices (abusing notation, $z = \rho(y)$ denotes a point z sampled according to ρ with the random bits y)

$$D = \begin{bmatrix} \phi(\mu(x), \rho(y)) & \phi(\mu(x), \rho(y')) \\ \phi(\mu(x'), \rho(y)) & \phi(\mu(x'), \rho(y')) \end{bmatrix}$$

Here, x, x', y, y' are uniformly random strings. We can assume that without loss of generality that x, x', y, y' are all the same length, by defaulting to the maximum necessary length for sampling μ, ρ (and padding the shorter strings with useless random bits). Therefore, identifying $\xi(x,y) := \phi(\mu(x), \rho(y))$, we can now see that

$$R_2(\xi) = \mathop{\mathbb{E}}_{\substack{\pi_f,\pi_g \\ z,w}}\left[\prod_{i,j\in\{0,1\}} C_{ij}\right]$$

It now readily follows that when $\mathfrak{G}[\phi, n, (\mu, \rho)]$ is (c, γ)-evaluated (which is true by assumption), then

$$R_2(\xi) = \mathop{\mathbb{E}}_{\substack{\pi_f,\pi_g \\ z,w}}\left[\prod_{i,j\in\{0,1\}} C_{ij}\right] \geq (\gamma\cdot 2^{-c})^4 \qquad (12)$$

On the other hand,

$$\mathop{\mathbb{E}}_{R}\left[\prod_{i,j\in\{0,1\}} R_{ij}\right] = 0$$

Algorithm 1 $L^{\mathrm{Ex}(f,\rho)}$

```
1: Input: $z \in I^n, n \in \mathbb{N}, \delta, \eta > 0$
2: Pick uniformly random values $b_1, b_2 \in \{0,1\}$.
3: Pick uniformly random values $r_{00}, r_{01}, r_{10}, r_{11} \in \{-1,1\}$
4: Sample $\pi_g \sim \mu$.
5: Sample $(w,y) \sim \mathrm{Ex}(f,\rho)$.
6: if $b_1 = b_2 = 0$ then
7:     $v \leftarrow \prod_{i,j\in\{0,1\}} r_{ij}$
8: if $b_1 = 0, b_2 = 1$ then
9:     $v \leftarrow y \cdot r_{00} \cdot \prod_{i,j\in\{0,1\}} r_{ij}$
10: if $b_1 = 1, b_2 = 0$ then
11:     $v \leftarrow \phi(\pi_g, w) \cdot \phi(\pi_g, z) \cdot \prod_{j\in\{0,1\}} r_{1j}$
12: if $b_1 = 1, b_2 = 1$ then
13:     $v \leftarrow y \cdot \phi(\pi_g, w) \cdot \phi(\pi_g, z) \cdot r_{11}$
14: $b \leftarrow r_{b_1 b_2}$
15: Output $b \cdot v$
```

Figure 1: Randomized predictor L.

$$H_1 = \mathcal{R}$$

$$H_2 = \begin{cases} C_{k\ell} \text{ when } k = 0, \ell = 0 \\ R_{k\ell} \text{ otherwise} \end{cases}$$

$$H_3 = \begin{cases} C_{k\ell} \text{ when } k = 0, \ell \leq 1 \\ R_{k\ell} \text{ otherwise} \end{cases}$$

$$H_4 = \begin{cases} C_{k\ell} \text{ when } k = 0 \text{ or } \ell \leq 0 \\ R_{k\ell} \text{ otherwise} \end{cases}$$

$$H_5 = \mathcal{M}$$

Figure 2: Hybrid sequence.

Now that we have established this, we may proceed by a hybrid argument. Define the neighboring hybrid distributions H_1, H_2, H_3, H_4, H_5 over 2×2 matrices, as in Figure 2.

It then follows that for random hybrid neighbors H_i, H_{i+1} ($i \in [4]$),

$$\mathop{\mathbb{E}}_{i \sim [4]} \left[\mathop{\mathbb{E}}_{H' \sim H_{i+1}} \left[\prod_{k,j \in \{0,1\}} H'_{kj} = 1 \right] - \mathop{\mathbb{E}}_{H \sim H_i} \left[\prod_{k,j \in \{0,1\}} H_{kj} = 1 \right] \right] \geq (\gamma \cdot 2^{-c})^4/4 \tag{13}$$

To ease notation, let $D(H) = \prod_{k,j \in \{0,1\}} H_{kj}$, and let V_i denote the event that $D(H_i) = 1$. Intuitively, the function D stands for "distinguisher," and can be thought of as such.

We continue by observing that, by definition, the value stored as v in L (Algorithm 1) is $D(H_i)$ for a random $i \in [4]$. Hence, the output of L, which is written as $D(H_i) \cdot b$, is interpreted as a prediction, where $b = r_{b_1 b_2}$ is the "guess bit." Note that, the string $b_1 b_2$ is the binary representation of i.

Now, conditioning on correctness of this guess bit, we have that for all ρ, and probabilities taken over $z \sim \rho$, $f \sim \mu$ and the randomness of L:

$$\begin{aligned} \Pr\left[L^{\mathrm{Ex}(f,\rho)}(z,n,\delta,\eta) = f(z)\right] &= \Pr\left[L^{\mathrm{Ex}(f,\rho)}(z,n,\delta,\eta) = f(z) \mid b = f(z)\right] \cdot \Pr[b = f(z)] \\ &\quad + \Pr\left[L^{\mathrm{Ex}(f,\rho)}(z,n,\delta,\eta) = f(z) \mid b \neq f(z)\right] \cdot \Pr[b \neq f(z)] \\ &= \frac{1}{2}\Big(\Pr[b \cdot D(H_i) = f(z) \mid b = f(z)] \\ &\quad + \Pr[b \cdot D(H_i) = f(z) \mid b \neq f(z)]\Big) \end{aligned}$$

Indeed, when V_i is unsatisfied, this means that the output of L is b. The case analysis follows:

$$\begin{aligned} \Pr[L^{\mathrm{Ex}(f,\rho)}(z,n,\delta,\eta) = f(z)] &= \frac{1}{2}\Big(\Pr[V_i \mid b = f(z)] + \Pr[\neg V_i \mid b \neq f(z)]\Big) \\ &= \frac{1}{2} + \frac{1}{2}\Big(\Pr[V_i \mid b = f(z)] - \Pr[V_i \mid b \neq f(z)]\Big) \end{aligned}$$

By conditioning, we know that:

$$\Pr[V_i] = \frac{1}{2}\Pr[V_i \mid b = f(z)] + \frac{1}{2}\Pr[V_i \mid b \neq f(z)]$$

rearranging the terms, we get:

$$\frac{1}{2}\Pr[V_i \mid b \neq f(z)] = \Pr[V_i] - \frac{1}{2}\Pr[V_i \mid b = f(z)]$$

We thus conclude:

$$\Pr[L^{\mathrm{Ex}(f,\rho)}(z,n,\delta,\eta) = f(z)] = \frac{1}{2} + \underbrace{\Pr[V_i \mid b = f(z)]}_{(\alpha)} - \underbrace{\Pr[V_i]}_{(\beta)}$$

The term (α) corresponds to the case that L computes the 2-party norm on a sample from H_{i+i} (i.e., the product of the entries of a matrix sampled from H_{i+i}), while term (β) is the case that L computes the 2-party norm on a sample from H_i (the product of the entries of a matrix sampled from H_i). Thus, by equation (13),

$$\begin{aligned} \Pr[L^{\mathrm{Ex}(f,\rho)}(z,n,\delta,\eta) = f(z)] &= \frac{1}{2} + (1 - \Pr[D(H_{i+1}) = 1] - (1 - \Pr[D(H_i) = 1]) \\ &\geq \frac{1}{2} + (\gamma \cdot 2^{-c})^4/8 \end{aligned}$$

■

Algorithm 2 $A^{\mathrm{Ex}(f,\rho)}$

```
1: Input: $n \in \mathbb{N}, \delta \in (0,1]$
2: Sample $m$ (sufficiently many) candidate circuits, using oracle access to $\mathrm{Ex}(f,\rho)$ as needed.
3: Sample $t$ (sufficiently many) additional random examples from $\mathrm{Ex}(f,\rho)$.
4: for each sampled circuit $C_i$, do
5:     Compute using random examples: $\alpha_i \leftarrow \mathrm{Cor}(f, C_i)$
6: output the circuit with largest $\alpha$ value.
```

Figure 3: Algorithm for sampling and testing candidate predictors.

Having established Claim 5.1, we now convert the randomized algorithm $L^{\mathrm{Ex}(f,\rho)}$ into a non-uniform learning algorithm by averaging. Let $L^{\mathrm{Ex}(f,\rho)}_{\mathrm{inp},i}(z;r)$ denote the Algorithm 1 where the random hybrid choice is fixed to be i, and the input parameters $\mathrm{inp} = (n, \delta, \eta)$ are hard-wired in, and the random bits r for computing other randomized aspects of the algorithm is treated as input. This allows us to consider the algorithm as a deterministic mapping of random bits and examples from $\mathrm{Ex}(f,\rho)$ to a circuit that weakly agrees with f. By a standard averaging argument, we obtain:

Claim 5.2 (Averaging, see lemma A.11 of [AB09])**.**

$$\Pr_r\left[\Pr_{z\sim\rho}\left[L^{\mathrm{Ex}(f,\rho)}_{\mathrm{inp},i}(z;r) = f(z) \mid r\right] > \frac{1}{2} + \frac{(\gamma\cdot 2^{-c})^4}{32}\right] > (\gamma\cdot 2^{-c})^4/4$$

Taking hybrid index i and r uniformly at random, we obtain "good" choices with good probability. Therefore such a circuit is efficiently found by randomized trial-and-error; we sample many candidate predictors in parallel and then compare each to the concept by checking random examples. By a standard application of Chernoff bounds, sufficiently many examples will be enough to check that a circuit with good enough accuracy is indeed good enough, with high probability.

Claim 5.3 (Without proof)**.** *With probability $1-\delta$, A (Algorithm 2 in Figure 3) outputs a "good" circuit that correctly classifies $1/2 + \mathrm{poly}(\gamma \cdot 2^{-c})$ fraction of points, where t, m are quantities that are polynomially bounded as a function of $\mathrm{poly}(\gamma\cdot 2^{-c})$ and $\log(\delta^{-1})$.*

From the above claims it now follows, from a Markov argument, that:

$$\Pr_{\pi_f\sim\mu}\left[\Pr_A\left[\forall\rho : \Pr_{z\sim\rho}\left[h(z)\neq f(z) : h\leftarrow A^{\mathrm{Ex}(f,\rho)}(n,\delta)\right] \le \frac{1}{2} - \mathrm{poly}(\gamma\cdot 2^{-c})\right] \ge 1-\delta/\eta\right] \ge 1-\eta$$

This concludes the proof of Theorem 5.1.

■

Remark. Using the 2-party norm as we do is a *universal* distinguisher. That is, it distinguishes any evaluation rule that is (c,γ)-evaluated from a random function (using the lower bound of $(\gamma\cdot 2^{-c})^4$). Therefore it holds that for *arbitrary* choice of distribution ρ, we obtain the desired guarantee. Indeed, the choice of ρ can be adversarial with respect to $f \sim \mu$, and it never needs to be known by A.

5.3 Main Theorem

Theorem 5.1 is enough to prove Theorem 1.2. Recall that $\mathfrak{C} = \{\mathfrak{C}_n\}_{n\in\mathbb{N}}$ is a boolean concept class that is $s(n)$-representable by the evaluation function $\phi = \{\phi_n\}_{n\in\mathbb{N}}$.

Theorem 5.2 (Theorem 1.2, restated)**.** *Let ϕ be an evaluation rule, and suppose that for every $n\in\mathbb{N}$, and product distribution (μ,ρ), $\mathfrak{G}[\phi, n, (\mu,\rho)]$ is (c,γ)-evaluated. Then, $(\mathfrak{C},\mu)$ is distPAC-learnable. For a time $t(n)$-samplable μ and $s(n)$-represented $\mathfrak{C}$, the learning algorithm runs in time polynomial in $n, t(n), s(n), \gamma^{-1}, \varepsilon^{-1}, \delta^{-1}, \eta^{-1}$, and 2^c.*

Proof. Immediate from Theorem 3.1 and Theorem 5.1. ■

Remark. The exponential dependency of c is likely necessary, since the theorem does not restrict $\mathfrak{C}$ (e.g., it can be $\mathsf{P/poly}$), and no matter what $c \leq n$.

6 Distributional PAC-Learning from Nisan's Natural Proofs

In this section, we will apply Theorem 5.2 to obtain polynomial time distPAC-learning algorithms. Theorem 6.1 restates Theorem 1.3.

Theorem 6.1 (distPAC-learning — Theorem 1.3 restated)**.** *Let $\phi \in \mathsf{MAJ} \circ \mathsf{THR}$ be any evaluation rule, and let μ be any polynomial time samplable target distribution. Then, for the ϕ-induced $s(n)$-represented concept class $\mathfrak{C}$, the pair $(\mathfrak{C}, \mu)$ is efficiently distPAC-learnable.*

Proof of Theorem 6.1. By Theorem 4.2, $\mathsf{MAJ} \circ \mathsf{THR}$ has a randomized communication protocol with cost $O(\log n)$ and bias at least $1/\text{poly}(n)$. This implies that for any product distribution (μ, ρ) it is the case that $\mathfrak{G}[\phi, n, (\mu, \rho)]$ is $(O(\log n), 1/\text{poly}(n))$-evaluated, since $\phi \in \mathsf{MAJ} \circ \mathsf{THR}$. This is enough to conclude the theorem by Theorem 5.2. ■

6.1 DistPAC-Learning on Natural Target Distributions

6.1.1 Majority-of-Thresholds

In this section, we will use Theorem 6.1 to show efficient distPAC-learning algorithms for $\mathsf{MAJ} \circ \mathsf{THR}$-circuits over natural target distributions. Recall that, as mentioned in the introduction, we must restrict the target distribution over polynomial size $\mathsf{MAJ} \circ \mathsf{THR}$-circuits if we hope to obtain efficient distPAC-learning without also obtaining polynomial time solutions to $\tilde{O}(n^{1.5})$-uSVP, and polynomial time quantum solutions to $\tilde{O}(n^{1.5})$-SVP and $\tilde{O}(n^{1.5})$-SIVP (see Corollary 4.1 for the formal statement).

The target distribution. Let $L = (T_1, \cdots T_m)$ be a list of $m := \text{poly}(n)$ linear threshold functions. Also, let μ be any $\text{poly}(n)$ time samplable distribution over $\{0,1\}^m$. We define the distribution μ_L over $\mathsf{MAJ} \circ \mathsf{THR}$-circuits as follows.

- Sample $\theta \sim \mu$.
- Output the $\mathsf{MAJ} \circ \mathsf{THR}$-circuit that is the majority vote over each $T_i \in L$ such that $\theta_i = 0$.

We will show that, for any μ, L as described above, $(\mathsf{MAJ} \circ \mathsf{THR}, \mu_L)$ is efficiently distPAC-learnable.

Theorem 6.2. *Let $L = (T_1, \cdots T_m)$ be any list of $m := \text{poly}(n)$ linear threshold functions over n-bit inputs, and let μ be any $\text{poly}(n)$ time samplable distribution over $\{0,1\}^m$. The pair $(\mathsf{MAJ} \circ \mathsf{THR}, \mu_L)$ is efficiently distPAC-learnable.*

Proof. To prove the theorem, we will show that the target distribution μ_L can be translated into a distribution μ_L^* over binary representations, so that given a representation $\pi_f \sim \mu_L^*$, there is an evaluation rule $\phi \in \mathsf{MAJ} \circ \mathsf{THR}$ such that, for every $x \in \{0,1\}^n$, it holds that $\phi(\pi_f, x) = f(x)$. In other words, the distribution over functions $\phi(\pi_f, \cdot)$ for $\pi_f \sim \mu_L^*$ is identical (when considering functional equivalence) to μ_L. This suffices to prove the theorem by invoking Theorem 5.2.

We construct ϕ in the following way. For $T_i \in L = (T_1, \cdots T_m)$, let $\theta_i \in \mathbb{Z}$ be the threshold parameter and let $w_{i,j} \in \mathbb{Z}$ be the j^{th} weight in T_i (i.e., $T_i = [w_{i,1}x_1 + \cdots + w_{i,n}x_n \geq \theta_i]$). Now, for every $T_i \in L$, add an input variable z_i that is weighted by $w_{z_i} = -(w_{i,1} + \cdots + w_{i,n} + \theta_i + 1)$. Call the new threshold function $T_i' = [w_{i,1}x_1 + \cdots + w_{i,n}x_n + w_{z_i}z_i \geq \theta_i]$

Also, define auxiliary variables $y_1 \cdots y_m$. We define $\phi : \{0,1\}^{2m} \times \{0,1\}^n \to \{-1,1\}$

$$\phi(z_1 \cdots z_m, y_1 \cdots y_m, x) := \mathsf{MAJ}(T_1'(x, z_1), \cdots, T_m'(x, z_m), y_1, \cdots, y_m)$$

We now define the distribution μ_L^* over $\{0,1\}^{2m}$. To sample μ_L^*, first sample $z \sim \mu$. Then, letting $|z|$ denote the number of ones in z, sample $y = y_1 \cdots y_m$ by choosing uniformly at random amongst the m-bit strings such that $|y| = (|z| + m)/2$. Finally, output $(z, y) \in \{0,1\}^{2m}$.

It only remains to show that the distribution over functions $\phi(\pi_f, \cdot)$ for $\pi_f := (z, y) \sim \mu_L^*$ is identical (when considering functional equivalence) to μ_L. To see this, observe that whenever $z_i = 1$, $T_i'(x, z_i) = 0$. This follows from the fact that $-w_{z_i} > w_{i,1} + \cdots + w_{i,n} + \theta_i$. On the other hand, clearly when $z_i = 0$, $T_i'(x, z_i) = T_i(x)$.

The goal is to output $\mathsf{MAJ}_i(T_i(x))$ for all T_i such that $z_i = 0$. Therefore, we need to balance the presence of the extra zero votes in $\mathsf{MAJ}(T_1'(x, z_1), \cdots, T_m'(x, z_m))$, which occur whenever $z_i = 1$, by adding $|z|$ one votes. Since we took $|y| = (|z| + m)/2$, then $|y| - (m - |y|) = |z|$, so adding $y_1, \cdots, y_m$ to the majority vote gives a surplus of $|z|$ one votes.

Therefore, we can see that the distribution over functions $\phi(z_1 \cdots z_m, y_1 \cdots y_m, \cdot)$ for $(z, y) \sim \mu_L^*$ is, when considering functional equivalence, identical to μ_L. Since μ_L^* is obviously polynomial time samplable, and $\phi \in \mathsf{MAJ} \circ \mathsf{THR}$, the theorem statement follows by Theorem 5.2. ■

Non-black-box access to μ_L. The distPAC-learning algorithm does not need descriptions of μ or L as input. Instead, it suffices to have black-box access to a sampling machine that outputs appropriately distributed polynomial size circuits that are functionally equivalent to the appropriate $\mathsf{MAJ} \circ \mathsf{THR}$-circuit, under some fixed encoding scheme. To see this, observe that the randomized predictor L (Figure 1) uses the evaluation rule ϕ and a concept representation π_g to basically obtain some labels of g. But it is not important what the representation is, or how ϕ is implemented. For the accuracy of the learning algorithm, it only matters that, it is *possible* to implement ϕ by a $\mathsf{MAJ} \circ \mathsf{THR}$ circuit (under some concept representation). Hence, the learner can work with some far messier representation of concepts, such as by arbitrary polynomial size circuits.

The fact that the learner does not need to know μ or L is a good property that enhances the convenience of the learning algorithm. Arguably, knowledge of μ and L would be a prohibitive assumption in practice; rather, black-box sample access to a more complex and messy form of μ_L (provided that the learner can still interpret the encoding well enough to evaluate it) is a significantly weaker assumption, corresponding to a scenario where the learner has some non-explicit knowledge about the possible concepts it may encounter.

6.2 Polytopes and DNFs

In this section we will explain how we can slightly modify the proof for Theorem 6.2 to also obtain distPAC-learning for natural distributions over polytopes and DNFs. Recall that, in distributional PAC-learning, subclasses are not necessarily distPAC-learnable if their superclass is, since it is possible that the subclass consists of functions that are hard for the superclass distPAC-learning algorithm.

A polytope is an intersection of linear threshold functions. In other words, it is any function that belongs to the class $\mathsf{AND} \circ \mathsf{THR}$. Therefore, it is easy to see that any polytope is computable by a $\mathsf{MAJ} \circ \mathsf{THR}$-circuit. We consider a slight modification of the distribution over $\mathsf{MAJ} \circ \mathsf{THR}$-circuits:

Polytope target distribution. Let $L = (T_1, \cdots T_m)$ be a list of $m := \mathrm{poly}(n)$ linear threshold functions. Also, let μ be any $\mathrm{poly}(n)$ time samplable distribution over $\{0,1\}^m$. We define the distribution $\mu_L^\wedge$ over polytopes as follows.

- Sample $\theta \sim \mu$.
- Output the polytope that is the AND over each $T_i \in L$ such that $\theta_i = 0$.

Theorem 6.3. *Let $L = (T_1, \cdots T_m)$ be any list of $m := \mathrm{poly}(n)$ linear threshold functions over n-bit inputs, and let μ be any $\mathrm{poly}(n)$ time samplable distribution over $\{0,1\}^m$. The pair $(\mathsf{AND} \circ \mathsf{THR}, \mu_L^\wedge)$ is efficiently distPAC-learnable.*

Proof. To complete the proof, we only modify how the evaluation rule $\phi : \{0,1\}^{3m} \times \{0,1\}^n \to \{-1,1\}$ is defined in the proof of Theorem 6.2:

$$\phi(v_1, \cdots v_m, z_1 \cdots z_m, y_1 \cdots y_m, x) := \mathsf{MAJ}(T_1'(x, z_1), \cdots, T_m'(x, z_m), y_1, \cdots, y_m, v_1, \cdots v_m)$$

We define a distribution μ_L^{**} over binary representations, which samples $(z, y) \sim \mu_L^*$, and then v is sampled uniformly at random from the set of m-bit strings with $|v| = |z|/2$. This means that $(m - |v|) - |v| = m - |z|$, that is, there is a surplus of $|m| - z$ zero votes within v. We do this in order to convert the top majority-gate into an and-gate, which is accomplished by adding the surplus of $m - |z|$ zero votes, because this forces every T_i for $z_i = 0$ (of which there are $m - |z|$ of them) to evaluate to 1, in order for ϕ to evaluate to 1, as desired.

Therefore, we can see that the distribution over functions $\phi(v, z, y, \cdot)$ for $(v, z, y) \sim \mu_L^{**}$ is, when considering functional equivalence, identical to $\mu_L^{\wedge}$. Since μ_L^{**} is obviously polynomial time samplable, and $\phi \in \mathsf{MAJ} \circ \mathsf{THR}$, the theorem statement follows by Theorem 5.2. ■

A DNF can be viewed as a polytope where each of the linear threshold functions simulate OR gates. One can simulate OR gates by restricting the weights to be either 0 or 1, and setting the threshold $\theta := 1$. We therefore consider a modification of the distribution over polytopes:

DNF target distribution. Let $L = (T_1, \cdots T_m)$ be a list of $m := \text{poly}(n)$ linear threshold functions, where each weight is either 0 or 1 and the threshold is fixed to $\theta := 1$. In other words, each T_i implements an OR of some selection of variables. Also, let μ be any $\text{poly}(n)$ time samplable distribution over $\{0,1\}^m$. We define the distribution $\mu_L^{\wedge\vee}$ over DNFs as follows.

- Sample $\theta \sim \mu$.
- Output the DNF that is the AND over each $T_i \in L$ such that $\theta_i = 0$.

Theorem 6.4. *Let $L = (T_1, \cdots T_m)$ be a list of $m := \text{poly}(n)$ linear threshold functions over n-bit inputs, where each weight is either 0 or 1 and the threshold is fixed to $\theta := 1$. Also, let μ be any $\text{poly}(n)$ time samplable distribution over $\{0,1\}^m$. The pair $(\text{DNF}, \mu_L^{\wedge\vee})$ is efficiently distPAC-learnable.*

Proof. Using the same argument as in the proof of Theorem 6.3, we can define

$$\phi(v_1, \cdots v_m, z_1 \cdots z_m, y_1 \cdots y_m, x) := \mathsf{MAJ}(T_1'(x, z_1), \cdots, T_m'(x, z_m), y_1, \cdots, y_m, v_1, \cdots v_m)$$

Then, because the threshold functions on the bottom layer are defined to simulate OR gates, the theorem now follows by the same arguments following the definition of the evaluation rule in Theorem 6.3. ■

7 Impossibility of Encoded-Input Weak PRFs

In this section, we will apply Theorem 5.2 to prove that weak PRFs augmented with a keyless encoding procedure cannot exist, when evaluating the encoded input on any key can be done by a $\mathsf{MAJ} \circ \mathsf{THR}$-circuit.

7.1 Encoded-Input Weak PRFs

We define encoded-input weak PRFs. Our definition is the natural relaxation of the definition of encoded-input *strong* PRFs of Boneh et al. [BIP+18].

Definition 7.1 (Encoded-input weak PRFs)**.** *Let λ be a security parameter, and $n := n(\lambda), \kappa := \kappa(\lambda), m := m(n)$ for polynomially bounded functions n, κ, m. Consider a trio of algorithms $\mathsf{f} : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}, \mathsf{gen} : \{1\}^\lambda \to \{0,1\}^\kappa, \mathsf{enc} : \{0,1\}^m \to \{0,1\}^n$.*

- gen *is a polynomial time sampling algorithm that given input parameter λ in unary and access to random coins $z \in \{0,1\}^{\text{poly}(\kappa)}$ outputs a key $k \in \{0,1\}^\kappa$.*

- enc *is a polynomial time algorithm that given a representation of an input* $r \in \{0,1\}^m$*, outputs an encoding* $x \in \{0,1\}^n$*.*
- f *is a polynomial time algorithm that given a key* k *and the* <u>*encoded*</u> *input* $x \in \{0,1\}^n$*, outputs a value* $\mathsf{f}(k,x) = v \in \{0,1\}$*.*

For $t = t(\lambda), \varepsilon = \varepsilon(\lambda)$*, we say that* $(\mathsf{f}, \mathsf{gen}, \mathsf{enc})$ *is a* (t,ε)*-e.i.weak PRF if, for every size* t *oracle circuit* C*,*

$$\left| \Pr_{k \sim \mathsf{gen}(1^\lambda)} \left[C^{\mathrm{Ex}(\mathsf{f})} = 1 \right] - \Pr_{\psi} \left[C^{\mathrm{Ex}(\psi)} = 1 \right] \right| \leq \varepsilon(\lambda)$$

where $\psi : \{0,1\}^n \to \{0,1\}$ *is a uniformly random function, and* $\mathrm{Ex}(h)$ *is an oracle that returns random points* $\langle \mathsf{enc}(r), h(\mathsf{enc}(r)) \rangle$ *for some function* $h : \{0,1\}^n \to \{0,1\}$ *and* $r \sim U_m$*.*

To measure the complexity of an e.i.weak PRF, we say that $(\mathsf{f}, \mathsf{gen}, \mathsf{enc})$ is *evaluated* by a uniform circuit class Λ, if $\mathsf{f} \in \Lambda$ (i.e., we ignore complexity of the encoding procedure). This differs from the notion of fixed-key complexity considered elsewhere in the literature for weak PRFs (where we want that, for every k, $\mathsf{f}(k,\cdot) \in \Lambda$). However, we remark that in some cases, arguably it is more important to know what is the circuit complexity of a single circuit that is capable of evaluating many inputs on many unknown keys.

7.2 Impossibility of Encoded-Input Weak PRFs Evaluated By $\mathsf{MAJ} \circ \mathsf{THR}$

In the remainder of this section, we will show that there cannot be any e.i.weak PRFs evaluated by $\mathsf{MAJ} \circ \mathsf{THR}$-circuits of polynomial size. The proof is a straightforward application of Theorem 5.2.

Theorem 7.1 (Theorem 1.7 restated)**.** *There exists no encoded-input weak PRF that is evaluated by a* $\mathsf{MAJ} \circ \mathsf{THR}$*-circuit.*

Proof. Let ϕ be an evaluation rule, and suppose that for every $n \in \mathbb{N}$, and product distribution (μ, ρ), $\mathfrak{G}[\phi, n, (\mu, \rho)]$ is (c, γ)-evaluated. By Theorem 5.2, $(\mathfrak{C}, \mu)$ is distPAC-learnable. For a poly(n)-samplable μ and poly(n)-represented $\mathfrak{C}$, the learning algorithm runs in time polynomial in $n, \gamma^{-1}, \varepsilon^{-1}, \delta^{-1}, \eta^{-1}$, and 2^c.

Now, consider any trio of algorithms, $(\mathsf{f}, \mathsf{gen}, \mathsf{enc})$, with $f \in \Lambda$. In order to apply Theorem 5.2, we view $\mathsf{f} \in \Lambda$ as an evaluation rule, gen as a poly(n)-samplable target distribution μ_{gen}, and enc as a sampling algorithm for an example distribution ρ_{enc}. It follows then that if for the product distribution $(\mu_{\mathsf{gen}}, \rho_{\mathsf{enc}})$, $\mathfrak{G}[\mathsf{f}, n, (\mu_{\mathsf{gen}}, \rho_{\mathsf{enc}})]$ is (c,γ)-evaluated, then $(\mathsf{f}, \mathsf{gen}, \mathsf{enc})$ cannot be a e.i.weak PRF with better than poly(2^c) security.

Finally, it follows that for any $\mathsf{f} \in \mathsf{MAJ} \circ \mathsf{THR}$, the trio $(\mathsf{f}, \mathsf{gen}, \mathsf{enc})$ cannot be an e.i.weak PRF, because we know that $\mathfrak{G}[\mathsf{f}, n, (\mu_{\mathsf{gen}}, \rho_{\mathsf{enc}})]$ is $(O(\log n), 1/\mathrm{poly}(n))$-evaluated (see Theorem 4.2). ■

Acknowledgements

I thank Mark Bun, Ran Canetti, Russell Impagliazzo, and Emanuele Viola for thoughtful conversations about this research. I also thank Mauricio Karchmer for advice on presentational aspects of this paper. Finally, I give special thanks to Marco Carmosino for helpful comments on a draft of this paper, as well as many discussions pertaining to this research.

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach.* Cambridge University Press, 2009.

[BCG$^+$21] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Low-complexity weak pseudorandom functions in $\mathsf{ac}^0[\mathsf{mod2}]$. In *Annual International Cryptology Conference*, pages 487–516. Springer, 2021.

[BFKL93] Avrim Blum, Merrick Furst, Michael Kearns, and Richard J Lipton. Cryptographic primitives based on hard learning problems. In *Annual International Cryptology Conference*, pages 278–291. Springer, 1993.

[BIP$^+$18] Dan Boneh, Yuval Ishai, Alain Passelègue, Amit Sahai, and David J Wu. Exploring crypto dark matter: New simple prf candidates and their applications. In *Theory of Cryptography: 16th International Conference, TCC 2018, Panaji, India, November 11–14, 2018, Proceedings, Part II*, pages 699–729. Springer, 2018.

[BNS92] László Babai, Noam Nisan, and Márió Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *Journal of Computer and System Sciences*, 45(2):204–232, 1992.

[Che18] Lijie Chen. Toward super-polynomial size lower bounds for depth-two threshold circuits. *arXiv preprint arXiv:1805.10698*, 2018.

[CIKK16] Marco L Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning algorithms from natural proofs. In *31st Conference on Computational Complexity (CCC 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[CT93] Fan RK Chung and Prasad Tetali. Communication complexity and quasi randomness. *SIAM Journal on Discrete Mathematics*, 6(1):110–123, 1993.

[DSS16] Amit Daniely and Shai Shalev-Shwartz. Complexity theoretic limitations on learning dnf's. In *Conference on Learning Theory*, pages 815–830. PMLR, 2016.

[DW$^+$00] Carlos Domingo, Osamu Watanabe, et al. Madaboost: A modification of adaboost. In *COLT*, pages 180–189, 2000.

[Fei02] Uriel Feige. Relations between average case complexity and approximation complexity. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 534–543, 2002.

[FX14] Vitaly Feldman and David Xiao. Sample complexity bounds on differentially private learning via communication complexity. In *Conference on Learning Theory*, pages 1000–1019. PMLR, 2014.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986.

[GK23] Halley Goldberg and Valentine Kabanets. Improved learning from kolmogorov complexity. *ECCC Report*, 2023.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[JLSW11] Jeffrey C Jackson, Homin K Lee, Rocco A Servedio, and Andrew Wan. Learning random monotone dnf. *Discrete Applied Mathematics*, 159(5):259–271, 2011.

[JS05] Jeffrey C Jackson and Rocco A Servedio. Learning random log-depth decision trees under uniform distribution. *SIAM Journal on Computing*, 34(5):1107–1128, 2005.

[KLMY19] Daniel Kane, Roi Livni, Shay Moran, and Amir Yehudayoff. On communication complexity of classification problems. In *Conference on Learning Theory*, pages 1903–1943. PMLR, 2019.

[KN96] Eyal Kushilevitz and Noam Nisan. Communication complexity, 1996.

[KNR99] Ilan Kremer, Noam Nisan, and Dana Ron. On randomized one-round communication complexity. *Computational Complexity*, 8:21–49, 1999.

[KOS04] Adam R Klivans, Ryan O'Donnell, and Rocco A Servedio. Learning intersections and thresholds of halfspaces. *Journal of Computer and System Sciences*, 68(4):808–840, 2004.

[KS09] Adam R Klivans and Alexander A Sherstov. Cryptographic hardness for learning intersections of halfspaces. *Journal of Computer and System Sciences*, 75(1):2–12, 2009.

[KV94] Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the ACM (JACM)*, 41(1):67–95, 1994.

[LS09] Nati Linial and Adi Shraibman. Learning complexity vs communication complexity. *Combinatorics, Probability and Computing*, 18(1-2):227–245, 2009.

[Nan21] Mikito Nanashima. A theory of heuristic learnability. In *Conference on Learning Theory*, pages 3483–3525. PMLR, 2021.

[Nis93] Noam Nisan. The communication complexity of threshold gates. *Combinatorics, Paul Erdos is Eighty*, 1:301–315, 1993.

[Raz87] Alexander A Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.

[Raz00] Ran Raz. The bns-chung criterion for multi-party communication complexity. *Computational Complexity*, 9(2):113–122, 2000.

[Reg04] Oded Regev. New lattice-based cryptographic constructions. *Journal of the ACM (JACM)*, 51(6):899–942, 2004.

[Reg09a] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):1–40, 2009.

[Reg09b] Oded Regev. On the complexity of lattice problems with polynomial approximation factors. In *The LLL Algorithm: Survey and Applications*, pages 475–496. Springer, 2009.

[RR97] Alexander A Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997.

[Sch90] Robert E. Schapire. The strength of weak learnability. *Mach. Learn.*, 5:197–227, 1990.

[Sel09] Linda Sellie. Exact learning of random dnf over the uniform distribution. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 45–54, 2009.

[Smo87] Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 77–82, 1987.

[Val84] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[Vio15] Emanuele Viola. The communication complexity of addition. *Combinatorica*, 35:703–747, 2015.

[VW07] Emanuele Viola and Avi Wigderson. Norms, xor lemmas, and lower bounds for gf (2) polynomials and multiparty protocols. In *Twenty-Second Annual IEEE Conference on Computational Complexity (CCC'07)*, pages 141–154. IEEE, 2007.

[Yao82] Andrew C Yao. Theory and application of trapdoor functions. In *23rd Annual Symposium on Foundations of Computer Science (SFCS 1982)*, pages 80–91. IEEE, 1982.

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il