Agnostic Membership Query Learning with Nontrivial Savings: New Results and Techniques

Ari Karchmer ARIKA@BU.EDU
Boston University

Editors: Claire Vernade and Daniel Hsu

Abstract

Designing computationally efficient algorithms in the agnostic learning model (Haussler, 1992; Kearns et al., 1994) is notoriously difficult. In this work, we consider agnostic learning with membership queries for touchstone classes at the *frontier* of agnostic learning, with a focus on how much computation can be *saved* over the trivial run-time of 2^n. This approach is inspired by and continues the study of "learning with nontrivial savings" (Servedio and Tan, 2017). To this end, we establish multiple agnostic learning algorithms, highlighted by:

- An agnostic learning algorithm for circuits consisting of a sublinear number of gates, which can each be any function computable by a sublogarithmic degree k polynomial threshold function (the depth of the circuit is bounded only by size). This algorithm runs in time $2^{n-s(n)}$ for $s(n) \approx n/(k+1)$, and learns over the uniform distribution over unlabelled examples on $\{0,1\}^n$.
- An agnostic learning algorithm for circuits consisting of a sublinear number of gates, where each can be any function computable by a SYM^+ circuit of subexponential size and sublogarithmic degree k. This algorithm runs in time $2^{n-s(n)}$ for $s(n) \approx n/(k+1)$, and learns over distributions of unlabelled examples that are products of $k+1$ *arbitrary and unknown* distributions, each over $\{0,1\}^{n/(k+1)}$ (assume without loss of generality that $k+1$ divides n).

Furthermore, we apply our new agnostic learning algorithms for these classes to also obtain algorithms for randomized compression, exact learning with membership and equivalence queries, and distribution-independent PAC-learning with membership queries.

Our core technique, which may be of independent interest, remixes the learning from natural proofs paradigm (Carmosino et al. 2016, 2017), so that we can tolerate concept classes fundamentally different than $\mathsf{AC}^0[p]$, and achieve *fully* agnostic learning. We make use of communication-complexity-based natural proofs (Nisan, 1993), rather than natural proofs of Razborov (1987) and Smolensky (1987) for $\mathsf{AC}^0[p]$.

Keywords: Agnostic PAC-Learning, Nontrivial Savings, Membership Queries, Natural Proofs, Threshold Functions, Communication Complexity

1. Introduction

Agnostic learning Haussler (1992); Kearns et al. (1992) is an important generalization of PAC-learning Valiant (1984). Agnostic learning is meant to more accurately capture a common approach to machine learning, where a predefined set of functions is explored in order to find the one achieving the least error on a set of data produced by some totally unknown process. Thus, roughly speaking, the objective of an *agnostic learning algorithm* for a complexity class Λ is to output a *hypothesis* h whose error in approximating an arbitrary concept is nearly as small as that of the *best possible* hypothesis within Λ. The class Λ is referred to as the *touchstone class*.

Designing *computationally efficient* (i.e., polynomial time) agnostic learning algorithms for expressive touchstone classes has historically been relatively hard. Even extremely simple touchstone classes such as parity functions are believed to be computationally hard to learn in the agnostic model Blum et al. (1993). Some positive results exist, however, including for piecewise functions Kearns et al. (1992), restricted fan-in two-layer neural nets Lee (1996), geometric patterns Goldman et al. (1997), decision trees, Gopalan et al. (2008), and halfspaces Kalai et al. (2008a).

If we take some combination of the common relaxations considered in computational learning theory, such as access to membership queries, distribution-specific learning, or super-polynomial runtime, more positive results become known. For instance, the famed polynomial time agnostic learning algorithm for parity functions due to Goldreich and Levin (1989) (also referred to sometimes as the KM algorithm after Kushilevitz and Mansour (1991)), uses membership queries and requires a uniform distribution over unlabelled examples. On the other hand, Blum et al. (2003) and Lyubashevsky (2005) show agnostic learning for parities using only random examples, but restricting to the uniform distribution and allowing *slightly* subexponential runtime.

In this work, we study agnostic learning under all three relaxations: access to membership queries, a varying degree of distribution-specificity, and super-polynomial runtime. Throughout the paper, we refer to agnostic learning algorithms that use membership queries as *AMQ-learners*, and the learning model as *AMQ-learning*. Distributional assumptions and/or super-polynomial runtimes are stated explicitly.

More specifically, the goal of this work is to increase the complexity of the touchstone classes that are known to be agnostically learnable. Thus, with respect to the runtime relaxation, we will focus merely on how much computation can be *saved* over the *trivial* runtime of 2^n (here n is the number of binary "inputs" to the target concept). To summarize, we will try to design AMQ-learners that assume some knowledge about the distribution over unlabelled examples, and which run in time $2^{n-s(n)}$ for a *savings function* $s(n)$. Clearly, when $s(n) = n - O(\log(n))$, runtime is polynomial. Generally speaking, we aim for $s(n) \in \omega(\log(n))$. Throughout the paper, we refer to a learning algorithm in a specified model that runs in time $2^{n-s(n)}$ as $s(n)$-nontrivial, or simply nontrivial, if $s(n) \in \omega(\log(n))$.

This perspective is heavily inspired by the idea of Servedio and Tan (2017), who first explicitly considered learning with nontrivial savings. However, our learning model differs significantly, as Servedio and Tan (2017) did not consider agnostic learning, but also did not make use of membership queries. Additionally, for technical reasons, they considered an online mistake-bound model of learning Littlestone (1988), which is actually stronger than distribution-independent PAC-learning Littlestone (1988); Blum (1994) and equivalent to exact learning with equivalence queries only Angluin (1988). In the online mistake-bound model, they obtained nontrivial learning algorithms for AC^0 circuits with a few LTF-gates, or augmented with mod p-gates, in addition to full-basis formulas, branching programs, and span programs of some fixed polynomial size (all less than n^2).

Understanding how much computation can be saved is a relatively new goal in computational learning theory, which, on the other hand, has a longer, fruitful history in complexity theory. For instance, there exist many examples of non-trivial upper bounds slightly better than 2^n for counting or satisfiability algorithms for CNFs, and other NP-hard or #P-hard problems (see Paturi et al. (1997); Schoning (1999); Paturi et al. (2005); Schuler (2005); Fomin and Kaski (2013); Impagliazzo et al. (2012)). Thus, as Servedio and Tan (2017) pointed out, finding out the extent of what can be learned with nontrivial savings is worthwhile to pursue in order to push computational learning theory forward. Additionally, as we will discuss later, a direct implication from nontrivial learning

algorithms to *compression* algorithms for Boolean circuit classes provides a concrete application for our study.

1.1. Our Results

We present a variety of new nontrivial AMQ-learning algorithms. For the sake of clarity, we now present somewhat weaker and slightly informal statements that still convey the main results of this work. The formal and more technically precise statements proved in the body of the paper are easily seen to imply them. An extended technical overview follows in Section 2, which includes a precise definition of AMQ-learning.

AMQ-learning over the uniform distribution. First, we obtain nontrivial AMQ-learners, with respect to a uniform distribution over unlabelled examples on $\{0,1\}^n$, for functions of degree k polynomial threshold functions (PTFs). Specifically, the following touchstone classes:

- **Class 1:** Circuits of at most $n^{0.99}$ gates, where each gate computes any function computable by a PTF of degree $k \leq \log(n)^{0.99}$, in time $2^{n-s(n)}$ for $s(n) \approx n/(k+1)$.
- **Class 2:** Decision trees of at most $n^{0.99}$ depth, where each node is allowed to make a query computed by a PTF of degree $k \leq \log(n)^{0.99}$, in time $2^{n-s(n)}$ for $s(n) \approx n/(k+1)$.

We use $\approx$ to suppress additive factors that are sublinear in n, and an unavoidable logarithmic dependency on the reciprocal of the error rate of the optimal hypothesis in the touchstone class.

AMQ-learning over t-product distributions. We also obtain nontrivial AMQ-learners when the distribution over unlabelled examples on $\{0,1\}^n$ is a *t-product*. A t-product is a distribution that is defined by $t \in \mathbb{Z}$ arbitrary distributions, each over $\{0,1\}^{n/t}$ (assume without loss of generality that t divides n). The t-product distribution is sampled by taking independent samples from each of the t distributions, and concatenating them to form an element of $\{0,1\}^n$. The ability to handle t-product distributions over unlabelled examples is a significant upgrade over the uniform distribution, for at least two reasons. First, this class of distributions allows for intricate dependencies between some of the bits of the unlabelled examples. Second, the t distributions need not even be known to the algorithm. That is, the algorithm is not specific to the choice of the t distributions that make up the t-product.[1]

We give AMQ-learners for functions of SYM^+ circuits of subexponential size and degree k (see touchstone classes 3 and 4), where the distribution over unlabelled examples is an unknown $(k+1)$-product. A SYM^+ circuit of size s and degree k is defined by a pair (p,θ), where p is an n-variable degree-k multilinear polynomial over the integers. Size s means that the magnitude of the coefficients of p is at most s. On the other hand, $\theta : \mathbb{Z} \to \{-1,1\}$ is an arbitrary function. The SYM^+ circuit (p,θ) evaluates by computing the function $s : \{0,1\}^n \to \{-1,1\}$ defined as $s(x) = \theta(p(x))$.

- **Class 3:** Circuits of at most $n^{1-\varepsilon}$ gates, where each gate computes any function computable by a SYM^+ circuit of size 2^{n^ζ} ($\zeta < \varepsilon$) and degree $k \leq \log(n)^{0.99}$, in time $2^{n-s(n)}$ for $s(n) \approx n/(k+1)$.

1. If we could upgrade further to arbitrary distributions, then we could remove the need for membership queries (see Feldman (2009b)). This is an interesting goal for future research.

- **Class 4:** Decision trees of at most $n^{1-\varepsilon}$ depth, where each node is allowed to make a query computed by a size $2^{n^{\zeta}}$ ($\zeta < \varepsilon$) and degree $k \leq \log(n)^{0.99}$ SYM^{+} circuit, in time $2^{n-s(n)}$ for $s(n) \approx n/(k+1)$.

Note, degree k PTFs with sum of coefficients less than s are a subclass of size s, degree k SYM^{+} circuits. So, to expand the class of distributions over unlabelled examples handled by the AMQ-learner for touchstone classes 1 and 2, we have to modify the touchstone class by bounding the weights of each PTF-gate slightly. However, size s, degree k SYM^{+} circuits are still more general than degree k PTFs with total weight bounded by s. Because of this, touchstone classes 3 and 4 are not necessarily supersets of touchstone classes 1 and 2.

1.2. Some Applications of Our Results

Several applications of the nontrivial AMQ-learners for classes 1-4 in are in order. We refer to Section 6 for precise statements and further discussion of these applications.

Compression Algorithms for Classes 1-4. Randomized compression algorithms for Boolean functions (Chen et al., 2015) are obtained in a simple and generic way from membership query learning over the uniform distribution (this was noticed already in Carmosino et al. (2016); Servedio and Tan (2017)). Even though our results are AMQ-learners, they can be used in the *realizable* setting as membership query learning algorithms as needed. Therefore, we also obtain randomized compression algorithms for touchstone classes 1-4, which were not known before.

Nontrivial Exact Learning and Distribution-Independent PAC-Learning for Classes 1-4. In a similar fashion, randomized learning algorithms in the exact learning with membership and equivalence queries model (Angluin, 1988) are obtained from membership query learning over the uniform distribution. Additionally, it is also known (by standard arguments – see Section 2.4 of Angluin (1988)) that algorithms for exact learning with membership and equivalence queries imply distribution-independent membership query learning algorithms in the realizable PAC model. Hence, we also obtain nontrivial exact learning and distribution-independent PAC learning algorithms for classes 1-4, which were not known prior.

1.3. Our Approach

The approach of Servedio and Tan (2017) was to convert circuit lower bound methods (random restrictions, Nečiporuk's method) into nontrivial learning algorithms in the online mistake bound model. Because of this, they asked (see section 5 of Servedio and Tan (2017) for exact quotations):

1. Can other proof techniques from computational complexity be used to obtain other nontrivial learning algorithms?

2. Many circuit classes have known lower bounds, but not nontrivial learning algorithms (in any model); can we design nontrivial learning algorithms for such classes?

The approach we take is led by this line of questioning. Specifically, we translate circuit lower bounds proved by a communication complexity based method due to Nisan (1993) into nontrivial AMQ-learners. This goes to answer question 1 above. It also means that we obtain answers to question 2, because, fixing a degree k, none of touchstone classes 1-4 can compute the generalized

inner product function of degree $k+1$ Nisan (1993), but prior to this work no nontrivial learning algorithms were known for these classes (in any learning model). The essential details and definitions of communication complexity in the relevant communication models will be explained in sufficient depth in the technical overview (Section 2); we point to Kushilevitz and Nisan (1996) for further reference.

At a very high level, the method of Nisan (1993) is the following. First, identify a function $\phi : \{0,1\}^n \to \{0,1\}$ that requires *high* communication complexity (e.g., $\Omega(n)$). This can be in any communication model, such as k-party number-on-forehead (NOF), two-party randomized, or two-party deterministic. Then, identify a circuit class $\mathscr{C}$ (with an associated size parameter $s(n)$) such that, for every function g computable by a circuit of type $\mathscr{C}$ and size $s(n)$, g is computable by a *low* communication protocol in that communication model. *Low* communication cost indicates a relatively small function of $s(n)$ like $\log(s(n))$. Finally, one can conclude that f requires large circuits of type $\mathscr{C}$.

At the core of Nisan's method is the *upper bound* on communication complexity for the circuit class $\mathscr{C}$. Thus, by specifically showing that communication complexity *upper bounds* in the k-party NOF model imply, in a general sense, approximately n/k-nontrivial AMQ-learners, we end up translating Nisan's general *lower bound* method into nontrivial AMQ-learners. Since circuit lower bounds for explicit functions can be proved for each of touchstone classes 1-4 using Nisan's method, this suffices to derive our AMQ-learners. In Section 2, we give an in-depth explanation of the construction of the AMQ-learner for deterministic NOF protocols.

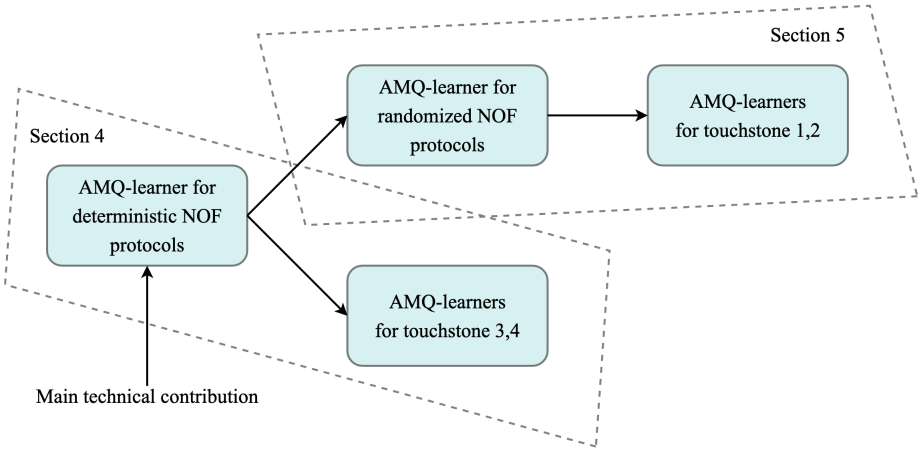

Figure 1: The progression of our constructions.

1.4. Related Results and Discussion

Not many touchstone classes that resemble ours are known to be nontrivially learnable, whether in AMQ-learning or the exact learning model. The most related results are the learning algorithms of Servedio and Tan (2017) and Carmosino et al. (2017). We now compare our results to each.

For comparison with Servedio and Tan (2017), let us focus on our results in the nontrivial exact learning model (see Section 1.2), which is the focus of their work. Classes 1 and 2 are possibly more interesting than the classes shown to be nontrivially learnable in the exact learning with (only) equivalence queries by (Servedio and Tan, 2017). For example, LTF-of-AC^0 and Parity-of-AC^0 – these circuit classes cannot compute the parity function or an LTF, respectively, while touchstone classes 1 and 2 can compute both. Additionally, our AMQ-learners for classes 1 and 2 are $n/O(1)$-nontrivial (when fixing a constant degree k for the PTFs), while (Servedio and Tan, 2017) only obtain $n/\text{polylog}(n)$-nontrivial learners for LTF-of-AC^0 and Parity-of-AC^0. However, it is important to note that our results are not necessarily formally stronger than Servedio and Tan (2017) because we require membership queries, while Servedio and Tan (2017) do not.

For comparison to Carmosino et al. (2017), let us focus on our results in the AMQ-learning model, which is their focus as well. Carmosino et al. (2017) obtain a uniform-distribution-specific algorithm with some relatively mild agnostic guarantees for $\mathsf{AC}^0[p]$ circuits, for any prime p. More specifically, the AMQ learner for $\mathsf{AC}^0[p]$ finds a hypothesis with error as much as $\text{polylog}(n)$ times the optimal hypothesis in $\mathsf{AC}^0[p]$. In contrast, our AMQ-learner for classes 3 and 4 is *fully* agnostic (i.e., it does not have the $\text{polylog}(n)$ loss factor). Additionally, our AMQ-learner works over k-product distributions, rather than just the uniform distribution. In these two respects, our algorithm compares favorably. Plus, touchstone classes 3 and 4 are not contained in $\mathsf{AC}^0[p]$ for any prime p. However, their algorithm is much faster – quasi-polynomial time – while ours is approximately $n/2$-nontrivial.

Could we improve our results? An interesting note regarding touchstone classes 3 and 4 is that seemingly small improvements would be a breakthrough in computational learning theory (indeed classes 3 and 4 reach the AMQ-learning *frontier*). To see this, recall that touchstone classes 3 and 4 are classes of functions of subexponential size but sublogarithmic degree SYM^+ circuits. Now observe that a *quasipolynomial* size SYM^+ circuit with *polylogarithmic* degree can compute all of ACC^0 (Beigel and Tarui, 1994). Nontrivial AMQ-learning of ACC^0 is an open problem, even when the distribution over unlabelled examples is fixed to be uniform, and the learning algorithm is not even agnostic. Thus, extending our AMQ-learner to handle a circuit with even *one single* gate that computes a quasi-polynomial size, polylogarithmic degree SYM^+ function would be a breakthrough in learning theory. In fact, since nontrivial learning algorithms for ACC^0 can be "sped-up" (by a result of Oliveira and Santhanam (2016)), we would obtain a learning algorithm in time 2^{n^ε}, for any $\varepsilon > 0$. On the other hand, pseudorandom functions with exponential security computable by ACC^0 circuits are conjectured Boneh et al. (2018). These conjectures preclude $n/O(1)$-nontrivial AMQ-learners for ACC^0.

2. Extended Technical Overview

In this section, we provide an in-depth overview of the technique behind our main technical contribution, from which all the AMQ-learners for touchstone classes 1-4 are derived. The main technical contribution constructs the AMQ-learner over k-product distributions for the touchstone class of functions admitting k-party NOF communication protocols of cost c (we will set c as a function of n and k later; for now, think of k as a constant and $c := \Theta(n^{0.99})$).

We begin by defining k-party NOF communication, then define nontrivial AMQ-learning precisely, and finally walk through the main technical contribution.

k-party NOF communication. The k-party NOF communication model is the following. There are k parties, each having unbounded computational power, who try to collectively compute a function. The input to the function is separated into k parts, each containing n/k of the inputs (assume without loss of generality that n is a multiple of k), and the i^{th} party sees all parts except the i^{th}. The communication between the parties is by broadcast: any party can send a bit to all others simultaneously.

All parties 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 1 ($\Pi[k,c]$ class) *$\Pi[k,c]$ is defined to be the class of functions $f:\{0,1\}^n \to \{-1,1\}$ that, for any partition of the n variables into k length n/k parts, can be computed by a k-party communication protocol with complexity c.*

Again, for now, one can think of n as being a multiple of k without loss of generality. Within this extended technical overview, we will use the notation $[x_1,\cdots x_k]$ to denote the concatenation of the k parts $x_1,\cdots x_k$, according to the appropriate partition, which is implicit. For simplicity, it is appropriate to think of the partition as contiguous blocks of n/k bits, from "left to right" (depicted later in Figure 2).

AMQ-learning. Valiant's PAC-learning model operates in the *realizable* setting, where the concept $f:\{0,1\}^n \to \{-1,1\}$ is assumed to be part of a fixed class. In agnostic learning, the concept is modelled instead as an arbitrary *distribution* $\mathscr{D}$ over $\{0,1\}^n \times \{-1,1\}$. For $(x,y) \sim \mathscr{D}$, x is called the *example*, and y is called the *label*. Note that the label for one example may be randomized. We refer to the marginal distribution ρ over x for $(x,y) \sim \mathscr{D}$ as the distribution over unlabelled examples, or example distribution for short.

The goal of agnostic learning for a touchstone class Λ is to find a hypothesis $h:\{0,1\}^n \to \{-1,1\}$ such that, with probability $1-\delta$, h satisfies

$$\mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}}[h(x)\cdot y] \geq \mathop{\mathrm{opt}}_{\mathscr{D}}(\Lambda) - \varepsilon$$

where the learning algorithm is given ε,δ as input, and $\mathop{\mathrm{opt}}_{\mathscr{D}}(\Lambda)$ is defined as

$$\mathop{\mathrm{opt}}_{\mathscr{D}}(\Lambda) := \max_{c\in\Lambda} \mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}}[c(x)\cdot y]$$

Throughout this paper, we will make assumptions about ρ, the distribution over unlabelled examples. We specify that we obtain agnostic learning when ρ is restricted to be part of some class of distributions (e.g., k-product distributions). In AMQ-learning, we allow the learning algorithm to have access to a membership query oracle, which works as follows. The AMQ-learner can submit an example z, of its choice, and receive back the label y with probability $\Pr[(x,y)\sim\mathscr{D}|x=z]$.

In nontrivial AMQ-learning, the two notions of complexity that we consider are:

- Query complexity: The total number of membership queries made by the AMQ-learner, in the worst-case.

- Time complexity: The worst-case run-time complexity of the AMQ-learner.

Normally, query and time complexity are measured as functions of n, ε, δ. This can make statements about complexity a bit messy in the nontrivial learning setting; we will fix $\delta := 2/3$, and $\varepsilon := 2^{-n^{0.99}}$, and thus bound complexity purely as a function of n. Recall, an AMQ-learner is $s(n)$-nontrivial if time complexity is at most $2^{n-s(n)}$.

Towards the AMQ-learner. Towards the full AMQ-learner, we construct a *weak* AMQ-learner. A weak AMQ-learner essentially preserves a small portion of the correlation between the optimal hypothesis in the touchstone, and the target concept. We use a standard notion due to Feldman (2009a); Kanade and Kalai (2009).

Definition 2 (Weak agnostic learning) *For $0 < \alpha \leq \gamma < 1$, a (γ, α)-weak agnostic learner for $\mathscr{C}$ is an algorithm $\mathscr{A}$ that (when receiving some oracle access to a target concept $\mathscr{D}$) outputs a hypothesis $h : \{0,1\}^n \to \{-1,1\}$ such that*

$$\operatorname*{opt}_{\mathscr{D}}(\mathscr{C}) = \max_{c \in \mathscr{C}} \mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}}[c(x)y] \geq \gamma \implies \Pr_{\mathscr{A}}\left[\mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}}[h(x)y] \geq \alpha\right] \geq 2/3$$

We construct a weak AMQ-learner of this kind, because Feldman (2009a); Kanade and Kalai (2009) show that it can be *boosted*, using *distribution-specific* boosting algorithms, into a full-blown agnostic learning algorithm. This means that the boosting algorithm only ever invokes the weak AMQ-learner on a single distribution over unlabelled examples. This is important for us, because our initial result does not handle arbitrary distributions over unlabelled examples.

For the sake of simplicity in exposition, the main technical contribution is stated below with respect to the uniform distribution over unlabelled examples, and the concept is considered an arbitrary *function*, rather than distribution. The corresponding theorem proved in the body of the paper extends this to k-product distributions over unlabelled examples (as discussed previously), as well as concepts that are arbitrary distributions.

Define $\mathrm{Cor}(f, \Lambda) := \max_{g \in \Lambda} \mathbb{E}_{z\sim\{0,1\}^n}[g(z) \cdot f(z)]$.

Theorem 3 (Main technical contribution) *Let $f : \{0,1\}^n \to \{-1,1\}$. There exists an oracle algorithm $\mathscr{A}$ with the guarantee that if $\mathrm{Cor}(f, \Pi[k,c]) \geq \gamma$, then $\mathscr{A}^f$ outputs $h : \{0,1\}^n \to \{-1,1\}$ such that*

$$\Pr_{\mathscr{A}}\left[\Pr_{z\sim\{0,1\}^n}[h(z) = f(z)] \geq 1/2 + \alpha\right] \geq 2/3$$

with:

- $\alpha := \Omega(\gamma \cdot 2^{-c2^k - k})$,
- *query complexity* $q := 2^{n-s(n)}$, *for* $s(n) = n/k - \log(k) + 4(\log(\gamma) - c2^k + k))$
- *time complexity* $t := O(q)$.

Here, $z \sim \{0,1\}^n$ denotes sampling over the uniform distribution.

The main tool we use towards this theorem is the "k-party norm" of a function $f : \{0,1\}^n \to \{-1,1\}$, denoted $R_k(f)$, defined as:

Definition 4 (k-party norm) *For $f : \{0,1\}^n \to \{-1,1\}$, the k-party norm of f is defined as*

$$R_k(f) := \mathop{\mathbb{E}}_{x_1^0,\cdots x_k^0,x_1^1,\cdots,x_k^1 \sim \{0,1\}^{n/k}} \left[\prod_{\varepsilon_1,\varepsilon_2 \in \{0,1\}} f([x_1^{\varepsilon_1},\cdots x_k^{\varepsilon_2}]) \right] \tag{1}$$

Again, we assume without loss of generality that k divides n.

The main insight towards a weak AMQ-learner for $\Pi[k,c]$ is that $R_k(f)$ upper bounds the correlation of f with functions computable by deterministic k-party NOF communication protocols. Proved in all three of Chung and Tetali (1993); Raz (2000); Viola and Wigderson (2007), is the following bound:

Theorem 5 (k-party correlation bound) *For every function $f : \{0,1\}^n \to \{-1,1\}$,*

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

for x uniformly distributed over $\{0,1\}^n$.

This correlation bound can be used to obtain a *natural proof*, in the sense of Razborov and Rudich (1997), against $\Pi[k,c]$. This has observed for many years as folklore, but explicitly shown by Karchmer (2023).

On first thought, it seems we should then be able to obtain learning algorithms using the technique of Carmosino et al. (2016, 2017), but this does not work. To see this, we can observe that the method of Carmosino et al. (2016, 2017) requires the concept class to contain $\mathsf{AC}^0[p]$—and $\mathsf{AC}^0[p]$ contains functions that have $\Omega(n/\exp(k))$ communication complexity (e.g. generalized inner-product Viola and Wigderson (2007)). Thus, we cannot in general apply their method with a natural proof against $\Pi[k,c]$, since this trivializes the bound in (2). To get a little more specific, Carmosino et al. (2016, 2017) requires the concept class to contain $\mathsf{AC}^0[p]$ because the "combinatorial designs" within the Nisan-Wigderson PRG, used at the heart of their construction, require $\mathsf{AC}^0[p]$-circuits to locally compute.

The way we get around this is as follows: we use the combinatorial design that *arises naturally* from the k-party norm itself, and then use the k-party correlation bounds as a *distinguisher*. Observe: $R_k(f)$ is the expectation of the product of f computed on all combinations of a list of $2k$, n/k-bit "slices." So, we view the $2n$ bits in this list as a "seed," and each of the 2^k combinations as the designs over slices of this seed. See Figure 2 for a graphical depiction.

x_1^0	x_2^0	x_3^0	x_4^0	x_5^0	x_6^0
x_1^1	x_2^1	x_3^1	x_4^1	x_5^1	x_6^1

x_1^0	x_2^0	x_3^0	x_4^0	x_5^0	x_6^0
x_1^1	x_2^1	x_3^1	x_4^1	x_5^1	x_6^1

Figure 2: Two examples of designs naturally arising from $R_6(f)$. Each x_j^i is a n/k bit slice of the $2n$ bit seed. Design indices '011000' and '110101' are highlighted on the left and right images respectively.

In other words, if we generate a random seed x, and then query the concept f at every point z_i indicated by the seed projected to design $i \in \{0,1\}^k$ (see Figure 3), then the distribution over

the labels $\langle f(z_i)\rangle_{i\in\{0,1\}^k}$ can be distinguished from a uniformly random string in $\{-1,1\}^{2^k}$! The distinguisher simply takes the product of the bits. By design, this product has expectation at least $(\gamma 2^{-c})^{2^k}$, which follows directly from Theorem 5 (if we assume that the concept f has γ correlation with $\Pi[k,c]$). For a random string, the expectation is zero. Therefore the distinguishing advantage is $(\gamma 2^{-c})^{2^k}$.

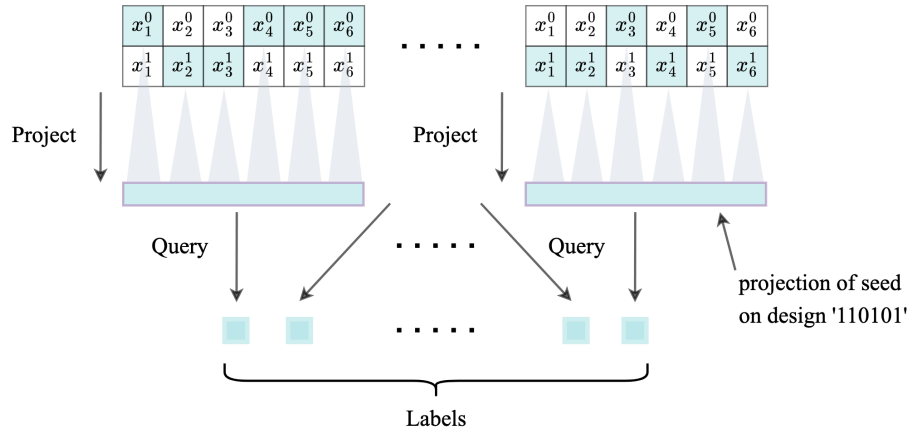

Figure 3: Example of projected designs to form queries. The claim is that the sequence of 2^k labels can be distinguished from a uniformly random string in the set $\{-1,1\}^{2^k}$, whenever the concept has a large k-party norm.

Distinguishing labels queried in this randomized manner from a random string is sufficient to obtain a randomized prediction algorithm for f with weak accuracy approximately $1/2+(\gamma 2^{-c})^{2^k}$, using a (rather involved) hybrid argument. Then, we can proceed as Carmosino et al. (2016) does: we use a randomized pre-processing stage to "derandomize" the randomized predictor. That is, we convert the randomized predictor to a randomized process that generates deterministic hypotheses with a weaker (but similar) prediction accuracy, with some non-negligible probability. Finally, we use a constructive averaging argument to obtain a deterministic hypothesis with weak accuracy, with very high probability.

Giving an upper bound on the queries required to implement this method is argued by bounding the number of subcubes that are queried in their entirety. The maximum number of subcubes is roughly k, and each is of size $2^{n/k}$. Hence we can bound the total number of queries roughly by $2^{n/k}$. We refer to Lemma 15 for details.

3. Preliminaries

We cover basics of circuits. Other important definitions used in the paper are defined as needed throughout.

Complexity classes, circuits, gates, etc. We 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 depth, polynomial size,

unbounded fan-in AND/OR/NOT/MODp circuits, where $p \in \mathbb{N}$ is a prime number. An XOR gate takes the sum modulo 2 of its inputs.

An linear threshold function (LTF) gate computes an LTF defined by $t(x_1, \cdots x_m) := \sum_{i=1}^{m} w_i x_i \geq^{?} \theta$, which outputs 1 if and only if the sum of the inputs weighted by real coefficients $w_1, \cdots w_m$ exceeds a threshold θ. When the weights are fixed to be 1 and $\theta = m/2$, we call it a majority gate. A polynomial threshold function (PTF) is defined by a polynomial $p(x_1, \cdots, x_n)$ with real number coefficients. The $p(x)$ for input $x \in \{0,1\}^n$ is 1 if $p(x_1, \cdots, x_n) \geq 0$ and -1 otherwise. Note, the domain of the PTF is $\{0,1\}^n$, so different polynomials can define identical PTFs. The degree of a PTF is the degree of the polynomial p. We use PT-Ckt$[k,m]$ to denote the class of circuits consisting of at most m *gates*, which can each compute a PTF of degree k, and PT-Dt$[k,d]$ to denote the class of decision trees consisting of at most depth d, with internal nodes computing PTFs of degree k.

A SYM$^+$ circuit of size s and degree d is a pair (p,θ), where p is an n-variable degree-d multilinear polynomial over the integers. Size s means that the magnitude of the coefficients of p is at most s. On the other hand, $\theta : \mathbb{Z} \to \{-1,1\}$ is any function. The SYM$^+$ circuit (p,θ) evaluates by computing the function $s : \{0,1\}^n \to \{-1,1\}$ defined as $s(x) = \theta(p(x))$. We also consider circuits, where each gate computes any function computable by a SYM$^+$ circuit. We denote by SYM$^+$-Ckt$[k,t,m]$ the class of circuits consisting of at most m gates, which can each compute any function computable by a SYM$^+$ circuit of degree k and size t. We denote by SYM$^+$-Dt$[k,t,d]$ the class of decision trees of depth d, with node queries computable by any SYM$^+$ circuit of degree k and size t.

4. Learning Efficient k-Party NOF Communication Protocols

In this section, we will construct an agnostic learning algorithm for any touchstone class that has an efficient k-party deterministic number-on-forehead (NOF) communication protocol (defined previously but repeated below for the reader's convenience).

The k-party NOF communication model is the following. There are k parties, each having unbounded computational power, who try to collectively compute a function. The input to the function is separated into k parts, each containing n/k of the inputs (assume without loss of generality that n is a multiple of k), and the i^{th} party sees all parts except the i^{th}. The communication between the parties is by broadcast: any party can send a bit to all others simultaneously.

All parties 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 6 ($\Pi[k,c]$ class) *$\Pi[k,c]$ is defined to be the class of functions $f : \{0,1\}^n \to \{-1,1\}$ that, for any partition of the n variables into k length n/k parts, can be computed by a k-party communication protocol with complexity c.*

Again, for now, one can think of n as being a multiple of k without loss of generality. We will obtain AMQ-learners that do not require this.

To construct the learning algorithm, we will use a two-step process: first, we will construct a *weak* agnostic learner, and then we will apply a distribution-specific agnostic boosting techniques, such as by Feldman (2009a) or Kanade and Kalai (2009).

4.1. Weak Learning

For *weak* agnostic learning, we use a standard notion introduced by Kalai et al. (2008b) , because this type of weak learner can be subsequently used as an oracle in the boosting algorithms of Feldman (2009a); Kanade and Kalai (2009). Basically, the weak agnostic learner is required to recover some of the advantage that exists for the optimal concept $c \in \mathscr{C}$.

Definition 7 (Weak agnostic learning) *For $0 < \alpha \leq \gamma < 1$, a (γ, α)-weak agnostic learner for $\mathscr{C}$ is an algorithm $\mathscr{A}$ that (when receiving some oracle access to a target concept $\mathscr{D}$) outputs a hypothesis $h : \{0,1\}^n \to \{-1,1\}$ such that*

$$\max_{c \in \mathscr{C}} \mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [c(x)y] \geq \gamma \implies \Pr_{\mathscr{A}} \left[\mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [h(x)y] \geq \alpha \right] \geq 2/3$$

Our AMQ-learner works for the following class of distributions, which we term t-products. We use the notation $a||b$ to denote string concatenation of a and b.

Definition 8 (t-product distribution) *Assume that $t \in \mathbb{Z}$ divides $n \in \mathbb{Z}$. A t-product distribution $\rho = \langle (\mu_i)_{i\in[t]}, \sigma \rangle$ is defined by t distributions $(\mu_i)_{i\in[t]}$ each over $\{0,1\}^{n/t}$, and a permutation σ over $[n]$. For $z \in \{0,1\}^n$, let $\sigma(z) := z_{\sigma(1)}||z_{\sigma(2)}||\cdots||z_{\sigma(n)}$. To sample ρ,*

1. *For each $i \in [t]$, sample $x_i \sim \mu_i$.*
2. *Output $\sigma(x_1||\cdots||x_t)$.*

Observe that a t-product distribution is defined to sample from $\{0,1\}^n$, since we assume that t divides n. We denote by Δ^t_σ the class of t-product distributions that apply the permutation σ.

The assumption that t divides n in the definition of t-products will also not sacrifice any generality in the ensuing AMQ-learner. Rephrased now more formally, the end-goal of this subsection is to prove the following theorem.

Theorem 9 *There exists a $(\gamma, \gamma \cdot 2^{-c2^k-k})$-weak agnostic learner for $\Pi[k,c]$. The learner uses membership queries and learns over any $\rho \in \Delta^k_\sigma$ (for some fixed σ), and has*

- *query complexity $q := 2^{n-s(n)}$, for $s(n) = n/k - \log(k) + 4(\log(\gamma) - c2^k + k))$*
- *time complexity $t := O(q)$.*

First, we will prove an easier version, and then extend it via simple lemmas proved after.

Definition 10 (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. Define $\mathrm{Cor}_\rho(f,\Lambda) := \max_{h\in\Lambda} |\mathbb{E}[f(x)\cdot h(x)]|$, where $x \sim \rho$.*

Theorem 11 *Let $f : \{0,1\}^n \to \{-1,1\}$. Given the permutation σ, there exists an oracle algorithm $\mathscr{A}$ with the guarantee that if $\mathrm{Cor}_\rho(f, \Pi[k,c]) \geq \gamma$, then for any $\rho \in \Delta^k_\sigma$, $\mathscr{A}^{f,\rho}$ (with query access to f and sampling access to ρ) outputs $h : \{0,1\}^n \to \{-1,1\}$ such that*

$$\Pr_{\mathscr{A}} \left[\Pr_{z\sim\rho} [h(z) = f(z)] \geq 1/2 + \alpha \right] \geq 2/3$$

with:

- $\alpha := \Omega(\gamma \cdot 2^{-c2^k-k})$,
- *query complexity* $q := 2^{n-s(n)}$, *for* $s(n) = n/k - \log(k) + 4(\log(\gamma) - c2^k + k))$
- *time complexity* $t := O(q)$.

Proof

To prove the theorem, we will:

1. Present an algorithm $\mathscr{P}^{f,\rho}$ that nearly witnesses the statement.
2. We will then argue that establishing advantage, query and time complexity parameters for $\mathscr{P}^{f,\rho}$ similar to those desired by Theorem 11 suffices to prove Theorem 11 (Lemma 12).
3. Finally, we will prove that $\mathscr{P}^{f,\rho}$ does satisfy those parameters (Lemmas 14, 15 and 16).

Lemmas 12, 14, 15 and 16 suffice to prove Theorem 11. ■

Notation. For any string $z \in \{0,1\}^*$ denote the bit-wise negation of z by $\bar{z}$. For the $2 \times n$ table $\mathbf{B}$ with entries $x_1^0, x_1^1 \cdots x_n^0, x_n^1 \in \{0,1\}^m$, use $\mathbf{B}|_z \in \{0,1\}^{mn}$ to denote the concatenation of strings $x_1^{z_1} x_2^{z_2} \cdots x_n^{z_n}$. For a "lookup" table $\mathbf{T}$, let $\mathbf{T}[i]$ (for $i \in \{0,1\}^m$) denote the value stored at location i of $\mathbf{T}$. For shorthand, let $k := k(n), c := c(n)$.

Algorithm 1 $\mathscr{P}^{f,\rho}$

Input: $k, n \in \mathbb{N}$. A description of σ. Query access to f, sample access to ρ.
▷ *Begin preprocessing.* ◁
Sample $x^0, x^1 \sim \rho$.
Apply $x^0 \leftarrow \sigma^{-1}(x^0)$ and $x^1 \leftarrow \sigma^{-1}(x^1)$
Split x^0, x^1 into $x_1^0, x_1^1 \cdots x_k^0, x_k^1 \in \{0,1\}^{n/k}$, which are k n/k-bit blocks so that each x_i^0, x_i^1 is sampled according to the i^{th} distribution that makes up ρ.
Choose uniformly random string $b \in \{0,1\}^k$.
Choose uniformly random string $r \in \{-1,1\}^{2^k}$. Partially fill a $2 \times k$ table $\mathbf{B}$, such that $\mathbf{B}[b_i, i] = x_i^{b_i}$. Fill other entries with $\{*\}^n$.
For every string $a < \bar{b}$

> Viewing $\mathbf{B}|_a$ as a partial assignment z^* of n bits, **query** f on all n-bit points consistent with $\sigma(z^*)$.
> Place the query-label pairs in a lookup table $\mathbf{T}$.

▷ *End preprocessing.* ◁
Generate and output a circuit h (with the random string $r \in \{-1,1\}^{2^k}$ hard-wired), according to the following template:

> On input $z \in \{0,1\}^n$,
> Place the i^{th} out of k, n/k-bit blocks of z inside $\mathbf{B}[\bar{b}_i, i]$.
> Compute the values
> $$v = \prod_{a<\bar{b}} \mathbf{T}[\mathbf{B}|_a] \text{ and } v' = \prod_{a \geq \bar{b}} r_a$$
> Output $v \cdot v' \cdot r_{\bar{b}}$

Lemma 12 *Suppose that $\mathscr{P}$ makes at most $q := q(n)$ queries while running in time $t := t(n)$, and that whenever $\mathrm{Cor}_\rho(f, \Pi[k,c]) \geq \gamma$, it holds that*

$$\Pr_{R, z\sim\rho}\left[h(z) = f(z) : h \leftarrow \mathscr{P}^{f,\rho}\right] \geq \frac{1}{2} + \Omega(\alpha) \tag{3}$$

for randomness $R = (r, b, x_1^0, x_1^1, \cdots x_k^0, x_k^1)$ of $\mathscr{P}$. Then there exists a randomized oracle algorithm $\mathscr{A}^{f,\rho}$ with the guarantee that if $\mathrm{Cor}_\rho(f, \Pi[k,c]) \geq \gamma$, then $\mathscr{A}^{f,\rho}$ outputs $h : \{0,1\}^n \to \{-1,1\}$ such that

$$\Pr_{\mathscr{A}}\left[\Pr_{z\sim\rho}\left[h(z) = f(x)\right] \geq \frac{1}{2} + \Omega(\alpha)\right] \geq 2/3$$

with:

- $\alpha := \Omega(\gamma \cdot 2^{-c2^k - k})$,
- *query complexity* $q' := O(\alpha^{-4}) \cdot q$,
- *time complexity* $t' := O(q')$.

Lemma 13 (Chernoff Bound, cf. Theorem 2.1 Janson et al. (2011)) *Let $X \sim \mathsf{Bin}(m, p)$ and $\lambda = m \cdot p$. For any $t \geq 0$,*

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \exp\left(\frac{-t^2}{2(\lambda + t/3)}\right)$$

Proof [Sketch.] The statement follows by a constructive averaging argument. First, consider that (3) implies that

$$\Pr_R\left[\Pr_z\left[h(z) = f(z) : h \leftarrow \mathscr{P}^{f,\rho}\right] \geq \frac{1}{2} + \Omega(\alpha)\right] \geq \Omega(\alpha) \tag{4}$$

by standard averaging (see Arora and Barak (2009) for reference).

Therefore, viewing $\mathscr{P}^{f,\rho}$ as a distribution over hypotheses h, it follows that $\mathscr{A}^{f,\rho}$ exists by efficiently constructing a "good" hypothesis by randomized trial-and-error. We may sample $O(\alpha^{-2})$ candidate hypotheses in parallel using $\mathscr{P}^{f,\rho}$ and then compare each to the concept by checking random examples. By application of Lemma 13, there will be a "good" hypothesis with constant probability, and $O(\alpha^{-2})$ examples will be enough to check that each circuit with good enough accuracy is indeed good enough, with constant probability. In total, with $O(\alpha^{-4})$ random samples, $\mathscr{A}^{f,\rho}$ uses $\mathscr{P}^{f,\rho}$ to output $h : \{0,1\}^n \to \{-1,1\}$ such that

$$\Pr_{\mathscr{A}}\left[\Pr_{z\sim\rho} h(z) = f(x) \geq \frac{1}{2} + \Omega(\alpha)\right] \geq 2/3$$

■

We will now establish that the properties of $\mathscr{P}^{f,\rho}$ are as desired.

Lemma 14 (Advantage) *It holds that*

$$\Pr_{R,z}\left[h(z) = f(z) : h \leftarrow \mathscr{P}^{f,\rho}\right] \geq \frac{1}{2} + \Omega(\alpha) \tag{5}$$

for randomness $R = (r, b, x_1^0, x_1^1, \cdots x_k^0, x_k^1)$ of $\mathscr{P}$.

Proof To prove the lemma, we will use the fact that the k-party norm can help distinguish random functions from functions that correlate with k-party protocols, together with a hybrid argument.

We will define the following distributions.

- Let Q_a be the distribution over the value at location $\mathbf{B}|_a$ inside lookup table $\mathbf{T}$ inside $\mathscr{P}^{f,\rho}$. This distribution is over the randomness of $z, b, x_1^0, x_1^1, \cdots x_k^0, x_k^1$.
- Let $Q = (Q_a)_{a\in\{0,1\}^k}$ denote the joint distribution over Q_a for all possible strings $a \in \{0,1\}^k$.
- For $b^* \in \{0,1\}^k$ (representing an integer in $[2^k]$), define the distribution H_{a,b^*} as

$$H_{a,b^*} = \begin{cases} Q_a \text{ when } a < b^* \\ U_1 \text{ otherwise} \end{cases}$$

- Define the distribution H_{b^*} over $\{-1,1\}^{2^k}$ as the joint distribution over $(H_{a,b^*})_{a\in\{0,1\}^k}$.

Now, observe that for $H_{\{1\}^k}$, we have that

$$\mathop{\mathbb{E}}_{\sigma\sim H_{\{1\}^k}}\left[\prod_{i\in[2^k]}\sigma_i\right] = \mathop{\mathbb{E}}_{\sigma\sim Q}\left[\prod_{i\in[2^k]}\sigma_i\right] = \mathop{\mathbb{E}}_{\substack{z,r,\\ x_1^0,x_1^1\\ \cdots\\ x_k^0,x_k^1}}\left[\prod_{a\in\{0,1\}^k}\mathbf{T}[\mathbf{B}|_a]\right]$$

Let $z^{(i)}$ denote the i^{th} (out of k) block of n bits of z, in order from most significant bit to least. Then,

$$\mathop{\mathbb{E}}_{\substack{z,r,\\ x_1^0,x_1^1\\ \cdots\\ x_k^0,x_k^1}}\left[\prod_{a\in\{0,1\}^k}\mathbf{T}[\mathbf{B}|_a]\right] = \mathop{\mathbb{E}}_{\substack{z,r,\\ x_1^0,x_1^1\\ \cdots\\ x_k^0,x_k^1}}\left[\prod_{a_1,\cdots a_k\in\{0,1\}} f(x_1^{a_1},\cdots x_k^{a_k}) \mid z^{(i)} = x_i^1 \ \forall\, i\in[k]\right] \tag{6}$$

Thus, when $z\sim\rho$, the quantity on the right-hand-side of (6) is, by definition, equivalent to $R_k(f\circ\rho)$. That is, the k-party norm of the composition of f with a sampler for the distribution $\rho\in\Delta_\sigma^k$. Therefore, since ρ has the k-product structure, our assumption implies that

$$\mathop{\mathbb{E}}_{\sigma\sim H_{\{1\}^k}}\left[\prod_{i\in[2^k]}\sigma_i\right] = R_k(f\circ\rho) = R_k(f) \geq \gamma\cdot 2^{-c2^k}$$

Again, the equality between $R_k(f\circ\rho) = R_k(f)$ holds because ρ is a k-product distribution, so if $\mathrm{Cor}_\rho(f,\Pi[k,c]) \geq \gamma$, then $\mathrm{Cor}(f\circ\rho,\Pi[k,c]) \geq \gamma$, because computation of each of the k components of ρ before f can be done by the appropriate party by with no added communication between the parties, or loss in success probability. Additionally, we are using the fact that we have defined $\Pi[k,c]$ so that for each function in the class, and every partition of the inputs, there is a protocol that transmits at most c bits.

On the other hand,

$$\mathop{\mathbb{E}}_{\sigma\sim H_{\{0\}^k}}\left[\prod_{i\in[2^k]}\sigma_i\right] = \mathop{\mathbb{E}}_{\sigma\sim\{-1,1\}^{2^k}}\left[\prod_{i\in[2^k]}\sigma_i\right] = 0$$

Proceeding by a hybrid argument, it is then the case that for random hybrid neighbors H_j, H_{j+1} ($j \in \{0,1\}^k$, with $+$ indicating integer addition),

$$\mathop{\mathbb{E}}_{j\sim\{0,1\}^k}\left[\mathop{\mathbb{E}}_{\sigma\sim H_{j+1}}\left[\prod_{i\in[2^k]}\sigma_i = 1\right] - \mathop{\mathbb{E}}_{\sigma\sim H_j}\left[\prod_{i\in[2^k]}\sigma_i = 1\right]\right] \geq \gamma\cdot 2^{-c2^k}\cdot 2^{-k} \tag{7}$$

Observe that, for $z\sim\rho$, the circuit h output by $\mathscr{P}^{f,\rho}$, by definition, outputs the value

$$h(z) = v\cdot v'\cdot r_{\bar{b}} = r_{\bar{b}}\prod_{i\in[2^k]}\sigma_i$$

where $\sigma\sim H_{\bar{b}}$ and $\bar{b}\sim U_k$. Hence, we interpret $h(z)$ as a prediction, where $r_{\bar{b}}$ is the "guess bit."

To ease notation, let $R = (z,r,b,x_1^0,x_1^1\cdots x_k^0,x_k^1)$. Conditioning on correctness of the guess bit,

$$\begin{aligned}\Pr_R\left[h(z)=f(z)\mid r_{\bar{b}}=f(z)\right] = \Pr_R\left[h(z)=f(z)\mid r_{\bar{b}}=f(z)\right]\cdot\Pr_R[r_{\bar{b}}=f(z)]\\ +\Pr_R[h(z)=f(z)\mid r_{\bar{b}}\neq f(z)]\cdot\Pr_R[r_{\bar{b}}\neq f(z)]\end{aligned}$$

Then by making the appropriate substitution, we obtain

$$\begin{aligned}\Pr_R\left[h(z)=f(z)\mid r_{\bar{b}}=f(z)\right] = \frac{1}{2}\Pr_R\left[r_{\bar{b}}\prod_{i\in[2^k]}\sigma_i = f(z)\mid r_{\bar{b}}=f(z)\right]\\ +\frac{1}{2}\Pr_R\left[r_{\bar{b}}\prod_{i\in[2^k]}\sigma_i = f(z)\mid r_{\bar{b}}\neq f(z)\right]\end{aligned}$$

Indeed, when $\prod_{i\in[2^k]}\sigma_i = 1$ ($\sigma\sim H_{\bar{b}}$), this means that $h(z) = r_{\bar{b}}$. The case analysis then follows:

$$\begin{aligned}\Pr_R\left[h(z)=f(z)\right] &= \frac{1}{2}\left(\Pr_R\left[\prod_{i\in[2^k]}\sigma_i = 1\mid r_{\bar{b}}=f(z)\right]+\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\mid r_{\bar{b}}\neq f(z)\right]\right)\\ &= \frac{1}{2}+\frac{1}{2}\left(-\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\mid r_{\bar{b}}=f(z)\right]+\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\mid r_{\bar{b}}\neq f(z)\right]\right)\end{aligned}$$

By conditioning we know that:

$$\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\right] = \frac{1}{2}\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\mid r_{\bar{b}}=f(z)\right]+\frac{1}{2}\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\mid r_{\bar{b}}\neq f(z)\right]$$

rearranging the terms, we get:

$$\frac{1}{2}\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\mid r_{\bar{b}}\neq f(z)\right] = \Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\right]-\frac{1}{2}\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\mid r_{\bar{b}}=f(z)\right]$$

We thus conclude that:

$$\Pr_R[h(z)=f(z)] = \frac{1}{2}+\underbrace{\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\right]}_{(1)}-\underbrace{\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=-1\mid r_{\bar{b}}=f(z)\right]}_{(2)}$$

The term (1) corresponds the the case of taking the probability of the parity of a sample from $H_{\bar{b}}$ and the result being -1, while the term (2) corresponds analagously to the case of taking the parity of a sample from $H_{\bar{b}+1}$.

$$\begin{aligned}\Pr_R[h(z)=f(z)] &= \frac{1}{2}+\left(1-\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=1\right]\right)-\left(1-\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=1 \mid r_{\bar{b}}=f(z)\right]\right)\\ &= \frac{1}{2}-\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=1\right]+\Pr_R\left[\prod_{i\in[2^k]}\sigma_i=1 \mid r_{\bar{b}}=f(z)\right]\end{aligned}$$

Therefore, by equation (7),

$$\Pr\left[h(z)=f(z)\right] \geq \frac{1}{2}+\gamma\cdot 2^{-c2^k}\cdot 2^{-(k+1)}$$

■

We will now prove that the query complexity of $\mathscr{P}^{f,\rho}$ is as needed.

Lemma 15 (Query complexity of $\mathscr{P}^{f,\rho}$) *$\mathscr{P}^{f,\rho}$ makes at most $2k\cdot 2^{n-n/k}$ queries.*

Proof We can (roughly) bound the number of queries q as a function of n,k as follows. Observe that, for any random choices $X=x_1^0,x_1^1,\cdots x_k^0,x_k^1,b$ of $\mathscr{P}^{f,\rho}$, the queries by $\mathscr{P}^{f,\rho}$ are a subset of union of the k "block restrictions"

$$\{\underbrace{**\cdots*}_{k-1\text{ times}}x_k^b\}\cup\{\underbrace{**\cdots*}_{k-2\text{ times}}x_{k-1}^b*\}\cup\cdots\cup\{x_1^b\underbrace{**\cdots*}_{k-1\text{ times}}\}$$

Here, $\{**\cdots*\,x_k^b\}$ is notation for the k^{th} block restriction which is the set of n-bit points that are consistent with the k^{th} block set to x_k^b. We call x_k^b the *representative* of $\{**\cdots*\,x_k^b\}$. It is easy to see that the size of the union set of these k "block" restrictions is upper bounded by $k2^{n-n/k}$. Now, since $\mathscr{P}^{f,\rho}$ is supposed to repeat the queries for all $a<\bar{b}$, actually we can see that we need the union set

$$\bigcup_{b\in\{0,1\}^k}\{\underbrace{**\cdots*}_{k-1\text{ times}}x_k^{b_k}\}\cup\{\underbrace{**\cdots*}_{k-2\text{ times}}x_{k-1}^{b_{k-1}}*\}\cup\cdots\cup\{x_1^{b_1}\underbrace{**\cdots*}_{k-1\text{ times}}\}$$

This amounts to at most $2k\cdot 2^{n-n/k}$ queries. ■

Finally, we bound the time complexity of $\mathscr{P}^{f,\rho}$.

Lemma 16 (Time complexity) *$\mathscr{P}^{f,\rho}$ runs in time $O(2k\cdot 2^{n-n/k})$.*

Proof It suffices to observe that the complexity of $\mathscr{P}^{f,\rho}$ is dominated by the generation and memorization of the queries, as well as the construction of the hypothesis circuit. Each of these tasks is completed in time $O(2k\cdot 2^{n-n/k})$, so the lemma follows. ■

Towards Theorem 9. In order to get a $(\gamma, \gamma \cdot 2^{-c2^k-k})$-weak agnostic learner for $\Pi[k,c]$, we need to generalize the oracle algorithm $\mathscr{A}^{f,\rho}$ we got by Theorem 11 to work for arbitrary (probabilistic) concepts.

To do this, we first show that $\mathscr{A}^{f,\rho}$ queries f entirely at distinct points. In most time complexity regimes this would be trivial, since f is a function and no new information is gained by making a duplicate query. However, when time complexity is just barely nontrivial, we need to take care that we can actually efficiently build a lookup table with no duplicates.

Lemma 17 *$\mathscr{A}^{f,\rho}$ need not make any duplicate queries.*

Proof Let us first inspect the queries made by the first run of $\mathscr{P}^{f,\rho}$. We observe that, when querying the union of block restrictions

$$\bigcup_{b\in\{0,1\}^k} \{\underbrace{**\cdots*}_{k-1 \text{ times}} x_k^{b_k}\} \cup \{\underbrace{**\cdots*}_{k-2 \text{ times}} x_{k-1}^{b_{k-1}} *\} \cup \cdots \cup \{x_1^{b_1} \underbrace{**\cdots*}_{k-1 \text{ times}}\}$$

it is possible to avoid duplicate queries in an online way: start with $\{**\cdots* x_k^0\}$ and $\{**\cdots* x_k^1\}$, and then, for $\{**\cdots x_{k-1}^0 *\}$ and $\{**\cdots x_{k-1}^1 *\}$ skip any element that has postfix x_k^1 or x_k^0 (for each element, this can be checked in time $O(kn)$). Repeat this process for each remaining pair of block restrictions to avoid duplicates, while maintaining time complexity $O(2^{n-n/k+\log(k)})$.

We need to consider what happens in each subsequent run of $\mathscr{P}^{f,\rho}$ executed by $\mathscr{A}^{f,\rho}$. Again, by tracking the list of representatives chosen in previous runs, duplicates can be efficiently avoided. There are $O(\alpha^{-4})$ runs, so at most the complexity of scanning the list of previous representatives is $O(\alpha^{-4} \cdot kn)$. Therefore time complexity of $\mathscr{A}^{f,\rho}$ remains asymptotically the same even after incorporating the duplicate-scanning procedure. ■

We are finally in position to argue that we have a weak agnostic learner for $\Pi[k,c]$. The main point is that the learning algorithm we have so far considered, $\mathscr{A}^{f,\rho}$, does not consider probabilistic concepts. Probabilistic concepts are not functions, as they may label the same point differently depending on some internal randomness.

Theorem 18 (Theorem 9 restated) *There exists a $(\gamma, \gamma \cdot 2^{-c2^k-k})$-weak agnostic learner for $\Pi[k,c]$. The learner uses membership queries and learns over any $\rho \in \Delta_\sigma^k$ (for some fixed σ), and has*

- *query complexity $q := 2^{n-s(n)}$, for $s(n) = n/k - \log(k) + 4(\log(\gamma) - c2^k + k))$*
- *time complexity $t := O(q)$.*

Proof For weak agnostic learning, the target concept is an arbitrary distribution over $\{0,1\}^n \times \{-1,1\}$. This is more general than the idea of a concept being an arbitrary Boolean function that labels points, because now the concept may be probabilistic (i.e., not the same label at each point, every time). We need to argue that, for $0 < \gamma < 1$, a there is an algorithm that outputs a hypothesis $h : \{0,1\}^n \to \{-1,1\}$ such that

$$\max_{\pi\in\Pi[k,c]} \mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [\pi(x)y] \geq \gamma \implies \Pr_{\mathscr{A}} \left[\mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [h(x)y] \geq \gamma \cdot 2^{-c2^k-k} \right] \geq 2/3$$

The algorithm from Theorem 11 suffices. This is because, by Lemma 17, the target concept might as well have been deterministic – there are no duplicate queries. Hence, Theorem 11, in combination

with Lemma 17, implies that for $0 < \gamma < 1$, we get an algorithm $\mathscr{A}^{\mathscr{D}}$ that uses membership query and random example access to $\mathscr{D}$ to simulate both f and the marginal distribution ρ, and outputs $h : \{0,1\}^n \to \{-1,1\}$ such that

$$\max_{\pi \in \Pi[k,c]} \mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [\pi(x)y] \geq \gamma \implies \Pr_{\mathscr{A}} \left[\mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [h(x)y] \geq \gamma \cdot 2^{-c2^k - k} \right] \geq 2/3$$

■

Now that we have a weak agnostic learning algorithm, we can use distribution-specific agnostic boosting algorithms to obtain fully agnostic learning (Theorem 9). Distribution-specific boosting algorithms convert weak AMQ-learners into strong AMQ-learners, by deliberately modifying the labels of queries. This contrasts with other types of boosting algorithms, which instead modify the distribution over unlabelled examples. In our case, we need to use a distribution-specific boosting algorithm, because our weak learning algorithm does not work over all distributions.

The following theorem is a restatement of distribution-specific boosting, due to Feldman (2009a) and Kanade and Kalai (2009).

Theorem 19 (Distribution-specific agnostic boosting) *There exists an algorithm* boost *that for every concept class $\mathscr{C}$ and distribution ρ over $\{0,1\}^n$, given an (γ,α)-weak agnostic learning algorithm $\mathscr{A}$ for $\mathscr{C}$ over ρ, agnostically learns $\mathscr{C}$ over ρ. Further,* boost *invokes $\mathscr{A}$ $O(\alpha^{-2})$ times and runs in time $T \cdot \mathrm{poly}(\alpha^{-1}, \varepsilon^{-1})$, where T is the running time of $\mathscr{A}$.*

We combine this theorem with Theorem 9, to obtain the strong learning algorithm:

Theorem 20 *There exists an agnostic membership query learning algorithm for $\Pi[k,c]$. The algorithm learns over any $\rho \in \Delta^k_\sigma$ (for some fixed σ), and has*

- *query complexity $q := 2^{n-s(n)}$, for $s(n) = n/k - \log(k) + 4(\log(\gamma) - c2^k + k))$*
- *time complexity $t := O(q) \cdot \mathrm{poly}(\varepsilon^{-1})$.*

Here, $\gamma := \mathop{\mathrm{opt}}_{\mathscr{D}}(\Pi[k,c])$.

Proof The theorem statement follows immediately from Theorem 19 and 9. To remove the assumption that k dives n, simply consider that if it does not, then $k - (n \bmod k)$ meaningless variables can be added to the input of the function so that k divides n, but the functionality is unchanged. Therefore this will not affect the functionality of the learning algorithm, and the complexity is increased by at most a constant factor. ■

From here, we derive as immediate corollaries:

Theorem 21 *There exists an agnostic membership query learning algorithm for circuits consisting of t_1 gates, where each gate can compute a SYM^+ circuit of size t_2, and degree $k-1$ (i.e., the class SYM^+-$\mathsf{Ckt}[k-1,t_2,t_1]$). The algorithm learns over any $\rho \in \Delta^k_\sigma$ (for some fixed σ), and has*

- *query complexity $q := 2^{n-s(n)}$, for $s(n) = n/k - \log(k) + 4(\log(\gamma) - 2^k t_1 \log(t_2) + k))$*

- *time complexity* $t := O(q) \cdot \text{poly}(\varepsilon^{-1})$.

Here, $\gamma := \underset{\mathscr{D}}{\text{opt}}(\mathsf{SYM}^+\text{-}\mathsf{Ckt}[k-1, t_2, t_1])$.

Proof SYM^+ circuits of size t_2 and degree $k-1$ are well-known to be a subset of $\Pi[k, \log(t_2)]$ (see e.g. Lemma 4 of Håstad and Goldmann (1991)). Then, by simulating a circuit of t_1 gates, each computing SYM^+ circuits of size t_2 and degree $k-1$, we can still get a protocol in the class $\Pi[k, t_1 \log(t_2)]$ (see also Nisan (1993), section 4). Recall, this means that there is a k-party deterministic protocol that communicates $t_1 \log(t_2)$ bits *for any* partition of the input into k parts. Then, the statement follows immediately from Theorem 20.

■

Theorem 22 *There exists an agnostic membership query learning algorithm for decision trees of depth d, with node queries computed by SYM^+ circuits of size t, and degree $k-1 \geq 1$ (i.e., the class $\mathsf{SYM}^+\text{-}\mathsf{Dt}[k-1, t, d]$). The algorithm learns over any $\rho \in \Delta_\sigma^k$ (for some fixed σ), and has*

- *query complexity* $q := 2^{n-s(n)}$, *for* $s(n) = n/k - \log(k) + 4(\log(\gamma) - 2^k d \log(t) + k))$
- *time complexity* $t := O(q) \cdot \text{poly}(\varepsilon^{-1})$.

Here, $\gamma := \underset{\mathscr{D}}{\text{opt}}(\mathsf{SYM}^+\text{-}\mathsf{Dt}[k-1, t, d])$.

Proof SYM^+ circuits of size t_2 and degree $k-1$ are well-known to be a subset of $\Pi[k, \log(t_2)]$ (see e.g. Lemma 4 of Håstad and Goldmann (1991)). Then, by simulating the decision tree of depth d, with SYM^+ queries of size t and degree $k-1$, we can still get a protocol in the class $\Pi[k, d \log(t)]$ (see also Nisan (1993), section 4). Then, the statement follows immediately from Theorem 20.

■

5. Learning Efficient Randomized k-Party NOF Communication Protocols

In this section, we will convert the agnostic learning algorithm for $\Pi[k, c]$ into an agnostic learning algorithm for the class of functions computable by *randomized* k-party communication protocols of cost c, $\text{r}\Pi[k, c]$. The set $\text{r}\Pi[k, c]$ trivially contains $\Pi[k, c]$, and may be more powerful. As a case in point, PTFs of degree k are not known to be computed by $\Pi[k, c]$, but *can* be computed by $\text{r}\Pi[k, c]$.

Definition 23 (Randomized $\Pi[k, c]$) *The randomized k-party communication model is the same as the deterministic model, except we now allow 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 $\text{r}\Pi[k, c, \alpha]$ the class of k-party randomized protocols that, for any partition of the n variables into k length n/k parts, transmit at most c bits and err with probability at most $1/2 - \alpha$. Additionally, we define $\text{r}\Pi[k, c, 1/6] := \text{r}\Pi[k, c]$ for shorthand.*

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. Again, we assume for now that k divides n.

To get started, we prove the following lemma which converts correlation with randomized protocols to correlation with deterministic protocols.

Lemma 24 *Let* $f : \{0,1\}^n \to \{-1,1\}$. *If* $\mathrm{Cor}(f, \mathrm{r}\Pi[k,c]) \geq \alpha$, *then* $\mathrm{Cor}(f, \Pi[k, O(c\log(\alpha^{-1}))]) \geq \alpha/2$.

Proof Guaranteed by the condition of the lemma is the existence of $g : \{0,1\}^n \to \{-1,1\}$ such that $g \in \mathrm{r}\Pi[k,c]$ and $\mathrm{Cor}(f,g) \geq \alpha$. Without loss of generality assume g is the lexicographically first of all functions that satisfy $g \in \mathrm{r}\Pi[k,c]$ and $\mathrm{Cor}(f,g) \geq \alpha$. Let π be the protocol that witnesses the inclusion $g \in \mathrm{r}\Pi[k,c]$, and let π^m be the direct-majority protocol of π so as to reduce error to $1-\alpha/2$. We thus have that $g \in \mathrm{r}\Pi[k, O(c\log(\alpha^{-1})), \alpha/2]$ and $\mathrm{Cor}(f,g) \geq \alpha$.

Now, here is a deterministic protocol π^* that witnesses $\mathrm{Cor}(f, \Pi[k, O(c\log(\alpha^{-1}))]) \geq \alpha/2$. First, both parties find g, and compute the set T of inputs that f and g agree on. Then, they convert π^m to a distributional protocol $\tilde{\pi}^m$ for the uniform distribution over T (possible by an averaging argument), and run $\tilde{\pi}^m$ and output the result. Because $2^{-n}|T| \geq 1/2 + \alpha/2$, we get that $\mathrm{Cor}(f, \pi^*) \geq 2(1/2+\alpha/2)(1-\alpha/2) - 1 \geq (1+\alpha)(1-\alpha/2) - 1 \geq \alpha/2$. ∎

Now we can obtain a weak agnostic learning algorithm for $\mathrm{r}\Pi[k,c]$, as long as the distribution over unlabelled examples is uniform. Then boosting gives us a strong agnostic learning algorithm.

Theorem 25 *There exists an agnostic membership query learning algorithm for* $\mathrm{r}\Pi[k,c]$. *The algorithm learns over the uniform distribution, and has*

- *query complexity* $q := 2^{n-s(n)}$, *for* $s(n) = n/k - k - O(\log(\alpha^{-1}) + c2^k))$
- *time complexity* $t := O(q)$.

Here, $\gamma := \operatorname{opt}_{\mathscr{D}}(\mathrm{r}\Pi[k,c])$.

Proof We need to prove that there is a membership query algorithm $\mathscr{A}$ that outputs a hypothesis $h : \{0,1\}^n \to \{-1,1\}$ such that, for any $\mathscr{D}$ such that the distribution over unlabelled examples, ρ, is uniform, we have that

$$\max_{c \in \mathrm{r}\Pi[k,c]} \mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}}[c(x)y] \geq \alpha \implies \Pr_{\mathscr{A}}\left[\mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}}[h(x)y] \geq \gamma\right] \geq 2/3$$

If we do so, then we can invoke boosting (Theorem 19) to get an agnostic learning algorithm for $\mathrm{r}\Pi[k,c]$. Thus, we prove the following:

Lemma 26 *There exists a* $(\gamma, \gamma \cdot 2^{-O(c\log(\gamma^{-1}))2^k - k})$*-weak agnostic learner for* $\mathrm{r}\Pi[k,c]$. *The learner uses membership queries and learns over the uniform distribution over unlabelled examples, and has*

- *query complexity* $q := 2^{n-s(n)}$, *for* $s(n) = n/k - k - O(\log(\gamma^{-1}) + c2^k))$

- *time complexity* $t := O(q)$.

Proof By lemma 24, if

$$\max_{c \in \mathrm{r}\Pi[k,c]} \mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [c(x)y] \geq \gamma$$

then we know that

$$\max_{c \in \Pi[k,O(c\log(\gamma^{-1}))]} \mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [c(x)y] \geq \gamma/2$$

Then, by Theorem 9, we know that there is a $(\gamma, \gamma \cdot 2^{c2^k-k})$-weak agnostic learning algorithm for $\Pi[k,c]$. Therefore, we conclude that there is an $(\gamma, \gamma \cdot 2^{-O(c\log(\gamma^{-1}))2^k-k})$-weak agnostic learner $\mathscr{A}_{wk}$ for $\mathrm{r}\Pi[k,c]$:

$$\max_{c \in \mathrm{r}\Pi[k,c]} \mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [c(x)y] \geq \gamma \overset{\text{(Lemma 24)}}{\Longrightarrow}$$

$$\max_{c \in \Pi[k,O(c\log(\gamma^{-1}))]} \mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [c(x)y] \geq \gamma/2 \overset{\text{(Theorem 9)}}{\Longrightarrow}$$

$$\Pr_{\mathscr{A}_{wk}} \left[\mathop{\mathbb{E}}_{(x,y)\sim\mathscr{D}} [h(x)y] \geq \gamma \right] \geq 2/3$$

■

Finally, Theorem 19 and Lemma 26 combine to indicate that $\mathscr{A}_{wk}$ can be boosted to a strong agnostic learning algorithm, that has query complexity $q := 2^{n-s(n)}$, for $s(n) = n/k - k - O(\log(\gamma^{-1}) + c2^k))$ and time complexity $t := O(q)$. Recall, from Theorem 19, the weak learner is invoked $O(\gamma^{-2}$ times), so that is where the $-O(\log(\gamma^{-1}) + c2^k)$ additive term comes from. ■

5.1. Learning Circuits with Threshold Gates

In this section, we will derive AMQ-learners for circuits with threshold gates, and decision trees that are allowed threshold queries. To do this, we use the following randomized communication protocols for such function representations, due to Nisan (1993).

Theorem 27 (Nisan (1993)) *Let* $f : \{0,1\}^n \to \{-1,1\}$ *be a function.*

1. *If* f *can be computed by a circuit consisting of* s *gates, each computing a polynomial threshold function of degree* k*, then* $f \in \mathrm{r}\Pi[k+1, O(sk^3 \log(n)\log(sn))]$
2. *If* f *can be computed by a decision tree of depth* d*, with each node computing a polynomial threshold function of degree* k*, then* $f \in \mathrm{r}\Pi[k+1, O(dk^3 \log(n)\log(dn))]$.

From this theorem, combined with Theorem 25, we obtain the following as corollaries. Recall $\mathsf{PT\text{-}Ckt}[k,m]$ is the class of circuits consisting of at most m polynomial threshold gates of degree k, and $\mathsf{PT\text{-}Dt}[k,d]$ is the class of decision trees consisting of at most depth d, with internal nodes computing polynomial threshold gates of degree k.

Theorem 28 *There exists an agnostic membership query learning algorithm for* $\mathsf{PT\text{-}Ckt}[k,m]$*. The algorithm learns over the uniform distribution, and has*

- *query complexity* $q := 2^{n-s(n)}$, *for* $s(n) = n/(k+1) - O(\log(\gamma^{-1}) + 2^k \cdot m \log(n) \log(mn))$
- *time complexity* $t := O(q)$.

Here, $\gamma := \operatorname{opt}_{\mathscr{D}}(\mathsf{PT\text{-}Ckt}[k,m])$.

Proof Immediate, from Theorem 25 and 27. ■

Theorem 29 *There exists an agnostic membership query learning algorithm for* $\mathsf{PT\text{-}Dt}[k,d]$. *The algorithm learns over the uniform distribution, and has*

- *query complexity* $q := 2^{n-s(n)}$, *for* $s(n) = n/(k+1) - O(\log(\gamma^{-1}) + 2^k \cdot d \log(n) \log(dn))$
- *time complexity* $t := O(q)$.

Here, $\gamma := \operatorname{opt}_{\mathscr{D}}(\mathsf{PT\text{-}Dt}[k,d])$.

Proof Immediate, from Theorem 25 and 27. ■

Remark. By setting $m := n^{0.99}$, $d := n^{0.99}$, $k := O(1)$, and $\gamma := O(2^{n^{0.99}})$, we recover the statements of the AMQ-learners for touchstone class 1 and 2, respectively, which were presented in Section 1.1.

6. Randomized Compression, Exact Learning, Distribution-Independent Learning

6.1. Randomized Boolean Function Compression

In this section, we show how the AMQ-learners we have obtained so far can be converted into *randomized* Boolean function compression algorithms. This is done via a simple technique of Carmosino et al. (2016), where a PAC-learning algorithm (optionally using membership queries) which learns over the uniform distribution is used to obtain a *lossy* compression algorithm, and then a linear scanning procedure is used to fix the mistakes.

Definition 30 (Compression – written essentially as in Oliveira and Santhanam (2016)) *Given a circuit class* $\mathscr{C}$, *a compression algorithm for* $\mathscr{C}$ *is an algorithm* $\mathscr{A}$ *for which the following hold:*

- *Given an input* $z \in \{0,1\}^{2^n}$, $\mathscr{A}$ *outputs a circuit* C *of size* $o(2^n/n)$ *such that if the function* f_z *represented by the truth table* z *is such that* $f_z \in \mathscr{C}$, *then* C *computes* f_z.
- $\mathscr{A}$ *runs in time* $\operatorname{poly}(2^n)$.

We say $\mathscr{C}$ *is compressible if there is a (polynomial time) compression algorithm for* $\mathscr{C}$. *If the algorithm* $\mathscr{A}$ *is probabilistic, and produces a correct circuit with probability at least* $2/3$ *over its random choices, then* $\mathscr{C}$ *is probabilistically compressible.*

Remark. One note about the above definition: The constant 2/3 as the desired success probability is not particularly important. Since the run-time needs to be $2^{O(n)}$, i.e., polynomial in the input length of 2^n, a randomized algorithm can check for success in linear time. Hence, one could change 2/3 to be $1-2^{-n}$ if desired.

Theorem 31 *There exists a sufficiently large constant K such that the following complexity classes are probabilistically compressible:*

1. PT-Ckt$[k,m]$, *whenever* $m\log(m)\log(mn)\cdot 2^k \leq n/K(k+1)$
2. PT-Dt$[k,d]$, *whenever* $d\log(d)\log(dn)\cdot 2^k \leq n/K(k+1)$
3. SYM$^+$-Ckt$[k,t,m]$, *whenever* $m\log(t)\cdot 2^k \leq n/K(k+1)$
4. SYM$^+$-Dt$[k,t,d]$, *whenever* $d\log(t)\cdot 2^k \leq n/K(k+1)$

Proof Given an AMQ-learner for a touchstone class $\mathscr{C}$, which runs in time $T(n,\varepsilon^{-1},\delta^{-1})$, we can transform it to a randomized compression algorithm for $\mathscr{C}$. The randomized compression algorithm is as follows. Given the truth table $T\in\{0,1\}^{2^n}$ as input, use it to simulate a membership query oracle in order to run the AMQ-learner, with $\varepsilon := 1/2^{n^{0.99}}$, $\delta := 2/3$, on the concept defined by T. Let h be the output of the AMQ-learner. Now, compare the truth table indicated by h to T, and, on whatever points they disagree, hard-wire that input/output pair into h. Output the amended h.

Let us analyze this algorithm. We upper bound the runtime of the learner and the size of the output circuit. Since we are aiming for a randomized compression, we only need to consider the case that the AMQ-learner outputs a hypothesis with the desired error, which happens with probability at least 2/3. The circuit size of the hypothesis is at most the running time of the AMQ-learner, so at most $T(n,\varepsilon^{-1},\delta^{-1})$. Then, after the amendment procedure, the circuit adds $O(\varepsilon 2^n)$ size. Hence, overall, the runtime and size of the output circuit of the randomized compression algorithm is $T(n,\varepsilon^{-1},\delta^{-1})+O(\varepsilon 2^n)$.

Finally, we can can invoke this transformation with a correct setting of ε as a function of n, using the AMQ-learners for each of the classes 1-4. By Theorems 28, 29, 21 and 29, setting $\varepsilon := 2^{-n^{0.99}}$, we obtain randomized compression algorithms for 1-4, respectively. The compressed circuit has size $O(2^{n-n^{0.99}})$ in all cases, because the AMQ-learner for each class has run-time bounded above by $2^{n/C}\in O(2^{n-n^{0.99}})$ for some constant $C>1$, depending only on k and K, and using the conditions stated for each class in the theorem. ■

6.2. Exact Learning with Membership and Equivalence Queries

In the final section, we explain that our AMQ-learners can be converted into learning algorithms in the exact learning with membership and equivalence queries model Angluin (1988), as well as learning algorithms in the distribution independent PAC model, with membership queries.

Let us start with a description of the exact learning with membership and equivalence queries model. Let $f:\{0,1\}^n\to\{-1,1\}$ be an unknown concept. Let $\mathscr{C}$ be a fixed *concept class*, and assume $f\in\mathscr{C}$. The learning algorithm aims to output a hypothesis $h:\{0,1\}^n\to\{-1,1\}$ such that for every $x\in\{0,1\}^n$, $h(x)=f(x)$. The learning algorithm does this by frequently updating the hypothesis through a number of timesteps, where at each timestep, the learning algorithm makes one of two types of oracle queries:

- A membership query: the learning algorithm selects $q \in \{0,1\}^n$ and obtains $f(q)$.
- An equivalence query: the learning algorithm presents the current hypothesis h', and receives either "success" (when $h'(x) = f(x)$ for all x), or a counterexample z such that $h'(z) \neq f(z)$.

The running time of the algorithm is the number of timesteps until the algorithm obtains the "success" response, in the worst case. In the randomized version, which we now consider exclusively, the running time of the algorithm is considered $T(n)$ if it the algorithm succeeds with probability $1-\delta$ and runs in time no more than $T(n) \cdot \log(\delta^{-1})$.

Theorem 32 *There exists a sufficiently large constant K such that the following complexity classes are learnable in the randomized exact learning with membership and equivalence queries model, in time* $\mathrm{poly}(n) \cdot 2^{n-n^{0.99}}$.

1. $\mathsf{PT\text{-}Ckt}[k,m]$, *whenever* $m \log(m) \log(mn) \cdot 2^k \leq n/K(k+1)$
2. $\mathsf{PT\text{-}Dt}[k,d]$, *whenever* $d \log(d) \log(dn) \cdot 2^k \leq n/K(k+1)$
3. $\mathsf{SYM^+\text{-}Ckt}[k,t,m]$, *whenever* $m \log(t) \cdot 2^k \leq n/K(k+1)$
4. $\mathsf{SYM^+\text{-}Dt}[k,t,d]$, *whenever* $d \log(t) \cdot 2^k \leq n/K(k+1)$

Proof The proof flows identically to the proof of Theorem 31, except instead of doing a linear scan of the input truth table to amend mistakes, we invoke the equivalence query oracle to obtain a point z that needs to be amended (by negating the existing value). It takes at most $\mathrm{poly}(n)$ time to amend each mistake. One can reduce the failure property at an exponential rate via testing and repeating, so the randomized algorithm incurs only a multiplicative factor of $\log(\delta^{-1})$ in runtime. ■

6.3. Distribution Independent PAC-Learning with Membership Queries

It is well known since Angluin (1988) that a learning algorithm in the exact learning with membership and equivalence queries model implies a distribution independent PAC-learning algorithm, which also uses membership queries. The basic idea is that equivalence queries can be simulated, up to some small probability of an incorrect answer, with random queries. One can do this by choosing sufficiently many random queries, and realizing that if the current hypothesis is more than ε-far from the unknown concept, then the random sample should contain a counterexample with high probability (on the other hand, if the current hypothesis is more than ε-far from the unknown concept, then the algorithm can terminate since we are in the PAC model of learning).

Quantitatively, one can transform an exact learning with membership and equivalence queries algorithm for a concept class $\mathscr{C}$ that runs in time $T(n)$ and uses $Q(n)$ queries into a distribution-independent PAC-learning with membership queries algorithm that runs in time $O(T(n)/\varepsilon \cdot \ln T(n)/\delta)$ and uses $O(Q(n)/\varepsilon \cdot \ln Q(n)/\delta)$ queries (where PAC-identification occurs with accuracy $1-\varepsilon$ and confidence $1-\delta$). We refer the reader to Servedio and Tan (2017) and Section 2.4 of Angluin (1988) for further details.

We present the following without formal proof.

Theorem 33 *There exists a sufficiently large constant K such that the following complexity classes are learnable, to accuracy $1-\varepsilon$ and confidence $1-\delta$, in the distribution-independent PAC-learning with membership queries model, in time* $\mathrm{poly}(n) \cdot 2^{n-n^{0.99}}/(\varepsilon\delta)$*:*

1. PT-Ckt$[k,m]$, *whenever* $m\log(m)\log(mn)\cdot 2^k \leq n/K(k+1)$
2. PT-Dt$[k,d]$, *whenever* $d\log(d)\log(dn)\cdot 2^k \leq n/K(k+1)$
3. SYM$^+$-Ckt$[k,t,m]$, *whenever* $m\log(t)\cdot 2^k \leq n/K(k+1)$
4. SYM$^+$-Dt$[k,t,d]$, *whenever* $d\log(t)\cdot 2^k \leq n/K(k+1)$

Acknowledgments

I'm grateful to Marco Carmosino for insightful discussions relating to this line of work. Part of this work was completed while I was visiting the Simons institute for the theory of computing. This work was supported by the DARPA SIEVE program, Agreement Nos. HR00112020021 and HR00112020023.

References

Dana Angluin. Queries and concept learning. *Machine learning*, 2:319–342, 1988.

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

Richard Beigel and Jun Tarui. On acc. *Computational Complexity*, 4:350–366, 1994.

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.

Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.

Avrim L Blum. Separating distribution-free and mistake-bound learning models over the boolean domain. *SIAM Journal on Computing*, 23(5):990–1000, 1994.

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.

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.

Marco L Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Agnostic learning from tolerant natural proofs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

Ruiwen Chen, Valentine Kabanets, Antonina Kolokolova, Ronen Shaltiel, and David Zuckerman. Mining circuit lower bound proofs for meta-algorithms. *Computational Complexity*, 24:333–392, 2015.

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

Vitaly Feldman. Distribution-specific agnostic boosting. *arXiv preprint arXiv:0909.2927*, 2009a.

Vitaly Feldman. On the power of membership queries in agnostic learning. *The Journal of Machine Learning Research*, 10:163–182, 2009b.

Fedor V Fomin and Petteri Kaski. Exact exponential algorithms. *Communications of the ACM*, 56 (3):80–88, 2013.

Sally A Goldman, Stephen S Kwek, and Stephen D Scott. Agnostic learning of geometric patterns. In *Proceedings of the tenth annual conference on Computational learning theory*, pages 325–333, 1997.

Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32, 1989.

Parikshit Gopalan, Adam Tauman Kalai, and Adam R Klivans. Agnostically learning decision trees. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 527–536, 2008.

Johan Håstad and Mikael Goldmann. On the power of small-depth threshold circuits. *Computational Complexity*, 1:113–129, 1991.

David Haussler. Decision theoretic generalizations of the pac model for neural net and other learning applications. *Information and Computation*, 100:78–150, 1992.

Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for ac0. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 961–972. SIAM, 2012.

Svante Janson, Tomasz Luczak, and Andrzej Rucinski. *Random graphs*. John Wiley & Sons, 2011.

Adam Tauman Kalai, Adam R Klivans, Yishay Mansour, and Rocco A Servedio. Agnostically learning halfspaces. *SIAM Journal on Computing*, 37(6):1777–1805, 2008a.

Adam Tauman Kalai, Yishay Mansour, and Elad Verbin. On agnostic boosting and parity learning. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 629–638, 2008b.

Varun Kanade and Adam Kalai. Potential-based agnostic boosting. *Advances in neural information processing systems*, 22, 2009.

Ari Karchmer. Distributional pac-learning from nisan's natural proofs. *arXiv preprint arXiv:2310.03641*, 2023.

Michael J Kearns, Robert E Schapire, and Linda M Sellie. Toward efficient agnostic learning. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 341–352, 1992.

Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 455–464, 1991.

Eyal Kushilevitz and Noam Nisan. Communication complexity, 1996.

Wee Sun Lee. Efficient agnostic learning of neural networks with bounded fan-in. *IEEE Transactions on Information Theory*, 42(6):2118–2132, 1996.

Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine learning*, 2:285–318, 1988.

Vadim Lyubashevsky. The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 378–389. Springer, 2005.

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

Igor C Oliveira and Rahul Santhanam. Conspiracies between learning algorithms, circuit lower bounds and pseudorandomness. *arXiv preprint arXiv:1611.01190*, 2016.

Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. In *Proceedings 38th Annual Symposium on Foundations of Computer Science*, pages 566–574. IEEE, 1997.

Ramamohan Paturi, Pavel Pudlák, Michael E Saks, and Francis Zane. An improved exponential-time algorithm for k-sat. *Journal of the ACM (JACM)*, 52(3):337–364, 2005.

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

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

T Schoning. A probabilistic algorithm for k-sat and constraint satisfaction problems. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, pages 410–414. IEEE, 1999.

Rainer Schuler. An algorithm for the satisfiability problem of formulas in conjunctive normal form. *Journal of Algorithms*, 54(1):40–44, 2005.

Rocco A Servedio and Li-Yang Tan. What circuit classes can be learned with non-trivial savings? In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

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

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.